AXI for MAXAda Reference Manual

0890518-000
November 1997

The information contained in this document is believed to be correct at the time of publication. It is subject to change without notice. Concurrent makes no warranties, expressed or implied, concerning the information contained in this document.

To report an error or comment on a specific portion of the manual, photocopy the page in question and mark the correction or comment on the copy. Mail the copy (and any additional comments) to Concurrent Computer Corporation, 2101 W. Cypress Creek Road, Ft. Lauderdale, FL 33309–1892. Mark the envelope **"Attention: Publications Department**." This publication may not be reproduced for any other reason in any form without written permission of the publisher.

Printed in U. S. A.

Revision History:	Level:	Effective With:
Original Release -- November 1997	000	PowerMAX OS 4.1

Preface

Scope of Manual

This manual is intended for use by Ada programmers interfacing to the X Window System™ and Motif™ Window Manager using Series 6000 or Power Hawk™ Systems and MAXAda™.

The Ada X Interface for MAXAda (AXI for MAXAda™) provides complete functionality from the X Window System and the Motif Window Manager.

Structure of Manual

This manual consists of thirteen chapters and two appendixes. A brief description of the chapters and appendixes follows:

- Chapters 1-12 describe the Xlib, X Toolkit, and Motif interfaces.
- Chapter 13 describes the STARS binding to the Xlib and Xt routines.
- Appendix A lists some of the complex data type definitions included in the packages Xlib, Xt, and Xm.
- Appendix B provides Concurrent-specific information.

Syntax Notation

The following notation is used throughout this guide:

italic	Books, reference cards, and items that the user must specify appear in *italic* type. Special terms may also appear in *italic*.
`list bold`	User input appears in **`list bold`** type and must be entered exactly as shown. Names of directories, files, commands, options and man page references also appear in **`list bold`** type.
`list`	Operating system and program output such as prompts and messages and listings of files and programs appears in `list` type.
`[ ]`	Brackets enclose command options and arguments that are optional. You do not type the brackets if you choose to specify such options or arguments.

Related Publications

The following publications are related to this document:

0890516	MAXAda Reference Manual
0890299	X Toolkit Programming Manual
0890300	X Window System User's Guide
0890380	OSF/Motif Documentation Set
0890395	NightView™ User's Guide
0890429	PowerMAX OS 4.1 System Administration Manual, Volume 1
0890430	PowerMAX OS 4.1 System Administration Manual, Volume 2

The Definitive Guides to the X Window System for Version 11, Volumes 1 - 7, O'Reilly and Associates, Sebastopol, CA, 1990.

Xlib - C Language X Interface, by Jim Gettys, Ron Newman and Robert Scheifler. Massachusetts Institute of Technology, Cambridge, MA, 1990. This document is a part of the X Version 11 distribution from MIT.

OSF/Motif Programmer's Reference Manual. Open Software Foundation, Cambridge, MA, 1990.

Contents

1 Introduction

2 Programming Considerations

3 Convenience Functions

4 Xlib Call Reference

5 X Resource Manager

6 X Color Management System

7 X Wide Character Strings

8 X Server Extensions

9 X Toolkit Intrinsics Interface

10 X Miscellaneous Utility Routines

11 Motif Interface

12 Motif Resource Manager

13 STARS Binding Routines

A Data Types

B Concurrent-Specific Information

1

Introduction

Overview

AXI for MAXAda, the Ada X Interface for MAXAda provides the Ada programmer complete functionality from the X Window System and the Motif Window Manager. The interface is intended to be as close as possible to the C Language X Interface syntax as defined and documented by the X Consortium. This allows programmers familiar with the X Window System to easily start using X facilities from Ada. In addition, programmers may use existing documentation such as the O'Reilly X Manuals to determine the exact syntax and behavior of a function.

This manual describes how to use AXI for MAXAda in your Ada program. The general syntax and conventions for the Ada Interface are described, as well as differences from the standard Xlib Interface. It is not intended to be a complete reference to any portion of the X Window System or Motif.

Introduction

This chapter provides a brief introduction to the X Window System™ and Motif™. The X Window System was developed jointly by MIT's Project Athena and Digital Equipment Corporation, and is now maintained by the X Consortium. Motif is a Window Manager that runs as a client of the X Window System. Motif is developed and maintained by Open Software Foundation.

The focus of this manual is AXI for MAXAda (Ada X Interface for MAXAda), an Ada interface to the X Library (Xlib), X Toolkit (Xt) and Motif Toolkit (Xm) written in C. Using this interface, Ada software can take full advantage of the X Library, X Toolkit and Motif. Because AXI closely follows the configuration of Xlib, Xt and Xm written in C, it is possible to rewrite Xlib/Xt and Xm applications written in C to Ada, and also to link C and Ada code into a single executable using the same Xlib, Xt and Xm libraries.

This manual does not substitute documentation about X and Motif or the capabilities of individual libraries. It assumes the user has some familiarity with X and programming in the X environment.

A Quick Tour of X11 and Motif 1.1

The X Window System is a windowing system for bit-mapped raster workstations and display devices. X includes support for multiple screens (monochrome, gray-scale and color), a keyboard and a pointing device (mouse, trackball or similar device).

X11R6 denotes version 11, release 6 of the X Window System. X11R6 is required for AXI for MAXAda.

Whereas X simply provides the base functionality needed to divide a display into windows, a window manager provides additional functionality in the form of standardized look and feel, icons, widgets and other features of convenience to a user. Motif 1.1 denotes release 1.1 of the Motif window manager. AXI for MAXAda supports Motif 1.1.

The Client-Server Model

The X Window System is primarily a network-based protocol definition. Being network-based, X allows an application program to output and control windows on a display different from the one on which it is invoked. Many applications run on the local display, but others may engage in X protocol requests across a network to either send information to another workstation, or receive and process requests from another workstation.

An X server is a program that runs on a workstation to control the display screen(s) on that workstation. The X Client-Server terminology is opposite in sense to the traditional meaning whereby a server is perhaps a remote program controlling access to some shared resource. File servers are an example of the traditional sense.

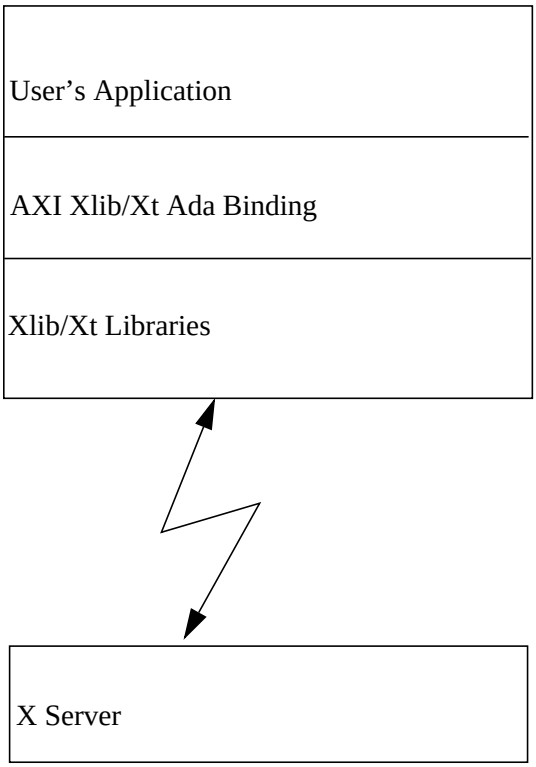
User programs that make use of X facilities are denoted X clients. The X server acts as a gateway between an X client and the physical resources on a display device (screen, keyboard and a pointing device).

X Protocol

In order to support a heterogeneous set of workstations and displays in a network, a layered network protocol forms the basis of the X Window System. This protocol specifies the contents of “packets” that are routed over the network. The X protocol defines the communication paradigm between a server and a client. The protocol supports the routing of the following types of information: requests, replies, events and errors.

X Library

The X library, or Xlib, as it is more commonly known, is a C language interface to the X Protocol. Xlib generates and sends protocol requests to the server and receives replies to some requests from the server. The AXI Ada Binding provides an interface to the C language Xlib, which in turn interfaces to the X server:



Xlib contains a collection of routines that deal with color, cursors, drawing shapes, fonts, keyboard, mouse,

errors and so on.

Intrinsics and Toolkits

Layers written directly on top of Xlib are known as toolkits. A commonly used toolkit is Xt (the X Intrinsics), which is provided by MIT. In addition, AXI also supports Xm, the Motif toolkit.

Widgets

Toolkits provide a simplified and standardized approach to graphics user interface (GUI) programming by supplying ready-made GUI elements such as buttons, labels, dialog boxes, scrollbars, and so on. These toolkit GUI elements are commonly referred to as widgets. For most GUI programmers, widgets are meant to be black boxes with well defined input and output, but no internal details. The Intrinsics libraries make it possible for a programmer to design and implement custom widgets.

The Xt and Xm toolkits provide the capability to create and use widgets. It is possible to write X applications using the Xlib interface exclusively. However, the higher-level widget interfaces make X programming much easier.

Widget Classes

Widgets are organized into classes. Each member of a class has certain features common to all instances of that class. These classes exist in a hierarchy. Some widget classes are inherited from classes higher in the hierarchy (superclasses). Other features are new to a particular class and will be inherited by any class lower in the hierarchy (subclasses). When creating an instance of a particular widget, the programmer must specify the class name of the desired widget.

Resources

Many widget features are customizable from resources (name-value pairs). These resources are typically stored in text files (~/.Xdefaults or /usr/lib/X11/app-defaults/*application-name*). These files are read by the X resource manager when the application is invoked. The resources control how the widgets look and behave. Typical resources control the foreground color, background color, x- and y-coordinates, border width, and so on. The resource concept is central to programming with Xt. For more information on resources, refer to Volume 4 of the O'Reilly and Associates series *The X Window System*.

Linking Widgets to an Application

Widgets are largely independent of the application. Xt dispatches events to the widgets (not the application) which processes events independent of the application. However, the widget may invoke sections of the application's code through *callbacks, actions* and *event handlers*:

- Callback routines are called when predefined callback conditions are met. These conditions are defined in the widget and may or may not be in response to events.
- Action routines are called in response to events specified in a user-defined translation table.
- Event handlers are called in response to a specified event selection mechanism.

The distinction between callbacks and actions is a little vague. A callback routine may be thought of as implementing a single application feature. An action table can be thought of as implementing a single widget feature. These two mechanisms differ in the way they are registered with Xt.

AXI Ada Bindings

There are several packages included in the AXI libraries. Each package provides access to a different part of the X Window System and Motif. The grouping of functions in packages follows from their organization

in various C archives in the MIT distribution. Package names are derived from common function name prefixes. For example, the AXI equivalent of the X Toolkit Intrinsics function XtOpenDisplay is in package **Xt** and is referenced as Xt.OpenDisplay. The naming of the **Xlib** package is the only exception to this. Whereas the intuitive name of this package would be **X**, the package has been named **Xlib** in order to allow the use of variables named x in client code.

The packages included and their contents are described below:

- **Package Xlib** - This package provides access to all of the functions in the X Window System Library Interface. The syntax for these functions is described in Chapter 4 of this manual. The functions themselves are described in detail in *The Definitive Guide to The X Window System, Volume Two, Xlib Reference Manual* by O'Reilly and Associates. All functions in this library begin with the prefix X.
- **Package XK** - This package provides definitions for all of the Keysyms in the X Window System. In the C interface, these constants are accessed as XK_name. In AXI, these constants have all been placed incorporated into a single package and are referenced as XK.K_name. The K_ was suffixed after the package name to eliminate any conflicts with reserved words and keysyms that start with a number.
- **Package Xrm** - This package provides access to the X Resource Manager data base. In the C Interface these routines are part of Xlib. In AXI, Resource Manager functions have been separated out into their own package. The syntax for these functions is described in Chapter 5 of this manual. The functions themselves are described in detail in *The Definitive Guide to The X Window System, Volume Two, Xlib Reference Manual* by O'Reilly and Associates. All functions in this library begin with the prefix Xrm.
- **Package XMITMisc** - This package provides access to the functions in the MIT Miscellaneous Extensions. The syntax for these functions is described in Chapter 6 of this manual. The functions themselves are described in detail in the documentation provided with X Window System Release 4 distribution from MIT. All functions in this library begin with the prefix XMITMisc.
- **Package XShape** - This package provides access to functions in the MIT Shape Extension. The syntax for these functions is described in Chapter 6 of this manual. The functions themselves are described in detail in *X11 Nonrectangular Window Shape Extension Version 1.0* by Keith Packard of the X Consortium. This document may be found in the X Window System Release 4 distribution from MIT. All functions in this library begin with the prefix XShape.
- **Package Xmbuf** - This package provides access to functions in the MIT Multi-Buffering Extension. The syntax for these functions is described in Chapter 6 of this manual. The functionality is described in detail in the *Extending X for Double-Buffering, Multi-Buffering and Stereo Version 3.2*, by Friedberg, Seiler and Vroom. All functions in this library begin with the prefix Xmbuf.
- **Package XShm** - This package provides access to functions in the MIT Shared Memory Extension. The syntax for these functions is described in Chapter 6 of this manual. The functions themselves are described in detail in *MIT Shared Memory Extension* by Jonathan Corbet. This document may be found in the X Window System Release 4 distribution from MIT. All functions in this library begin with the prefix XShm.
- **Package Xct** - This package provides access to Compound Text functions in the X Miscellaneous Utilities library. The syntax for these functions is described in Chapter 6 of this manual. The functions themselves are described in detail in X*mu Library*, a document in the X Window System Release 4 distribution from MIT. All functions in

this library begin with the prefix Xct.

- **Package Xt** - This library provides access to functions in the X Toolkit Intrinsics. The syntax for these functions is described in Chapter 7 of this manual. The functions themselves are described in detail in *The Definitive Guide to The X Window System, Volume Five, X Toolkit Intrinsics Reference Manual*, by O'Reilly and Associates. All functions in this library begin with the prefix Xt.
- **Package Xtdef** - This library defines the string constants in the X Toolkit. In the C language interface, these are contained in the header file <X11/StringDefs.h>.
- **Widget packages** - To create your own widgets, you will need to use the various widget packages. There is one package for each widget class. The package name is derived from the name of the widget class with the word Widget added at the end. For example, the Core widget class is contained in the package CoreWidget. The widget packages do not define any new routines, instead they define all of the structures and constants needed to subclass existing widgets.
- **Package Xmu** - This library provides access to functions in the X Miscellaneous Utilities library. The syntax for these functions is described in Chapter 8 of this manual. The functions themselves are described in detail in X*mu Library*, a document in the X Window System Release 4 distribution from MIT. All functions in this library begin with the prefix Xmu.
- **Package Xm** - This library provides access to functions in the Motif Library. The syntax for these functions is described in Chapter 9 of this manual. The functions themselves are described in detail in O*SF/Motif Programmer's Reference Manual*. All functions in this library begin with the prefix Xm.
- **Package Mrm** - This package provides access to the Motif Resource Manager, which complements the X Resource Manager. Mrm creates interface objects based on definitions in UID files. The syntax for these functions is described in Chapter 10 of this manual. The functions themselves are described in detail in O*SF/Motif Programmer's Reference Manual*. All functions in this library begin with the prefix Mrm.
- **Package Xmdef** - This library defines the string constants in the Motif Library. This includes all resource names and types used by Motif Widgets.
- **Motif Widget packages** - As with the X Toolkit Intrinsics widgets, if you want to create new widgets that are subclasses of existing Motif widgets, you will need to use the various widget packages that are defined. The package names for the Motif widgets are formed in the same way as the Intrinsic widgets, except that Xm is prefixed to the name. For example, the Primitive widget class is contained in the package XmPrimitiveWidget.

The following new packages were introduced in AXI 3.0:

- **Package XlibR5** - This package provides access to Xlib routines which are unique to X11R5. Segregating these calls into a different package allows the AXI user to link with either X11R4 or X11R5 libraries
- **Package Xcms** - This package provides the X11R5 device-independent color management routines and data types.
- **Package Xwc** - This package provides the X11R5 wide character support for international character sets.

- **Package XmuR5** - This package provides access to MIT's "miscellaneous utility" routines and data types which are unique to X11R5.
- **Package XtR5** - This package provides the X Intrinsics routines and data type which are unique to X11R5.

X11R5

X Windows Version 11 Release 5 is upwardly compatible with X11R4. The major new components are:

- **Font Service and Scalable Fonts** - Prior to the release of X11R5, X windows was limited to bitmapped fonts. These could not easily be resized or rotated, and each X server was required to maintain its own database of font definitions for each combination of typeface, style and point size. X11R5 now supports font scaling.
- **Device-Independent Color** - Prior to X11R5, all colors were expressed in terms of R,G,B triplets. X11R5 introduces the X Color Management System (XCMS) to support the precise description of device-independent colors.
- **Internationalization** - X11R5 provides support for both "Wide Characters" and "Multi-Byte" characters for internationalized text input and output.
- **Enhanced Resource Management** - X11R5 introduces several new functions to simplify X resource management..
- **New Xmu routines** - The MIT Miscellaneous Utilities package provides new string conversion routines for Xt and a new "Widget node" type for analyzing a widget class hierarchy.
- **PEX** - X11R5 introduces a standard protocol for 3-dimensional graphics, PEX. PEX stands for "PHIGS Extension to X". There are two APIs to PEX: PHIGS, a high-level 3D graphics library and PEXLib, a low-level library. Bindings to PEX are not provided by AXI.

STARS Bindings

The STARS Ada/Xlib Bindings are an Ada data structure interface to the MIT Xlib™ library of window manipulation primitives. The STARS bindings were originally produced by SAIC™ under a STARS Foundation contract. This implementation of the STARS Bindings connect to the AXI™ Ada Bindings, rather than interface directly to C. This release of the product only provides bindings to the Xlib functionality, future releases may support Xt as well. This binding is compatible with X11R4. Through the use of this interface, Ada software can take full advantage of the X library. In some cases, use of Ada's abstraction, naming and repackaging capabilities have been employed, however, the functionality of each subprogram corresponds exactly to the C Xlib layer.

Product Enhancements

This implementation of the STARS Ada/Xlib binding has been enhanced enormously to take care of some of the previous problems with the interface. Most of these enhancements are inherent in the implementation and will not affect any user's code other than improve performance. There were however few cases when the API had to be changed to make the implementation safe from the memory leaks that plagued previous versions. These API changes are minimal, but any existing code will not run "as is". A few changes will need to be made to existing code to make it compatible with this implementation. This inconvenience is regretted, but was considered essential to provide a better interface. All changes from the previous version are documented in the AXI 3.0 Release Notes.

Ada Tasking

See Appendix B of this manual for detailed information about AXI and tasking programs.

If a program is using tasking, a good way to handle incoming events is to spawn off all of the background tasks, then simply go into a loop that waits for events and dispatches them.

The following loop will block in the X_Next_Event() call, and will return when an event arrives.

```
while ( quit = False ) loop
        X_Lib.Events.X_Next_Event( dpy, event );
        HandleEvent( event );
end loop;
```

Note that other tasks in the application *may* continue to run even though the task (or "main" subprogram) that calls X_Next_Event() will block until the next event arrives. See Appendix B for more information about tasks and blocking behavior.

Packaging

The STARS Ada/Xlib Bindings are a collection of packages that provide an interface to the X Window System. All of the functions in the X Window System Library Interface are provided in package **X_Lib**. The functions are then further grouped together into packages depending on functionality. The capabilities of these packages can be imported into Ada applications by means of context clauses.

The STARS Ada/Xlib Bindings are composed of the following packages:

- **Package Key_Syms** -Provides definitions for all of the Keysyms in the X Window System. These constants are accessed as Xk_name, similar to the C interface.
- **Package X_Lib** - Encapsulates all of the functions in the X Window System Library Interface and is composed of the following packages:
- **Package X_Lib.Atoms** - This package provides access to all of the data types and functions in the X Window System related to setting and querying atoms. Also defined are the necessary constants corresponding to the predefined standard properties.
- **Package X_Lib.Fonts** - This package provides access to all the data types and functions related to the allocation, loading, freeing and use of fonts in the X Window System.
- **Package X_Lib.Colors** - This package provides access to all the data type definitions and functions related to the allocation, creation, storing, freeing, installation and use of colormaps in the X Window System.
- **Package X_Lib.Graphic_Output** - This package provides access to all the data type definitions and functions related to graphics primitives and the creation and setting of the Graphic Context resource. This package also defines the constants that should be used to set the different members of the GC.
- **Package X_Lib.Cursors** -This package provides access to all the data type definitions and functions related to the creation, setting and freeing of the Cursor resource. This package also defines the constants corresponding to the different predefined cursors.
- **Package X_Lib.Cut_And_Paste** - This package provides access to all the functions related to the Cut Buffer property.
- **X_Lib.Regions** - This package provides access to all the data type definitions and functions related to the creation, destruction, resizing and moving of X regions. Also included are functions that provide mathematical operations on regions.

- **X_Lib.Keyboard** - This package provides access to all the data type definitions and functions related to the keyboard mapping and creation of keyboard preferences. In addition, all keyboard related macros are included in this package implemented as functions. Also included are constants for the different masks provided.
- **X_Lib.Events** - This package provides access to all the data type definitions for events and functions for getting and handling them. Also included are the constants for the different Event Types that can be generated and the different Event masks supported.
- **Package X_Lib.Window_Manager** -This package provides access to all the data type definitions for the window manager functions like grabbing the pointer or keyboard, adding or removing hosts, enabling or disabling access control. Also provided are different related constants.
- **Package X_Lib.Resource_Manager** -This package provides access to the X Resource Manager data base.

2

Programming Considerations

The X Window System is coded in the C language. Fundamental differences between C and Ada preclude a syntactically equivalent Ada interface to X. The Ada X Interface is intended to be as close to the C language X interface as possible, recognizing that changes would be necessary. A set of guidelines was followed in making interface modifications, in order to make the overall interface consistent. The naming conventions and guidelines followed by the Ada X Interface are discussed in the following sections.

General Naming Conventions

All routines and structures have the same names as in the C Language Interface to X Windows. The major conventions are:

- Function and procedure names with compound words are constructed by capitalizing the first letter of each word. For example, a typical function name would be CreateWindow. In the C Interface all functions and many structures begin with an X; this has been omitted so that when a function is called in Ada, it will be called as Xlib.CreateWindow, and not Xlib.XCreateWindow.
- Structure names follow the same rules as function and procedure names. Once again, the names do not have an X at the beginning of the structure. For example, the C structure XGCValues, would be referenced as Xlib.GCValues in Ada, while the Display structure in C would be referenced as Xlib.Display in Ada.
- The names of all members of data structures are in lower case. Compound words, where needed, are constructed using underscores (_).
- C Macros have been replaced with functions and follow the same naming conventions as functions and procedures.
- The names of symbolic constants defined by X are mixed case, with the first letter of each word capitalized. Lower-case symbols are reserved for variables and user symbols are all upper-case, following existing conventions. The only exception is that predefined Atom names use all upper-case letters, with underscores separating words. Atom names begin with XA_ to distinguish them from user symbols.
- All data structures have associated with them an array type of the same name with List added at the end. For example, the Xlib.Rectangle structure has a corresponding array of Xlib.RectangleList. There are a few exceptions to this (mainly in the X Toolkit), where the structure ends in Rec, and in the corresponding array name, List replaces Rec.
- An access type for all structures is also defined. The name for the access type is formed by adding an _Ptr to the end of the structure name. For example, the Xlib.Rectangle structure would have an access type of Xlib.Rectangle_Ptr.

A general Ada naming convention states that all function and structure names should have underscores

separating each word in a compound word name. This convention has not been followed, to allow the interface to be as close as possible to the Standard X Interface.

Name Restrictions

Reserved words for Ada differ from those of C. Therefore, some of the procedure and structure names have been changed. The names chosen are usually shortened versions of the original names. For example, the element function in Xlib.GCValues was shortened to func.

Also, because Ada is not case sensitive, many of the parameter names had to be abbreviated. In many instances the C library would declare a type with mixed case, and declare a parameter with all lower case. In the Ada library, the name of the parameter is again shortened to make it distinct.

Procedure and Function Parameters

Most of the parameters to the functions and procedures are identical to those specified in the X Interface manuals. There are, however, several exceptions. In C, many of the routines require an address of a variable so that a value can be returned in that structure. Such parameters have been replaced in Ada with an out parameter. For example, in C the XNextEvent routine would be as follows:

```
void XNextEvent( event )
        XEvent *event; /* RETURN */
```

In Ada this function would appear as follows:

```
procedure NextEvent(
        report : out Event );
```

Many of the C routines that return an array will accept the address of a pointer as well as the address of an integer for the parameter. When the routine returns, the pointer contains the address of the first element of the array, and the integer contains the number of elements in the array. Status for the entire routine is then usually returned to indicate whether or not the call was successful. Such routines have been replaced in the Ada Interface by both a function and a procedure.

The function will return a pointer to an array, or NULL if the function failed, and the length of the array can be obtained from the Length attribute of the array. This single return value then takes the place of the two parameters and return value.

Such procedures will have parameters identical to the corresponding C function with an additional return status parameter. An example is shown below:

In C:

```
Status      XGetIconSizes( display, w, size_list, count )
Display     *display;
Window      w;
XIconSize   **size_list;
int         *count;
```

In Ada:

```
function    GetIconSizes(
dpy         : in Display;
w           : in Window ) return IconSizeList_Ptr;

procedure   GetIconSizes(
dpy         : in            Display;
w           : in            Window;
```

```
size_list       :       out     IconSizeList_Ptr;
stat            :       out     Status );
```

In the procedure, the out parameter Status could be eliminated, because a NULL is returned in size_list on failure. The information is redundant, but is kept in place for compatibility with the X Consortium Interface.

Specification File Naming Conventions

The naming of specifcation files generally follows the package names. There are a few expections with the widget packages. Use the following table as a road map to arrive at the filenames from package names:

Table 2-1 Xlib/Xt Packages

Package	Filename
BasicTypes	BasicTypes.a
Xlib	Xlib.a
XlibR5	XlibR5.a
Xt	Xt.a
XtR5	XtR5.a
XK	XK.a
Xrm	Xrm.a
XMITMisc	XMITMisc.a
XShape	XShape.a
Xmbuf	Xmbuf.a
XShm	XShm.a
Xcms	Xcms.a
Xwc	Xwc.a
Xtdef	Xtdef.a
Xmu	Xmu.a
XmuR5	XmuR5a
Xct	Xct.a
CoreWidget	Core.a
CompositeWidget	Composite.a
ConstraintWidget	Constraint.a
ShellWidgfet	Shell.a
WMShellWidget	WMShell.a

Table 2-1 Xlib/Xt Packages (Cont.)

Package	Filename
OverrideShellWidget	OverrideShell.a
VendorShellWidget	VendorShell.a
TransientShellWidget	TransientShell.a
TopLevelShellWidget	TopLevelShell.a
ApplicationShellWidget	ApplicationShell.a
XtObject	Object.a
RectObj	RectObj.a

Table 2-2 Motif Packages

Package	Filename
Xmdef	Xmdef.a
Xm	Xm.a
Mrm	Mrm.a
XmPrimitiveWidget	Primitive.a
XmManagerWidget	Manager.a
XmGadget	Gadget.a
XmExt	Ext.a
XmDesktop	Desktop.a
XmScreen	Screen.a
XmShell	Shell.a
XmVendorShell	Vendor.a
XmDisplay	Display.a
XmDialogShellWidget	DialogShell.a
XmArrowButtonWidget	Arrow.a
XmArrowButtonGadget	ArrowG.a
XmBulletinBoardWidget	Bulletin.a
XmLabelWidget	Label.a
XmLabelGadget	LabelG.a
XmCascadeButtonWidget	Cascade.a

Table 2-2 Motif Packages (Cont.)

Package	Filename
XmCascadeButtonGadget	CascadeG.a
XmSashWidget	Sash.a
XmScrollBarWidget	ScrollBar.a
XmDrawingAreaWiidget	DrawingArea.a
XmRwoColumnWidget	RowColumn.a
XmSelectionBoxWidget	SelectionBox.a
XmScrolledWindowWidget	ScrolledWin.a
XmScaleWidget	Scale.a
XmMainWindowWidget	MainWindow.a
XmDrawButtonWidget	DrawnButton.a
XmPanedWindowWidget	PanedWindow.a
XmFileSelectionBoxWidget	FileSelBox.a
XmCommandWidget	Command.a
XmFormWidget	Form.a
XmFrameWidget	Frame.a
XmListWidget	List.a
XmPushButtonWidget	PushButton.a
XmPushButtonGadget	PushButtonG.a
XmMenuShellWidget	MenuShell.a
XmMessageBoxWidget	MessageBox.a
XmSeparatorWidget	Separator.a
XmSeparatorGadget	SeparatorG.a
XmTextWidget	Text.a
XmToggleButtonWidget	Toggle.a
XmToggleButtonGadget	ToggleG.a

Table 2-3 STARS-Xlib Packages

Package	Filename
X_Lib	auto_stars.a
Atoms	auto_stars.a
Fonts	auto_stars.a
Colors	auto_stars.a
Graphic_Output	auto_stars.a
Cursors	auto_stars.a
Cut_And_Paste	auto_stars.a
Regions	auto_stars.a
Keyboard	auto_stars.a
Events	auto_stars.a
Window_Manager	auto_stars.a
Resource_Manager	auto_stars.a
Key_Syms	x_keysyms_.a

Boolean Values

Ada has a predefined data type Boolean with the possible values of True and False. The X Library defines its form of a boolean type called Bool with the same predefined possible values of True and False and the X Toolkit defines another type called Boolean. This causes some naming conflicts between the two languages. The Ada X Interface defines two types with slightly different names to provide compatibility with the C Libraries.

In the Xlib package, type Bool is defined along with the constants XTrue and XFalse. In the Xt package, type XtBoolean is defined along with the constants XtTrue and XtFalse. Therefore, an Ada program using this type would be as follows:

```
with Xlib;
with Xt;
...
declare
        x_boolean : Xlib.Bool;
        xt_boolean : Xt.XtBoolean;
begin
        x_boolean := Xlib.XTrue;
        xt_boolean := Xt.XtFalse;
end;
```

The routines in the libraries that return Boolean or Bool have two forms in the Ada X Interface. The first form returns the original type, be it Boolean or Bool. The second returns the Ada Boolean type. One such function declaration is shown below.

```
function DoesSaveUnders(
                s : in Screen_Ptr ) return Bool;

function DoesSaveUnders(
                s : in Screen_Ptr ) return Boolean;
```

The three distinct types are necessary because they have different sizes.

Bitwise Operations

X employs bitwise masking widely to specify flags and for many logical operations. AXI provides the "or" operation to do a bitwise-OR, "and" operation to do a bitwise-AND and an "xor" operation to do a bitwise-XOR.

Macros

Because Ada does not support text macros, all C macros have been replaced by functions or procedures. The functionality of these routines is identical to the C Library routines, but their names are slightly different. In C, all functions begin with a capital X, and macros have no signature prefix. In the Ada X Interface, because the macros are contained in the Xlib package, both macros and library routines will begin with an Xlib. Therefore, the DoesSaveUnders macros would be referenced as Xlib.DoesSaveUnders.

Strings

Ada has the concept of unbounded arrays of characters to allow varying length strings to be implemented. This technique gives the physical length of the memory allocated for the string, but does not give the programmer any indication as to the actual number of valid characters in the string. C handles this by requiring a NULL character to terminate all of its strings. Because the Ada X Interface is binding to existing C libraries, a ascii.nul character is appended internally wherever required. In addition, strings that are returned by AXI routines will be null terminated.

There are, however, a number of situations in code where it is not possible for AXI to automatically append a NULL character at end of a given string. Doing so would raise the possibility of memory leaks in code. In all such cases, AXI requires the programmer to take the responsibility of NULL terminating strings. Such calls are distinguised by the use of the NullString type to declare strings that require NULL termination.

Another problem with Ada strings is the issue of arrays of strings. Two dimensional arrays can be used to represent an array of strings, but this technique differs from the way in which C handles strings. C uses an array of character pointers for an array of strings. To provide compatibility with this method, a new type called StringList has been added.

A StringList is an array of String_Ptr's. AXI uses dynamic memory allocation to provide storage space for the array. There are routines that will build a structure with specified string lengths, and free the allocated storage when the StringList is no longer needed. There is also a routine that will return the command line arguments to a program in a string list. The command line can then be passed to several of the X Library routines that use and modify the command line argument list.

Another practice that should be avoided is the inclusion of backslash characters used in the C Language. In C, backslash characters are used as an *escape* mechanism to place control characters in a string. For example, the character sequence \n is converted to a Control-J character. This substitution is performed by the C Compiler. In Ada, if a \n or other backslash-prefixed character is placed in a string, the character sequence will actually appear in the text when displayed. These characters must be replaced by the appropriate characters from the ascii package. The \n would have to be replaced by the constant ascii.lf in your source code.

Dynamic Memory Allocation

Many of the X Library routines use dynamic memory allocation to return values to the calling routine. This technique has been maintained in the Ada X Interface. Consequently, programmers must be very careful to free all dynamic storage that is allocated by the library routines. The routines that require memory to be freed are noted in the X Library Manuals.

If a routine does allocate a structure, it will return an access type. There will then be a routine provided for freeing this memory. The routine name will be Free, followed by the name of the structure. For example, AXI defines a data type ColormapList_Ptr and a function FreeColormapList to free the corresponding memory.

Callbacks

The X Toolkit Interface makes extensive use of callback routines. These are routines that are passed into the Toolkit, and are called indirectly by the Toolkit in response to an asynchronous event, or some other action. Almost all widgets use callbacks to tie in the actions that your code must perform. These routines are passed into AXI by taking the address of the procedure. Because these routines are called by the Toolkit, they take a standard set of parameters. The O'Reilly X Toolkit Intrinsics Manual should be referenced for a complete description of the parameters each function should expect.

When using callbacks, the user should use certain precautions to ensure proper usage. The callback procedure should be in a separate package from the main program invoking the callback. The callback should never be a nested procedure local to the program invoking it. Also, if the compiler vendor supports it, the callback procedure should be marked as callable from C. This is usually done through an implementation-dependent pragma. See Appendix B of this manual for more information on the use of callbacks for this implementation.

A simple example of a callback routine is illustrated below:

```
package body Callbacks is

        procedure HelloWorldCB (
                w       : Widget;
                client  : Pointer;
                call    : Pointer ) is
        begin
        Text_IO.Put_Line( "Hello World" );
        end HelloWorldCB;

end Callbacks;
with Callbacks;
procedure HelloWorld is
        button : Widget;

 begin
        -- Widget creation code
        ...
        Xt.AddCallback  ( button, Xtdef.NCallback,
                          Callbacks.HelloWorldCB'Address,
                          Xt.XtNULL );
        ...
        Xt.AppMainLoop( app_context );
 end HelloWorld;
```

This example will add the callback HelloWorldCB to the button widget.

Routines that are defined as local procedures or functions SHOULD NOT be used as callbacks. Below is an example of a callback that is erroneous and will likely cause a PROGRAM_ERROR (segmentation violation) at run time.

```
procedure BadHelloWorld is
        button : Widget;

        procedure BadHelloWorldCB (
                w       : Widget;
                client  : Pointer;
                call    : Pointer ) is
        begin
        Text_IO.Put_Line( "Hello World" );
         end BadHelloWorldCB;

begin
         -- Widget creation code
        ...
        Xt.AddCallback( button, Xtdef.NCallback,
                        BadHelloWorldCB'Address,
                        Xt.XtNULL );
        ...
        Xt.AppMainLoop( app_context );

end BadHelloWorld;
```

This version will not work.

Tasking and AXI

See Appendix B of this manual for detailed information about AXI and tasking programs.

If a program is using tasking, a good way to handle incoming events is to spawn off all of the background tasks, then simply go into a loop that waits for events and dispatches them.

```
. . .
while ( quit = False ) loop
        Xlib.NextEvent( dpy, event );
        HandleEvent( event );
end loop;
```

or for the X Toolkit:

```
...
while ( quit = False ) loop
        Xt.AppNextEvent( app_context, event );
        Xt.DispatchEvent( event );
end loop;
```

These two loops will block in the NextEvent() call, and will return when an event arrives.

Note that other tasks in the application *may* continue to run even though the task (or "main" subprogram) that calls NextEvent() will block until the next event arrives. See Appendix B for more information about tasks and blocking behavior.

Using the Xlib Interface

The set of demonstration programs included with the AXI distribution provides an excellent starting point for learning how to write programs that use the Ada Bindings. There are many books available describing in detail the capabilities of the C language interface to Xlib, Xt and Motif. Because AXI closely resembles the C interface, users will not find many problems in calling the Ada subprograms.

Analysis of a Sample Program

In this section, we analyze a simple Ada Binding client program, illustrating a few of the capabilities of the binding. Portions of the code are provided. The source code is in the file VADS_location/axi/examples/xlibxt/hello_world.a.

```
--
--      This is a sample program that shows a simple usage of the Ada binding to
--      X Windows. This program shows the basic calls needed to construct
--      a client application based on the binding.
--
--      In specific, this program shows how to:
--
--      Open a display.
--      Get the default Screen.
--      Get the root window of that screen.
--      Create a Window.
--      Create a cursor for that window.
--      Set up input event masks.
--      Set up Icon names.
--      Load a font style.
--      Create a graphic context.
--      Map a window to the display.
--      Set the foreground and background colors for the window.
--      Read in events from the event queue.
--      Handle those events.
--      Draw strings of text to the window.

with XK;
with Text_Io;
with Xlib;
```

These context classes establish the dependence on the typical set of packages required in developing an X client program.

```
procedure Hello_World is
use     Xlib;
 --
 Buffer:String(1..1) := " ";
 Display_Id:Xlib.Display;
 Display_Not_Open:Exception;
 Done:Boolean := False;
 Drawable_Window:Xlib.Drawable;
 Gc_Value_Mask:Xlib.UnsignedLong := GcFont;
 Gc_Value_Rec:Xlib.GcValues;
 Hello_World_Bounds:Xlib.Rectangle;
 Hello_World_Border_Width:Xlib.Int := 2;
 Main_Attrs:Xlib.SetWindowAttributes;
 Main_Screen:Xlib.Int;
```

```
Main_Window:Xlib.Window;
Position_Point:Xlib.Point;
Pointer_String:constant String := "Hello World!" & Ascii.nul;
Root_Wind:Xlib.Window;
Status:Xlib.ComposeStatus;
Number:Xlib.Int := 0;
The_Symbol:Xlib.KeySym;
Title_Point:Xlib.Point;
Title_String:constant String := "Hello World" & Ascii.nul;
Wa_Values:Xlib.UnsignedLong :=
                    CWBackPixel or CWBorderPixel or CWCursor;
Window_Context:Xlib.GC;
X_Event:Xlib.Event;
```

These object declarations are typical of the standard variables required for establishing communication with Xlib. They are described in detail below where they are used:

```
--
--Beginning of main procedure
--
begin
--
--Opening the display, get the new Display_Id, get the default screen, set up the
--bounds of the panel, get the root window id and then create a window with our
--new defaults for the screen.
--
Text_Io.Put_Line ("Started the program");
Display_Id := Xlib.OpenDisplay ("");
--
-- Xlib.Synchronize (Display_Id);
--
Text_Io.Put_Line ("after open display");
if (Display_Id = Xlib.XNULL) then
      raise Display_Not_Open;
end if;
```

The first things that any client does is establish a connection to the server. The null string parameter specifies that the display to be opened is the one specified in the DISPLAY environment variable. The xstart program initializes this environment variable to screen 0 of the local workstation (the default screen).

The value returned is a pointer to the display, which includes many of the specifications of the server and its screens. If the pointer returned is null, then the call did not execute successfully.

```
Main_Screen := Xlib.DefaultScreen (Display_Id);
Root_Wind := Xlib.RootWindow (Display_Id, Main_Screen);
```

These calls return the values of the default screen and root (desk-top) window for use in later calls. The root window is the ancestor of every other window. Users can think of the root window as the "gray area" or background that encompasses the entire screen.

```
--
--      Here we are setting up the bounds for our window after we set up the bounds,
--      we calculate the starting position point from those bounds.
--
```

```
Hello_World_Bounds := (  5,   5, 810, 672);
Title_Point.x := Short (Hello_World_Bounds.Width / 2);
Title_Point.y := Short (Hello_World_Bounds.Height / 2);
```

These assignments setup the placement and size of the window to be created. They also set the placement of the title string for the window if such a capability exists under the window manager being used. The placement parameters correspond to an X value of 5, Y value of 5, width of 810 and height of 672. All of these values are in pixels.

```
--
--      Create the font cursor with the shape of a dot
--
Main_Attrs.Window_Cursor := Xlib.CreateFontCursor (Display_Id, Xlib.C_Dot);
Main_Attrs.Border_Pixel := Xlib.BlackPixel (Display_Id, Main_Screen);
Main_Attrs.Background_Pixel :=  Xlib.WhitePixel (Display_Id, Main_Screen);
Main_Attrs.Bit_Gravity := CenterGravity;
```

These assignments establish the cursor type and other window attributes that will be used for the window to be created. Here, the shape of the cursor will be a round dot, the window border will be black, the background color will be white, and the behavior upon resizing of a window is such that the existing bits will be placed in the center of the resized window.

```
--
--      With many of the parameters that we have just populated
--      we now create the window. The parameter Main_Window will
--      be the new window_id associated with this new window.
--
Main_Window :=
      Xlib.CreateWindow (Display_Id, Root_Wind, Xlib.Int (Hello_World_Bounds.x),
      Xlib.Int (Hello_World_Bounds.y), Xlib.Int (Hello_World_Bounds.Width),
      Xlib.Int (Hello_World_Bounds.Height), Hello_World_Border_Width,
      Xlib.DefaultDepth (Display_Id, Main_Screen),
      Xlib.InputOutput,
      Xlib.DefaultVisual (Display_Id, Main_Screen), Wa_Values,
      Main_Attrs);
```

This call creates a new window with the attributes established in previous code. The ID of the window is returned.

```
--
--      Set up the event mask for use with the event handler. Then call
--      Select_Input to make the mask visible to the server.
--
Main_Attrs.Event_Mask  := ExposureMask or ButtonPressMask or
                  ButtonMotionMask or
                  KeyPressMask;
Xlib.SelectInput (Display_Id, Main_Window, Main_Attrs.Event_Mask);
```

This code specifies the class of X Protocol events that will be sent to this window. For example, any key or mouse button that is pressed and any movement of the mouse will be sent to the window.

```
--
--      Set the icon name and the name that would be used as the title bar if we had one.
--
Xlib.SetIconName (Display_Id, Main_Window, "Hello_World");
```

This call establishes the text to be displayed when the window is iconified, for window managers that support this feature.

```
Xlib.StoreName (Display_Id, Main_Window, "Hello World");
```

This call establishes a name for the window when not iconified for use by the window manager (if any).

```
Xlib.DefineCursor (Display_Id, Main_Window, Main_Attrs.Window_Cursor);
```

This call binds the specified cursor assigned in previous code to the window.

```
Drawable_Window := Drawable (Main_Window);
--
--      Here we set the text font to a 9X15 size and create a Graphic_Context for the
--      specified drawable window.
--
Gc_Value_Rec.Font_Id := Xlib.LoadFont (Display_Id, "9x15" );
Window_Context := Xlib.CreateGc (Display_Id, Drawable_Window,
                        Gc_Value_Mask, Gc_Value_Rec);
```

The first assignment loads a font to be used later in the window. The second returns a default graphics context which can be used later when writing to the window. A graphics context specifies drawing attributes such as foreground and background color.

```
--
--      Now we are going to Map the Main_Window to the specified display
--
Xlib.MapWindow (Display_Id, Main_Window);
Xlib.Flush (Display_Id);
```

The first call brings up the window on the display. The second call causes all queued requests and events to be processed.

```
Xlib.SetBackground (Display_Id, Window_Context, 0);
Xlib.SetForeground (Display_Id, Window_Context, 1);
```

These calls select the pixel values of the foreground and background of the graphics context.

```
while not Done loop
--
--      Inside this loop is where we process events. The next call, NextEvent, returns the
--      next event that is in our event queue. We may then apply the event kind to a case
--      statement and perform the processing.
--
Xlib.NextEvent (Display_Id, X_Event);

case X_Event.event_type is
```

These statements define the main processing loop of the client. Events are taken from the event queue and processed according to their class.

```
   when Xlib.Expose  =>
        if (X_Event.Expose.Count = 0) then
--
--      here we will redraw the title string to the window each time we receive an expose
--      event, we want to draw the Title_String at the Title_Point.
```

```
--
            Xlib.DrawImageString (Display_Id, Drawable_Window, Window_Context,
               Xlib.Int (Title_Point.x), Xlib.Int (Title_Point.y), Title_String, Title_String'Length);
         end if;
```

This code will cause the title string of a window to be redrawn each time a window is exposed, perhaps after being made visible from another window that was in front of it.

It is worth noting that a window does not actually exist until an expose event for that window is received. This the reason why nearly all client programs wait on an expose event after mapping a window but before continuing with any other processing.

```
      when Xlib.ButtonPress =>
--
--       If a button press event is received by this client then
--       we want to display the pointer string at those coordinates.

         Text_Io.Put_Line("got a button press event");

         Position_Point.x := Xlib.Short (X_Event.xbutton.x);
         Position_Point.y := Xlib.Short (X_Event.xbutton.y);

         Xlib.DrawImageString
           (Display_Id, Drawable_Window, Window_Context,
            Xlib.Int (Position_Point.x), Xlib.Int (Position_Point.y),
            Pointer_String, Pointer_String'Length);
```

These statements cause the text stored in the object Pointer_String to be written to the location of the pointer device whenever a button is pressed.

```
      when Xlib.KeyPress =>
--
--       When a key press event is received by this client, we are
--       finished so we set the Done flag.

         Xlib.LookupString (X_Event, Buffer, The_Symbol, Status, Number);

         if Buffer (1) = 'q' or Buffer (1) = 'Q' then
         Text_Io.Put_Line ("Program has been terminated with """ &
         Buffer (1) & """ key");
         Done := True;
         else
                 Text_Io.Put_Line("Key pressed was """ & Buffer (1) & """");
         end if;

      when others =>
         null;
    end case;
  end loop;
```

This code handles all key presses. If the 'q' key is pressed (either shifted or not) then processing of the loop is set to complete, which will terminate the client's execution. For all other keys, the name of the key pressed is output.

```
  Xlib.CloseDisplay (Display_Id);
```

This line closes the connection to the server.

```
exception
   when Display_Not_Open  =>
        Text_Io.Put_Line ("Could not open Display");
end Hello_World;
```

This code is the exception handler executed when the request to open a connection in the inital part of this program is denied.

A Comparison with a C Client

The following program illustrates the semblance between a C client and an Ada client performing the same X client capabilities. We use the Hello World program described in the previous section as an example on which to base the C version below:

```
#include <stdio.h>
#include <X11/Xlib.h>
#include <X11/cursorfont.h>

main ()

  {
  char Buffer;
  Display*DisplayId;
  unsigned longGcValueMask = GCFont;
  XGCValuesGcValueRec;
  XSetWindowAttributesMainAttrs;
  int    MainScreen;
  WindowMainWindow;
  char *PointerString = "Hello World";
  WindowRootWind;
  KeySymTheSymbol;
  char *TitleString = "Hello World";
  unsigned longWaValues = CWBackPixel | CWBorderPixel | CWCursor;
  GC   WindowContext;
  XEventEvent, *ev; /* Note that name change due to type name*/

  printf ("Started the program\n");
  DisplayId = XOpenDisplay ("");
  printf ("after open display\n");
  if (! DisplayId)
     {
     printf ("Could not open Display\n");
     exit (1);
     }

  MainScreen = DefaultScreen (DisplayId);
  RootWind = RootWindow (DisplayId, MainScreen);
  MainAttrs.cursor = XCreateFontCursor (DisplayId, XC_dot);
  MainAttrs.border_pixel = BlackPixel (DisplayId, MainScreen);
  MainAttrs.background_pixel = WhitePixel (DisplayId, MainScreen);
  MainAttrs.bit_gravity = CenterGravity;
```

```
    MainWindow = XCreateWindow (DisplayId, RootWind, 5, 5, 810, 672, 2,
          DefaultDepth (DisplayId, MainScreen),
          InputOutput, DefaultVisual (DisplayId, MainScreen), WaValues, MainAttrs);
    XSelectInput (DisplayId, MainWindow,
          ExposureMask | ButtonPressMask | ButtonMotionMask | KeyPressMask);
    XSetIconName (DisplayId, MainWindow, TitleString);
    XStoreName (DisplayId, MainWindow, TitleString);
    XDefineCursor (DisplayId, MainWindow, MainAttrs.cursor);
    GcValueRec.font = XLoadFont (DisplayId, "9x15");
    WindowContext = XCreateGC (DisplayId, MainWindow, GcValueMask, GcValueRec);
    XMapWindow (DisplayId, MainWindow);
    XFlush (DisplayId);
    XSetBackground (DisplayId, WindowContext, 0);
    XSetForeground (DisplayId, WindowContext, 1);
    while (XNextEvent (DisplayId, &Event))
{
ev = &Event;
switch (ev->type)
    {
    case Expose:
          {
          XExposeEvent *e = (XExposeEvent *) ev;
          if (e->count == 0)
             {
             XDrawImageString (DisplayId, MainWindow, WindowContext, 810/2,
                      672/2, TitleString, strlen (TitleString));
             }
          break;
          }

    case ButtonPress:
          {
          XButtonEvent *e = (XButtonEvent *) ev;
          printf ("got a button press event\n");
          XDrawImageString (DisplayId, MainWindow, WindowContext, e->x, e->y,
                    PointerString, strlen (PointerString));
          break;
          }

    case KeyPress:
          {
          XLookupString (&Event, &Buffer, 1, &TheSymbol, NULL);
          if (Buffer)
             {
             if (Buffer == 'q' || Buffer == 'Q')
                   {
                   printf ("Program has been terminated with \"%c\" key\n", Buffer);
                   exit (0);
                   }
             else
                   {
                   printf ("Key pressed was \"%c\"\n", Buffer);
                   }
             }
```

```
            break;
            }
    }       /* end switch*/
}           /* end while */
}           /* end main*/
```

Compilation and Linking

Compilation and linking procedures are specific to the computer system and Ada compiler environment in use. See Appendix B of this manual for detailed information and sample instructions.

3

Convenience Functions

Depending on the Ada environment, several functions would be added to the Ada X Interface to simplify tasks that would commonly be performed by a programmer. This chapter contains descriptions for all of the added routines.

In this environment, there is one function, GetArguments in the Xlib package.

Xlib.GetArguments

Get Command Line Arguments

Calling Sequence

```
function GetArguments return StringList_Ptr;
```

Description

This function returns a StringList containing all of the command line arguments. This would be the same as argv in a C program. The list is required for several of the X Toolkit routines.

Memory for the list is dynamically allocated, and must be freed with Xlib.FreeStringList when it is no longer needed.

Return

A StringList_Ptr containing all command line arguments, or null on error.

See Also

Xlib.FreeStringList

4

Xlib Call Reference

Xlib routines

```
procedure ActivateScreenSaver(
     dpy  : in Display );

procedure AddHost(
     dpy  : in Display;
     host : in HostAddress );

procedure AddHosts(
     dpy       : in Display;
     hosts     : in HostAddressList;
     num_hosts : in Int );

procedure AddHosts(
     dpy   : in Display;
     hosts : in HostAddressList );

function AddPixel(
     img   : in Image_Ptr;
     value : in UnsignedLong ) return Int;

procedure AddToSaveSet(
     dpy  : in Display;
     w    : in Window );

function AllocClassHint return ClassHint_Ptr;

procedure AllocColor(
     dpy           : in     Display;
     cmap     : in     Colormap;
     colorcell_def : in out Color;
     return_status :    out Boolean );

procedure AllocColorCells(
     dpy           : in     Display;
     cmap          : in     Colormap;
     contig        : in     Boolean;
     plane_masks   : in out UnsignedLongList;
     nplanes       : in     UnsignedInt;
     pixels        : in out UnsignedLongList;
     ncolors       : in     UnsignedInt;
```

```
       return_status :    out Boolean );


procedure AllocColorCells(
       dpy           : in     Display;
       cmap          : in     Colormap;
       contig        : in     Boolean;
       plane_masks   : in out UnsignedLongList;
       pixels        : in out UnsignedLongList;
       return_status :    out Boolean );

  procedure AllocColorPlanes(
       dpy           : in     Display;
       cmap          : in     Colormap;
       contig        : in     Boolean;
       pixels        : in out UnsignedLongList;
       ncolors       : in     Int;
       nreds         : in     Int;
       ngreens       : in     Int;
       nblues        : in     Int;
       rmask         :    out UnsignedLong;
       gmask         :    out UnsignedLong;
       bmask         :    out UnsignedLong;
       return_status :    out Boolean );

  procedure AllocColorPlanes(
       dpy           : in     Display;
       cmap          : in     Colormap;
       contig        : in     Boolean;
       pixels        : in out UnsignedLongList;
       nreds         : in     Int;
       ngreens       : in     Int;
       nblues        : in     Int;
       rmask         :    out UnsignedLong;
       gmask         :    out UnsignedLong;
       bmask         :    out UnsignedLong;
       return_status :    out Boolean );

  function AllocIconSize return IconSize_Ptr;

  procedure AllocNamedColor(
       dpy           : in     Display;
       cmap          : in     Colormap;
       colorname     : in     String;
       colorcell_def :    out Color;
       rgb_db_def    :    out Color;
       return_status :    out Boolean );

  function AllocSizeHints return SizeHints_Ptr;

  function AllocStandardColormap return StandardColormap_Ptr;

  function AllocWMHints return WMHints_Ptr;
```

```
procedure AllowEvents(
     dpy        : in Display;
     event_mode : in Int;
     t          : in Time );

procedure AutoRepeatOff(
     dpy        : in Display );

procedure AutoRepeatOn(
     dpy        : in Display );

procedure Bell(
     dpy     : in Display;
     percent : in Int );

procedure ChangeActivePointerGrab(
     dpy        : in Display;
     event_mask : in UnsignedInt;
     curs       : in Cursor;
     t          : in Time );

procedure ChangeGC(
     dpy        : in Display;
     agc        : in GC;
     value_mask : in UnsignedLong;
     values     : in GCValues );

procedure ChangeKeyboardControl(
     dpy        : in Display;
     value_mask : in UnsignedLong;
     values     : in KeyboardControl );

procedure ChangeKeyboardMapping(
     dpy                : in Display;
     first_keycode      : in Int;
     keysyms_per_keycode : in Int;
     keysyms            : in KeySymList;
     num_keycodes       : in Int );

procedure ChangePointerControl(
     dpy               : in Display;
     do_accel          : in Boolean;
     do_threshold      : in Boolean;
     accel_numerator   : in Int;
     accel_denominator : in Int;
     threshold         : in Int );

procedure ChangeProperty(
     dpy       : in Display;
     w         : in Window;
     property  : in Atom;
     ptype     : in Atom;
     format    : in Int;
     mode      : in Int;
```

```
      data      : in Address;
      nelements : in Int  );

  procedure ChangeSaveSet(
      dpy         : in Display;
      w           : in Window;
      change_mode : in Int );

  procedure ChangeWindowAttributes(
      dpy        : in Display;
      w          : in Window;
      value_mask : in UnsignedLong;
      attributes : in SetWindowAttributes );

  procedure CheckIfEvent(
      dpy         : in     Display;
      report      :    out Event;
      predicate   : in     Address;
      arg         : in     String;
      event_found :    out Boolean );

  procedure CheckMaskEvent(
      dpy         : in     Display;
      event_mask  : in     Long;
      report      :    out Event;
      event_found :    out Boolean );

  procedure CheckTypedEvent(
      dpy         : in     Display;
      event_type  : in     Int;
      report      :    out Event;
      event_found :    out Boolean );

  procedure CheckTypedWindowEvent(
      dpy         : in     Display;
      w           : in     Window;
      event_type  : in     Int;
      report      :    out Event;
      event_found :    out Boolean );

  procedure CheckWindowEvent(
      dpy         : in     Display;
      w           : in     Window;
      event_mask  : in     Long;
      report      :    out Event;
      event_found :    out Boolean );

  procedure CirculateSubwindows(
      dpy        : in Display;
      w          : in Window;
      direction  : in Int );

  procedure CirculateSubwindowsDown(
      dpy        : in Display;
```

```
     w          : in Window );

procedure CirculateSubwindowsUp(
     dpy        : in Display;
     w          : in Window );

procedure ClearArea(
     dpy       : in Display;
     w         : in Window;
     x         : in Int;
     y         : in Int;
     width     : in Int;
     height    : in Int;
     exposures : in Boolean );

procedure ClearWindow(
     dpy : in Display;
     w   : in Window );

procedure ClipBox(
     r    : in     Region;
     rect : in out Rectangle );

procedure CloseDisplay(
     dpy : in out Display );

procedure ConfigureWindow(
     dpy        : in Display;
     w          : in Window;
     value_mask : in UnsignedInt;
     values     : in WindowChanges );

procedure ConvertSelection(
     dpy       : in Display;
     selection : in Atom;
     target    : in Atom;
     property  : in Atom;
     requestor : in Window;
     T         : in Time );

procedure CopyArea(
     dpy            : in Display;
     src            : in Drawable;
     dest           : in Drawable;
     agc            : in GC;
     src_x          : in Int;
     src_y          : in Int;
     width          : in UnsignedInt;
     height         : in UnsignedInt;
     dest_x         : in Int;
     dest_y         : in Int );

function CopyColormapAndFree(
     dpy  : in Display;
```

```
      cmap : in Colormap ) return Colormap;

  procedure CopyGC(
      dpy        : in Display;
      src        : in GC;
      value_mask : in UnsignedLong;
      dest       : in GC );

  procedure CopyPlane(
      dpy        : in Display;
      src        : in Drawable;
      dest       : in Drawable;
      agc        : in GC;
      src_x      : in Int;
      src_y      : in Int;
      width      : in UnsignedInt;
      height     : in UnsignedInt;
      dest_x     : in Int;
      dest_y     : in Int;
      plane      : in UnsignedLong );

  function CreateBitmapFromData(
      dpy    : in Display;
      draw   : in Drawable;
      data   : in UnsignedCharList;
      width  : in UnsignedInt;
      height : in UnsignedInt ) return Pixmap;

  function CreateColormap(
      dpy     : in Display;
      w       : in Window;
      xvisual : in Visual_Ptr;
      alloc   : in Int ) return Colormap;

  function CreateFontCursor(
      dpy   : in Display;
      shape : in UnsignedInt ) return Cursor;

  function CreateGC(
      dpy        : in Display;
      draw       : in Drawable;
      value_mask : in UnsignedLong;
      values     : in GCValues ) return GC;

  function CreateGlyphCursor(
      dpy              : in Display;
      source_font      : in Font;
      mask_font        : in Font;
      source_char      : in UnsignedInt;
      mask_char        : in UnsignedInt;
      foreground_color : in Color;
      background_color : in Color ) return Cursor;

  function CreateImage(
```

```
     dpy            : in Display;
     xvisual        : in Visual_Ptr;
     depth          : in UnsignedInt;
     format         : in Int;
     offset         : in Int;
     data           : in Address;
     width          : in UnsignedInt;
     height         : in UnsignedInt;
     bitmap_pad     : in Int;
     bytes_per_line : in Int ) return Image_Ptr;

function CreatePixmap(
     dpy      : in Display;
     draw     : in Drawable;
     width    : in UnsignedInt;
     height   : in UnsignedInt;
     depth    : in UnsignedInt ) return Pixmap;

function CreatePixmapCursor(
     dpy        : in Display;
     source     : in Pixmap;
     mask       : in Pixmap;
     foreground : in Color;
     background : in Color;
     x_hot      : in UnsignedInt;
     y_hot      : in UnsignedInt ) return Cursor;

function CreatePixmapFromBitmapData(
     dpy    : in Display;
     draw   : in Drawable;
     data   : in UnsignedCharList;
     width  : in UnsignedInt;
     height : in UnsignedInt;
     fg     : in UnsignedLong;
     bg     : in UnsignedLong;
     depth  : in UnsignedInt ) return Pixmap;

function CreateRegion return Region;

function CreateSimpleWindow(
     dpy          : in Display;
     parent       : in Window;
     x            : in Int;
     y            : in Int;
     width        : in UnsignedInt;
     height       : in UnsignedInt;
     border_width : in UnsignedInt;
     border       : in UnsignedLong;
     background   : in UnsignedLong ) return Window;

function CreateWindow(
     dpy          : in Display;
     parent       : in Window;
     x            : in Int;
```

```
      y            : in Int;
      width        : in UnsignedInt;
      height       : in UnsignedInt;
      border_width : in UnsignedInt;
      depth        : in Int;
      class        : in UnsignedInt;
      vis          : in Visual_Ptr;
      mask         : in UnsignedLong;
      attributes   : in SetWindowAttributes ) return Window;

  procedure DefineCursor(
      dpy  : in Display;
      w    : in Window;
      curs : in Cursor );

  function DeleteContext(
      dpy     : in Display;
      w       : in Window;
      contxt  : in Context ) return Int;

  function DeleteModifiermapEntry(
      modmap       : in ModifierKeymap;
      keysym_entry : in KeyCode;
      modifier     : in Int ) return ModifierKeymap;

  procedure DeleteProperty(
      dpy      : in Display;
      w        : in Window;
      property : in Atom );

  function DestroyImage(
      img : in Image_Ptr ) return Int;

  procedure DestroyRegion(
      r : in Region );

  procedure DestroySubwindows(
      dpy : in Display;
      w   : in Window );

  procedure DestroyWindow(
      dpy : in Display;
      w   : in Window );

  procedure DisableAccessControl(
      dpy : in Display );

  procedure DisplayKeycodes(
      dpy         : in     Display;
      min_keycode :    out Int;
      max_keycode :    out Int );

  function DisplayName return String_Ptr;
```

```
  function DisplayName(
       name : in String ) return String_Ptr;

  procedure DrawArc(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       x        : in Int;
       y        : in Int;
       width    : in UnsignedInt;
       height   : in UnsignedInt;
       angle1   : in Int;
       angle2   : in Int );

procedure DrawArcs(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       arcs     : in ArcList;
       narcs    : in Int );

  procedure DrawArcs(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       arcs     : in ArcList );

  procedure DrawImageString(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       x        : in Int;
       y        : in Int;
       text     : in String;
       length   : in Int );

  procedure DrawImageString(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       x        : in Int;
       y        : in Int;
       text     : in String );

  procedure DrawImageString16(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       x        : in Int;
       y        : in Int;
       text     : in Char2bList;
       length   : in Int );

  procedure DrawImageString16(
```

```
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      text     : in Char2bList );

  procedure DrawLine(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x1       : in Int;
      y1       : in Int;
      x2       : in Int;
      y2       : in Int );

  procedure DrawLines(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      points   : in PointList;
      npoints  : in Int;
      mode     : in Int );

 procedure DrawLines(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      points   : in PointList;
      mode     : in Int );

  procedure DrawPoint(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int );

  procedure DrawPoints(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      points   : in PointList;
      npoints  : in Int;
      mode     : in Int );

  procedure DrawPoints(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      points   : in PointList;
      mode     : in Int );

  procedure DrawRectangle(
```

```
        dpy      : in Display;
        draw     : in Drawable;
        agc      : in GC;
        x        : in Int;
        y        : in Int;
        width    : in UnsignedInt;
        height   : in UnsignedInt );

    procedure DrawRectangles(
        dpy         : in Display;
        draw        : in Drawable;
        agc         : in GC;
        rectangles  : in RectangleList;
        nrectangles : in Int );

    procedure DrawRectangles(
        dpy         : in Display;
        draw        : in Drawable;
        agc         : in GC;
        rectangles  : in RectangleList );

    procedure DrawSegments(
        dpy       : in Display;
        draw      : in Drawable;
        agc       : in GC;
        segments  : in SegmentList;
        nsegments : in Int );

    procedure DrawSegments(
        dpy       : in Display;
        draw      : in Drawable;
        agc       : in GC;
        segments  : in SegmentList );

procedure DrawString(
        dpy      : in Display;
        draw     : in Drawable;
        agc      : in GC;
        x        : in Int;
        y        : in Int;
        text     : in String;
        length   : in Int );

    procedure DrawString(
        dpy      : in Display;
        draw     : in Drawable;
        agc      : in GC;
        x        : in Int;
        y        : in Int;
        text     : in String );

    procedure DrawString16(
        dpy      : in Display;
        draw     : in Drawable;
```

```
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      text     : in Char2bList;
      length   : in Int );

  procedure DrawString16(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      text     : in Char2bList );

  procedure DrawText(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      items    : in TextItemList;
      nitems   : in Int );

  procedure DrawText(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      items    : in TextItemList );

  procedure DrawText16(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      items    : in TextItem16List;
      nitems   : in Int );

  procedure DrawText16(
      dpy      : in Display;
      draw     : in Drawable;
      agc      : in GC;
      x        : in Int;
      y        : in Int;
      items    : in TextItem16List );

  function EmptyRegion(
      r : in Region ) return Boolean;

  procedure EnableAccessControl(
      dpy : in Display );
```

```
function EqualRegion(
     r1 : in Region;
     r2 : in Region ) return Boolean;

function EventsQueued(
     dpy  : in Display;
     mode : in Int ) return Int;

procedure FetchBuffer(
     dpy    : in     Display;
     data   :    out String;
     nbytes :    out Int;
     buffer : in     Int );

function FetchBuffer(
     dpy    : in     Display;
     buffer : in     Int ) return String_Ptr;

procedure FetchBytes(
     dpy    : in     Display;
     data   :    out String;
     nbytes :    out Int );

function FetchBytes(
     dpy  : in Display ) return String_Ptr;

procedure FetchName(
     dpy         : in     Display;
     w           : in     Window;
     window_name :    out String;
     stat        :    out Status );

function FetchName(
     dpy         : in Display;
     w           : in Window ) return String_Ptr;

procedure FillArc(
     dpy      : in Display;
     draw     : in Drawable;
     agc      : in GC;
     x        : in Int;
     y        : in Int;
     width    : in UnsignedInt;
     height   : in UnsignedInt;
     angle1   : in Int;
     angle2   : in Int );

procedure FillArcs(
     dpy      : in Display;
     draw     : in Drawable;
     agc      : in GC;
     arcs     : in ArcList;
     narcs    : in Int );
```

```
procedure FillArcs(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       arcs     : in ArcList );

  procedure FillPolygon(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       points   : in PointList;
       npoints  : in Int;
       shape    : in Int;
       mode     : in Int );

  procedure FillPolygon(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       points   : in PointList;
       shape    : in Int;
       mode     : in Int );

  procedure FillRectangle(
       dpy      : in Display;
       draw     : in Drawable;
       agc      : in GC;
       x        : in Int;
       y        : in Int;
       width    : in UnsignedInt;
       height   : in UnsignedInt );

  procedure FillRectangles(
       dpy         : in Display;
       draw        : in Drawable;
       agc         : in GC;
       rectangles  : in RectangleList;
       nrectangles : in Int );

  procedure FillRectangles(
       dpy        : in Display;
       draw       : in Drawable;
       agc        : in GC;
       rectangles : in RectangleList );

  procedure FindContext(
       dpy   : in     Display;
       w     : in     Window;
       cont  : in     Context;
       data  : in out Pointer;
       found :    out Int );

  procedure Flush(
       dpy : in Display );
```

```
procedure ForceScreenSaver(
     dpy  : in Display;
     mode : in Int );

procedure Free(
     data : in Pointer );

procedure FreeColormap(
     dpy  : in Display;
     cmap : in Colormap );
procedure FreeColors(
     dpy     : in Display;
     cmap    : in Colormap;
     pixels  : in UnsignedLongList;
     npixels : in Int;
     planes  : in UnsignedLong );

procedure FreeColors(
     dpy     : in Display;
     cmap    : in Colormap;
     pixels  : in UnsignedLongList;
     planes  : in UnsignedLong );

procedure FreeCursor(
     dpy : in Display;
     cur : in Cursor );

procedure FreeExtensionList(
     list : in out StringList_Ptr );

procedure FreeFont(
     dpy         : Display;
     font_struct : FontStruct_Ptr );

procedure FreeFontInfo(
     names : in out StringList_Ptr;
     info  : in out FontStructList_Ptr );

procedure FreeFontNames(
     list : in out StringList_Ptr );

procedure FreeFontPath(
     paths : in out StringList_Ptr );

procedure FreeGC(
     dpy  : in Display;
     agc  : in GC );

procedure FreeModifiermap(
     modmap : in ModifierKeymap );

procedure FreePixmap(
     dpy  : Display;
```

```
       pmap : Pixmap );

  function GContextFromGC(
       agc : in GC ) return GContext;

  procedure Geometry(
       dpy           : in     Display;
       screen        : in     Int;
       user_geom     : in     String;
       default_geom  : in     String;
       bwidth        : in     UnsignedInt;
       fwidth        : in     UnsignedInt;
       fheight       : in     UnsignedInt;
       xadder        : in     Int;
       yadder        : in     Int;
       x             :    out Int;
       y             :    out Int;
       width         :    out Int;
       height        :    out Int;
       return_status :    out Status );

function GetAtomName(
       dpy  : in Display;
       atm  : in Atom ) return String_Ptr;

  procedure GetCommand(
       dpy           : in     Display;
       w             : in     Window;
       argv_return   : in out StringList;
       argc_return   :    out Int;
       return_status :    out Status );

  procedure GetClassHint(
       dpy           : in     Display;
       w             : in     Window;
       class_hints   :    out ClassHint;
       return_status :    out Status );

  procedure GetDefault(
       dpy     : in     Display;
       program : in     String;
       option  : in     String;
       value   :    out String;
       stat    :    out Status );

  procedure GetErrorDatabaseText(
       dpy            : in     Display;
       name           : in     String;
       message        : in     String;
       default_string : in     String;
       buffer         : in out String;
       length         : in     Int );

  procedure GetErrorText(
```

```
     dpy    : in Display;
     code   : in Int;
     buffer : in String;
     length : in Int );

function GetFontPath(
     dpy  : in Display ) return  StringList_Ptr;

procedure GetFontProperty(
     font_struct : in     FontStruct_Ptr;
     a      : in     Atom;
     value       :    out UnsignedLong;
     success     :    out Boolean );

procedure GetGCValues(
     dpy           : in     Display;
     agc           : in     GC;
     valuemask     : in     Int;
     values        :    out GCValues;
     return_status :    out Status );

procedure GetGeometry(
     dpy           : in     Display;
     draw          : in     Drawable;
     root          :    out Window;
     x             :    out Int;
     y             :    out Int;
     width         :    out UnsignedInt;
     height        :    out UnsignedInt;
     border_width  :    out UnsignedInt;
     depth         :    out UnsignedInt;
     return_status :    out Status );
procedure GetIconName(
     dpy       : in     Display;
     w         : in     Window;
     icon_name : in out String_Ptr;
     stat      :    out Status );

function GetIconName(
     dpy       : in Display;
     w         : in Window ) return String_Ptr;

procedure GetIconSizes(
     dpy       : in     Display;
     w         : in     Window;
     size_list :    out IconSIzeList_Ptr;
     stat      :    out Status );

function GetIconSizes(
     dpy  : in Display;
     w    : in Window ) return IconSizeList_Ptr;

function GetImage(
     dpy        : in Display;
```

```
      draw       : in Drawable;
      x          : in Int;
      y          : in Int;
      width      : in UnsignedInt;
      height     : in UnsignedInt;
      plane_mask : in UnsignedLong;
      format     : in Int ) return Image_Ptr;

  procedure GetInputFocus(
      dpy       : in     Display;
      focus     :    out Window;
      revert_to :    out Int);

  procedure GetKeyboardControl(
      dpy     : in     Display;
      values  :    out KeyboardState );

  procedure GetKeyboardMapping(
      dpy                 : in     Display;
      first_keycode       : in     Int;
      keycode_count       : in     Int;
      keysyms_per_keycode :    out Int;
      keysyms             :    out KeySymList_Ptr );

  function GetModifierMapping(
      dpy   : in Display ) return ModifierKeymap;

  procedure GetMotionEvents(
      dpy        : in     Display;
      w          : in     Window;
      start      : in     Time;
      stop       : in     Time;
      nevents    :    out Int;
      time_coord :    out TimeCoordList_Ptr;
      success    :    out Boolean );

  function GetMotionEvents(
      dpy    : in Display;
      w      : in Window;
      start  : in Time;
      stop   : in Time ) return TimeCoordList_Ptr;


procedure GetNormalHints(
      dpy           : in     Display;
      w             : in     Window;
      hints         :    out SizeHints;
      return_status :    out Status );

  function GetPixel(
      img  : in Image_Ptr;
      x    : in Int;
      y    : in Int ) return UnsignedLong;
```

```
procedure GetPointerControl(
     dpy               : in     Display;
     accel_numerator   :    out Int;
     accel_denominator :    out Int;
     threshold         :    out Int );

procedure GetPointerMapping(
     dpy       : in     Display;
     map       :    out UnsignedCharList;
     nmap      : in     Int;
     nelements :    out Int );

procedure GetPointerMapping(
     dpy       : in     Display;
     map       :    out UnsignedCharList;
     nelements :    out Int );

procedure GetRGBColormaps(
     dpy           : in     Display;
     w             : in     Window;
     cmaps         : in out StandardColormapList_Ptr;
     property      : in     Atom;
     return_status : in out Status );

function GetRGBColormaps(
     dpy      : in Display;
     w        : in Window;
     property : in Atom ) return StandardColormapList_Ptr;

procedure GetScreenSaver(
     dpy              : in     Display;
     timeout          :    out Int;
     interval         :    out Int;
     prefer_blanking  :    out Int;
     allow_exposures  :    out Int );

function GetSelectionOwner(
     dpy  : in Display;
     atm  : in Atom ) return Window;

procedure GetSizeHints(
     dpy           : in     Display;
     w             : in     Window;
     hints         :    out SizeHints;
     property      : in     Atom;
     return_status :    out Status );

procedure GetStandardColormap(
     dpy           : in     Display;
     w             : in     Window;
     cmap_info     :    out StandardColormap;
     property      : in     Atom;
     return_status :    out Status );
```

```
  function GetSubImage(
       dpy          : in Display;
       draw         : in Drawable;
       x            : in Int;
       y            : in Int;
       width        : in UnsignedInt;
       height       : in UnsignedInt;
       plane_mask   : in UnsignedLong;
       format       : in Int;
       dest_image   : in Image_Ptr;
       dest_x       : in Int;
       dest_y       : in Int ) return Image_Ptr;

  procedure GetTextProperty(
       dpy           : in     Display;
       w             : in     Window;
       text_prop     :    out TextProperty;
       property      : in     Atom;
       return_status :    out Status );

  procedure GetTransientForHint(
       dpy           : in     Display;
       w             : in     Window;
       prop_window   :    out Window;
       return_status :    out Status );

  function GetVisualInfo(
       dpy            : in Display;
       vinfo_mask     : in Long;
       vinfo_template : in VisualInfo ) return VisualInfoList_Ptr;

  procedure GetWindowAttributes(
       dpy               : in     Display;
       w                 : in     Window;
       window_attributes :    out WindowAttributes;
       return_status     :    out Status );

  procedure GetWindowProperty(
       dpy           : in     Display;
       w             : in     Window;
       property      : in     Atom;
       long_offset   : in     Long;
       long_length   : in     Long;
       delete        : in     Boolean;
       req_type      : in     Atom;
       actual_type   :    out Atom;
       actual_format :    out Int;
       nitems        :    out Int;
       bytes_after   :    out UnsignedLong;
       prop          :    out String_Ptr;
       success       :    out Boolean );

  procedure GetWMClientMachine(
       dpy           : in     Display;
```

```
      w             : in     Window;
      text_prop     :    out TextProperty;
      return_status :    out Status );

  procedure GetWMColormapWindows(
      dpy              : in     Display;
      w                : in     Window;
      colormap_windows : in out WindowList_Ptr;
      return_status    :    out Status );


function GetWMColormapWindows(
      dpy       : in     Display;
      w         : in     Window ) return WindowList_Ptr;

  procedure GetWMIconName(
      dpy           : in     Display;
      w             : in     Window;
      text_prop     :    out TextProperty;
      return_status :    out Status );

  procedure GetWMName(
      dpy           : in     Display;
      w             : in     Window;
      text_prop     :    out TextProperty;
      return_status :    out Status );

  procedure GetWMNormalHints(
      dpy           : in     Display;
      w             : in     Window;
      hints         :    out SizeHints;
      supplied      :    out Long;
      return_status :    out Status );

  function GetWMHints(
      dpy : in Display;
      w   : in Window ) return WMHints_ptr;

  function GetWMProtocols(
      dpy           : in     Display;
      w             : in     Window ) return AtomList_Ptr;

  procedure GetWMSizeHints(
      dpy           : in     Display;
      w             : in     Window;
      hints         :    out SizeHints;
      supplied      :    out Long;
      property      : in     Atom;
      return_status :    out Status );

  procedure GetZoomHints(
      dpy           : in     Display;
      w             : in     Window;
      zhints        :    out SizeHints;
```

```
      return_status :    out Status );

  function GrabButton(
      dpy           : in Display;
      button        : in UnsignedInt;
      modifiers     : in UnsignedInt;
      grab_window   : in Window;
      owner_events  : in Boolean;
      event_mask    : in UnsignedInt;
      pointer_mode  : in Int;
      keyboard_mode : in Int;
      confine_to    : in Window;
      curs          : in Cursor )  return Int;

  procedure GrabKey(
      dpy           : in Display;
      keycode       : in Int;
      modifiers     : in Int;
      grab_window   : in Window;
      owner_events  : in Boolean;
      pointer_mode  : in Int;
      keyboard_mode : in Int );
 function GrabKeyboard(
      dpy           : in Display;
      grab_owner    : in Window;
      owner_events  : in Boolean;
      pointer_mode  : in Int;
      keyboard_mode : in Int;
      t             : in Time ) return Int;

  function GrabPointer(
      dpy           : in Display;
      grab_window   : in Window;
      owner_events  : in Boolean;
      event_mask    : in UnsignedInt;
      pointer_mode  : in Int;
      keyboard_mode : in Int;
      confine_to    : in Window;
      curs          : in Cursor;
      t             : in Time ) return Int;

  procedure GrabServer(
      dpy  : Display );

  function IconifyWindow(
      dpy           : in Display;
      w             : in Window;
      screen_number : in Int ) return Status;

  procedure IfEvent(
      dpy         : in     Display;
      report      :    out Event;
      predicate   : in     Address;
      arg         : in     String );
```

```
function InsertModifiermapEntry(
     modmap       : in ModifierKeymap;
     keysym_entry : in KeyCode;
     modifier     : in Int ) return ModifierKeymap;

procedure InstallColormap(
     dpy  : in Display;
     cmap : in Colormap );

function InternAtom(
     dpy            : in Display;
     property_name  : in String;
     only_if_exists : in Boolean := True ) return Atom;

procedure IntersectRegion(
     sra : in Region;
     srb : in Region;
     dr  : in Region );

function KeycodeToKeysym(
     dpy       : in Display;
     key_code  : in KeyCode;
     index     : in Int ) return KeySym;

function KeysymToKeycode(
     dpy      : in Display;
     key_sym  : in KeySym ) return KeyCode;

procedure KeysymToString(
     key_sym  : in      KeySym;
     str      :    out String );

procedure KillClient(
     dpy      : in Display;
     resource : in XID );

function ListDepths(
     dpy           : in Display;
     screen_number : in Int ) return IntList_Ptr;

function ListExtensions(
     dpy : in Display ) return StringList_Ptr;

function ListFonts(
     dpy          : in Display;
     pattern      : in String;
     maxnames     : in Int ) return StringList_Ptr;

procedure ListFontsWithInfo(
     dpy       : in     Display;
     pattern   : in     String;
     maxnames  : in     Int;
     names     :    out StringList_Ptr;
```

```
      info      :    out FontStructList_Ptr );

  procedure ListHosts(
      dpy     : in     Display;
      hosts   :    out HostAddressList_Ptr;
      state   :    out Boolean );

  function ListInstalledColormaps(
      dpy   : in     Display;
      w     : in     Window ) return ColormapList_Ptr;

  function ListPixmapFormats(
      dpy    : in Display ) return PixmapFormatValuesList_Ptr;

  function ListProperties(
      dpy  : in Display;
      w    : in Window ) return AtomList_Ptr;

  function LoadFont(
      dpy  : in Display;
      name : in String  ) return Font;

  function LoadQueryFont(
      dpy  : in Display;
      name : in String ) return FontStruct_Ptr;

  procedure LookUpColor(
      dpy           : in     Display;
      cmap          : in     Colormap;
      colorname     : in     String;
      rgb_db_def    :    out Color;
      hardware_def  :    out Color;
      return_status :    out Status );

  function LookupKeysym(
      ev    : in Event;
      index : in Int ) return Keysym;

  procedure LookupString(
      ev        : in     Event;
      buffer    :    out String;
      num_bytes : in     Int;
      key_sym   :    out KeySym;
      stat      :    out ComposeStatus;  -- not implemented
      length    :    out Int );

  procedure LookupString(
      ev        : in     Event;
      buffer    :    out String;
      key_sym   :    out KeySym;
      stat      :    out ComposeStatus;  -- not implemented
      length    :    out Int );

  procedure LowerWindow(
```

```
     dpy : in Display;
     w   : in Window );

procedure MapRaised(
     dpy : in Display;
     w   : in Window );

procedure MapSubwindows(
     dpy : in Display;
     w   : in Window );

procedure MapWindow(
     dpy   : in Display;
     win   : in Window );

procedure MaskEvent(
     dpy        : in     Display;
     event_mask : in     Long;
     rep        :    out Event );

procedure MatchVisualInfo(
     dpy           : in     Display;
     screen        : in     Int;
     depth         : in     Int;
     class         : in     Int;
     vinfo         :    out VisualInfo;
     return_status :    out Status );

procedure MoveResizeWindow(
     dpy    : in Display;
     w      : in Window;
     x      : in Int;
     y      : in Int;
     width  : in UnsignedInt;
     height : in UnsignedInt );

procedure MoveWindow(
     dpy  : in Display;
     w    : in Window;
     x    : in Int;
     y    : in Int );

function NewModifiermap(
     max_keys_per_mod  : in Int ) return ModifierKeymap;

procedure NextEvent(
     dpy    : in     Display;
     report :    out Event );

procedure NoOp(
     dpy    : in Display );

procedure OffsetRegion(
     r   : in Region;
```

```
      dx  : in Int;
      dy  : in Int );

  function OpenDisplay(
      name  : in String ) return Display;

  function OpenDisplay return Display;

  procedure ParseColor(
      dpy           : in     Display;
      cmap          : in     Colormap;
      ch            : in     String;
      rgb_db_def    :    out Color;
      return_status :    out Status );

  procedure ParseGeometry(
      parsestring   : in     String;
      x             :    out Int;
      y             :    out Int;
      width         :    out Int;
      height        :    out Int;
      return_status :    out Status );

  procedure PeekEvent(
      dpy    : in     Display;
      report :    out Event );

  procedure PeekIfEvent(
      dpy        : in     Display;
      report     :    out Event;
      predicate  : in     Address;
      arg        : in     String );

  function Pending(
      dpy : Display ) return Int;

  function permalloc(
      size : in UnsignedInt ) return Address;

  function PointInRegion(
      r  : in Region;
      x  : in Int;
      y  : in Int ) return Int;

  function PolygonRegion(
      points    : in PointList;
      n         : in Int;
      fill_rule : in Int ) return Region;

  function PolygonRegion(
      points    : in PointList;
      fill_rule : in Int ) return Region;

  procedure PutBackEvent(
```

```
     dpy   : Display;
     ev    : Event );

procedure PutImage(
     dpy       : in Display;
     draw      : in Drawable;
     agc       : in GC;
     img       : in Image_Ptr;
     src_x     : in Int;
     src_y     : in Int;
     dest_x    : in Int;
     dest_y    : in Int;
     width : in UnsignedInt;
     height    : in UnsignedInt );

function PutPixel(
     img   : in Image_Ptr;
     x     : in Int;
     y     : in Int;
     pixel : in UnsignedLong ) return Int;

procedure QueryBestCursor(
     dpy           : in     Display;
     draw          : in     Drawable;
     width         : in     Int;
     height        : in     Int;
     rwidth        :    out Int;
     rheight       :    out Int;
     return_status :    out Status );

procedure QueryBestSize(
     dpy           : in     Display;
     class         : in     Int;
     draw          : in     Drawable;
     width         : in     UnsignedInt;
     height        : in     UnsignedInt;
     rwidth        :    out UnsignedInt;
     rheight       :    out UnsignedInt;
     return_status :    out Status );

procedure QueryBestStipple(
     dpy           : in     Display;
     draw          : in     Drawable;
     width         : in     Int;
     height        : in     Int;
     rwidth        :    out Int;
     rheight       :    out Int;
     return_status :    out Status );

procedure QueryBestTile(
     dpy           : in     Display;
     draw          : in     Drawable;
     width         : in     UnsignedInt;
     height        : in     UnsignedInt;
```

```
      rwidth        :    out UnsignedInt;
      rheight       :    out UnsignedInt;
      return_status :    out Status );

  procedure QueryColor(
      dpy           : in     Display;
      cmap          : in     Colormap;
      colorcell_def : in out Color );

  procedure QueryColors(
      dpy            : in     Display;
      cmap           : in     Colormap;
      colorcell_defs : in out ColorList;
      ncolors        : in     Int );

  procedure QueryColors(
      dpy            : in     Display;
      cmap           : in     Colormap;
      colorcell_defs : in out ColorList );

  procedure QueryExtension(
      dpy          : in     Display;
      name         : in     String;
      major_opcode :    out Int;
      first_event  :    out Int;
      first_error  :    out Int;
      present      :    out Boolean );

  function QueryFont(
      dpy     : in Display;
      font_id : in XID ) return FontStruct_Ptr;

  procedure QueryKeymap(
      dpy   : in     Display;
      keys  :    out String);

  procedure QueryPointer(
      dpy            : in     Display;
      w              : in     Window;
      root           :    out Window;
      child          :    out Window;
      root_x         :    out Int;
      root_y         :    out Int;
      win_x          :    out Int;
      win_y          :    out Int;
      keys_buttons   :    out UnsignedInt;
      success        :    out Boolean );

  procedure QueryTextExtents(
      dpy       : in     Display;
      font_id   : in     XID;
      text      : in     String;
      nchars    : in     Int;
      direction :    out Int;
```

```
     ascent    :    out Int;
     descent   :    out Int;
     overall   :    out CharStruct;
     status    :    out Int );

procedure QueryTextExtents(
     dpy       : in     Display;
     font_id   : in     XID;
     text      : in     String;
     direction :    out Int;
     ascent    :    out Int;
     descent   :    out Int;
     overall   :    out CharStruct;
     status    :    out Int );

procedure QueryTextExtents16(
     dpy       : in     Display;
     font_id   : in     XID;
     text      : in     Char2bList;
     nchars    : in     Int;
     direction :    out Int;
     ascent    :    out Int;
     descent   :    out Int;
     overall   :    out CharStruct;
     status    :    out Int );

procedure QueryTextExtents16(
     dpy       : in     Display;
     font_id   : in     XID;
     text      : in     Char2bList;
     direction :    out Int;
     ascent    :    out Int;
     descent   :    out Int;
     overall   :    out CharStruct;
     status    :    out Int );

procedure QueryTree(
     dpy       : in     Display;
     w         : in     Window;
     root      :    out Window;
     parent    :    out Window;
     children  :    out WindowList_Ptr;
     stat      :    out Status );

procedure RaiseWindow(
     dpy : in Display;
     w   : in Window );

procedure ReadBitmapFile(
     dpy           : in     Display;
     draw          : in     Drawable;
     filename      : in     String;
     width         :    out Int;
     height        :    out Int;
```

```
      bitmap        :    out Pixmap;
      x_hot         :    out Int;
      y_hot         :    out Int;
      return_status :    out Int );

  procedure RebindKeysym(
      dpy       : in Display;
      key_sym   : in KeySym;
      mod_list  : in KeySymList;
      mod_count : in Int;
      str       : in String;
      num_bytes : in Int );

  procedure RecolorCursor(
      dpy              : in Display;
      curs             : in Cursor;
      foreground_color : in Color;
      background_color : in Color );

  function ReconfigureWMWindow(
      dpy            : in Display;
      w              : in Window;
      screen_number  : in Int;
      value_mask     : in UnsignedInt;
      values         : in WindowChanges ) return Status;

  function RectInRegion(
      r      : in Region;
      x      : in Int;
      y      : in Int;
      width  : in UnsignedInt;
      height : in UnsignedInt ) return Int;

  procedure RefreshKeyboardMapping(
      ev  : in Event );

  procedure RemoveFromSaveSet(
      dpy    : in Display;
      w      : in Window );

  procedure RemoveHost(
      dpy    : in Display;
      host   : in HostAddress );


 procedure RemoveHosts(
      dpy       : in Display;
      hosts     : in HostAddressList;
      num_hosts : in Int );

  procedure RemoveHosts(
      dpy   : in Display;
      hosts : in HostAddressList );
```

```
procedure ReparentWindow(
     dpy    : in Display;
     w      : in Window;
     parent : in Window;
     x      : in Int;
     y      : in Int );

procedure ResetScreenSaver(
     dpy    : in Display );

procedure ResizeWindow(
     dpy    : in Display;
     w      : in Window;
     width  : in UnsignedInt;
     height : in UnsignedInt );

function ResourceManagerString(
     dpy  : in Display ) return String_Ptr;

procedure RestackWindows(
     dpy      : in Display;
     windows  : in WindowList;
     nwindows : in Int );

procedure RestackWindows(
     dpy      : in Display;
     windows  : in WindowList );

procedure RotateBuffers(
     dpy    : in Display;
     rotate : in Int );

procedure RotateWindowProperties(
     dpy        : in Display;
     w          : in Window;
     properties : in AtomList;
     num_prop   : in Int;
     npositions : in Int );

function SaveContext(
     dpy   : in Display;
     w     : in Window;
     cont  : in Context;
     data  : in Pointer ) return Int;

procedure SelectInput(
     dpy        : in Display;
     w          : in Window;
     event_mask : in UnsignedLong );

function SendEvent(
     dpy        : in Display;
     w          : in Window;
     propagate  : in Boolean;
```

```
      event_mask : in Long;
      ev         : in Event ) return Status;
  procedure SetAccessControl(
      dpy  : in Display;
      mode : in Int );

  function SetAfterFunction(
      dpy  : in Display;
      func : in Address ) return Address;

  procedure SetArcMode(
      dpy      : in Display;
      agc      : in GC;
      arc_mode : in Int );

  procedure SetBackground(
     dpy        : in Display;
     agc        : in GC;
     background : in UnsignedLong );

  procedure SetClassHint(
      dpy         : in Display;
      w           : in Window;
      class_hints : in ClassHint );

  procedure SetClipMask(
      dpy       : in Display;
      agc       : in GC;
      clip_mask : in Pixmap );

  procedure SetClipOrigin(
      dpy           : in Display;
      agc           : in GC;
      clip_x_origin : in Int;
      clip_y_origin : in Int );

  procedure SetClipRectangles(
      dpy           : in Display;
      agc           : in GC;
      clip_x_origin : in Int;
      clip_y_origin : in Int;
      rectangles    : in RectangleList;
      nrectangles   : in Int;
      ordering      : in Int );

  procedure SetClipRectangles(
      dpy           : in Display;
      agc           : in GC;
      clip_x_origin : in Int;
      clip_y_origin : in Int;
      rectangles    : in RectangleList;
      ordering      : in Int );

  procedure SetCloseDownMode(
```

```
     dpy        : in Display;
     close_mode : in Int );

procedure SetCommand(
     dpy  : in Display;
     w    : in Window;
     argv : in StringList );

procedure SetDashes(
     dpy         : in Display;
     agc         : in GC;
     dash_offset : in Int;
     dash_list   : in String;
     n           : in Int );

function SetErrorHandler(
     handler : in Address ) return Address;

procedure SetFillRule(
     dpy       : in Display;
     agc       : in GC;
     fill_rule : in Int );

procedure SetFillStyle(
     dpy        : in Display;
     agc        : in GC;
     fill_style : in Int );

procedure SetFont(
     dpy      : in Display;
     agc      : in GC;
     font_id  : in Font );

procedure SetFontPath(
     dpy         : in Display;
     directories : in StringList );

procedure SetForeground(
     dpy        : in Display;
     agc        : in GC;
     foreground : in UnsignedLong );

procedure SetFunction(
     dpy  : in Display;
     agc  : in GC;
     func : in Int );

procedure SetGraphicsExposures(
     dpy         : in Display;
     agc                 : in GC;
     graphics_exposures : in Boolean );

procedure SetIconName(
     dpy       : in Display;
```

```
      w         : in Window;
      icon_name : in String);

  procedure SetIconSizes(
      dpy       : in Display;
      w         : in Window;
      size_list : in IconSizeList;
      count     : in Int );

  procedure SetIconSizes(
      dpy       : in Display;
      w         : in Window;
      size_list : in IconSizeList );

  procedure SetInputFocus(
      dpy       : in Display;
      w         : in Window;
      revert_to : in Int;
      t         : in Time );

  function SetIOErrorHandler(
      handler : in Address ) return Address;


 procedure SetLineAttributes(
      dpy        : in Display;
      agc        : in GC;
      line_width : in UnsignedInt;
      line_style : in Int;
      cap_style  : in Int;
      join_style : in Int );

  function SetModifierMapping(
      dpy      : in Display;
      mod_map  : in ModifierKeymap ) return Int;

  procedure SetNormalHints(
      dpy   : in Display;
      w     : in Window;
      hints : in SizeHints );

  procedure SetPlaneMask(
      dpy        : in Display;
      agc        : in GC;
      plane_mask : in UnsignedLong );

  function SetPointerMapping(
      dpy   : in Display;
      map   : in UnsignedCharList;
      nmap  : in Int ) return Xlib.Int;

  function SetPointerMapping(
      dpy   : in Display;
      map   : in UnsignedCharList ) return Xlib.Int;
```

```
procedure SetRGBColormaps(
     dpy      : in Display;
     w        : in Window;
     cmaps    : in StandardColormapList;
     count    : in Int;
     property : in Atom );

procedure SetRGBColormaps(
     dpy      : in Display;
     w        : in Window;
     cmaps    : in StandardColormapList;
     property : in Atom );

procedure SetRegion(
     dpy : in Display;
     agc : in GC;
     r   : in Region );

procedure SetScreenSaver(
     dpy             : in Display;
     timeout         : in Int;
     interval        : in Int;
     prefer_blanking : in Int;
     allow_exposures : in Int );

procedure SetSelectionOwner(
     dpy       : in Display;
     selection : in Atom;
     owner     : in Window;
     t         : in Time );

procedure SetSizeHints(
     dpy      : in Display;
     w        : in Window;
     hints    : in SizeHints;
     property : in Atom );

procedure SetStandardColormap(
     dpy       : in Display;
     w         : in Window;
     cmap_info : in StandardColormap;
     property  : in Atom );

procedure SetStandardProperties(
     dpy         : in Display;
     win         : in Window;
     window_name : in String;
     icon_name   : in String;
     icon_pixmap : in Pixmap;
     argv   : in StringList;
     hints       : in SizeHints );

procedure SetState(
```

```
       dpy        : in Display;
       agc        : in GC;
       foreground : in UnsignedLong;
       background : in UnsignedLong;
       func       : in Int;
       plane_mask : in UnsignedLong );

  procedure SetStipple(
       dpy      : in Display;
       agc      : in GC;
       stipple  : in Pixmap );

  procedure SetSubwindowMode(
       dpy            : in Display;
       agc            : in GC;
       subwindow_mode : in Int );

  procedure SetTextProperty(
       dpy       : in Display;
       w         : in Window;
       text_prop : in TextProperty;
       property  : in Atom );

  procedure SetTile(
       dpy  : in Display;
       agc  : in GC;
       tile : in Pixmap );

  procedure SetTransientForHint(
       dpy         : in Display;
       w           : in Window;
       prop_window : in Window );

  procedure SetTSOrigin(
       dpy         : in Display;
       agc         : in GC;
       ts_x_origin : in Int;
       ts_y_origin : in Int );

  procedure SetWMClientMachine(
       dpy       : in Display;
       w         : in Window;
       text_prop : in TextProperty );



procedure SetWMColormapWindows(
       dpy              : in Display;
       w                : in Window;
       colormap_windows : in WindowList;
       count            : in Int );

  procedure SetWMColormapWindows(
       dpy              : in Display;
```

```
     w                : in Window;
     colormap_windows : in WindowList );

procedure SetWMHints(
     dpy      : in Display;
     w        : in Window;
     wm_hints : in WMHints );

procedure SetWMIconName(
     dpy       : in Display;
     w         : in Window;
     text_prop : in TextProperty );

procedure SetWMName(
     dpy       : in Display;
     w         : in Window;
     text_prop : in TextProperty );

procedure SetWMNormalHints(
     dpy      : in Display;
     w        : in Window;
     wm_hints : in SizeHints );

procedure SetWMProperties(
     dpy          : in Display;
     w            : in Window;
     window_name  : in TextProperty;
     icon_name    : in TextProperty;
     argv         : in StringList;
     normal_hints : in SizeHints;
     wm_hints     : in WMHints;
     class_hints  : in ClassHint );

function SetWMProtocols(
     dpy       : in Display;
     w         : in Window;
     protocols : in AtomList;
     count     : in Int ) return Status;

function SetWMProtocols(
     dpy       : in Display;
     w         : in Window;
     protocols : in AtomList ) return Status;

procedure SetWMSizeHints(
     dpy      : in Display;
     w        : in Window;
     hints    : in SizeHints;
     property : in Atom );

procedure SetWindowBackground(
     dpy              : in Display;
     w                : in Window;
     background_pixel : in UnsignedLong );
```

```
 procedure SetWindowBackgroundPixmap(
      dpy            : in Display;
      w               : in Window;
      background_tile : in Pixmap );

  procedure SetWindowBorder(
      dpy          : in Display;
      w            : in Window;
      border_pixel : in UnsignedLong );

  procedure SetWindowBorderPixmap(
      dpy         : in Display;
      w           : in Window;
      border_tile : in Pixmap );

  procedure SetWindowBorderWidth(
      dpy   : in Display;
      w     : in Window;
      width : in UnsignedInt );

  procedure SetWindowColormap(
      dpy  : in Display;
      w    : in Window;
      cmap : in Colormap );

  procedure SetZoomHints(
      dpy    : in Display;
      w      : in Window;
      zhints : in SizeHints );

  procedure ShrinkRegion(
      r   : in Region;
      dx  : in Int;
      dy  : in Int );

  procedure StoreBuffer(
      dpy    : in Display;
      bytes  : in String;
      nbytes : in Int;
      buffer : in Int := 0 );

  procedure StoreBuffer(
      dpy    : in Display;
      bytes  : in String;
      buffer : in Int := 0 );

  procedure StoreBytes(
      dpy    : in Display;
      bytes  : in String;
      nbytes : in Int );
```

```
procedure StoreBytes(
     dpy    : in Display;
     bytes  : in String );

procedure StoreColor(
     dpy       : in Display;
     cmap      : in Colormap;
     colorcell : in Color );

procedure StoreColors(
     dpy            : in Display;
     cmap           : in Colormap;
     colorcell_defs : in ColorList;
     ncolors        : in Int );

procedure StoreColors(
     dpy            : in Display;
     cmap           : in Colormap;
     colorcell_defs : in ColorList );

procedure StoreName(
     dpy  : in Display;
     win  : in Window;
     name : in String );

procedure StoreNamedColor(
     dpy       : in Display;
     cmap      : in Colormap;
     colorname : in String;
     pixel     : in UnsignedLong;
     flags     : in Int );

procedure StringListToTextProperty(
     list      : in     StringList;
     count     : in     Int;
     text_prop :    out TextProperty;
     stat      :    out Status );

procedure StringListToTextProperty(
     list      : in     StringList;
     text_prop :    out TextProperty;
     stat      :    out Status );

function StringToKeysym(
     str : in String ) return KeySym;

function SubImage(
     ximage          : in Image_Ptr;
     x               : in Int;
     y               : in Int;
     subimage_width  : in UnsignedInt;
     subimage_height : in UnsignedInt ) return Image_Ptr;

procedure SubtractRegion(
```

```
      sra : in Region;
      srb : in Region;
      dr  : in Region );

  procedure Sync(
      dpy     : in Display;
      discard : in Boolean );

  function Synchronize(
      dpy   : in Display;
      onoff : in Boolean ) return Address;

  procedure TextExtents(
      font_struct : in     FontStruct_Ptr;
      str         : in     String;
      nchars      : in     Int;
      direction   :    out Int;
      ascent      :    out Int;
      descent     :    out Int;
      overall     :    out CharStruct );

  procedure TextExtents(
      font_struct : in     FontStruct_Ptr;
      str         : in     String;
      direction   :    out Int;
      ascent      :    out Int;
      descent     :    out Int;
      overall     :    out CharStruct );

  procedure TextExtents16(
      font_struct : in     FontStruct_Ptr;
      str         : in     Char2bList;
      nchars      : in     Int;
      direction   :    out Int;
      ascent      :    out Int;
      descent     :    out Int;
      overall     :    out CharStruct );

  procedure TextExtents16(
      font_struct : in     FontStruct_Ptr;
      str         : in     Char2bList;
      direction   :    out Int;
      ascent      :    out Int;
      descent     :    out Int;
      overall     :    out CharStruct );

  procedure TextPropertyToStringList(
      text_prop  : in     TextProperty;
      list       :    out StringList_Ptr;
      stat       :    out Status );

  function TextWidth(
      font  : in FontStruct_Ptr;
      str   : in String;
```

```
     count : in Int ) return Int;

function TextWidth(
     font  : in FontStruct_Ptr;
     str   : in String ) return Int;

function TextWidth(
     font  : in FontStruct_Ptr;
     str   : in String;
     count : in Int ) return Short;

function TextWidth(
     font  : in FontStruct_Ptr;
     str   : in String ) return Short;

function TextWidth16(
     font  : in FontStruct_Ptr;
     str   : in Char2bList;
     count : in Int ) return Int;

function TextWidth16(
     font  : in FontStruct_Ptr;
     str   : in Char2bList ) return Int;

procedure TranslateCoordinates(
     dpy     : in     Display;
     src_w   : in     Window;
     frame_w : in     Window;
     src_x   : in     Int;
     src_y   : in     Int;
     new_x   :    out Int;
     new_y   :    out Int;
     child   :    out Window;
     success :    out Boolean );

procedure UndefineCursor(
     dpy : in Display;
     w   : in Window );

procedure UngrabButton(
     dpy       : in Display;
     button    : in UnsignedInt;
     modifiers : in UnsignedInt;
     w         : in Window );

procedure UngrabKey(
     dpy       : in Display;
     keycode   : in Int;
     modifiers : in UnsignedInt;
     w         : in Window );

procedure UngrabKeyboard(
     dpy  : in Display;
     t    : in Time );
```

```
  procedure UngrabPointer(
       dpy  : in Display;
       t    : in Time );

  procedure UngrabServer(
       dpy  : in Display );

  procedure UninstallColormap(
       dpy  : in Display;
       cmap : in Colormap );

  procedure UnionRectWithRegion(
       rect        : in Rectangle;
       src_region  : in Region;
       dest_region : in Region );

  procedure UnionRegion(
       sra  : in Region;
       srb  : in Region;
       dr   : in Region );

  function UniqueContext return Context;

  procedure UnloadFont(
       dpy     : in Display;
       font_id : in Font );

  procedure UnmapSubwindows(
       dpy  : in Display;
       w    : in Window );

  procedure UnmapWindow(
       dpy  : in Display;
       w    : in Window );

  function VisualIDFromVisual(
       vis : in Visual_Ptr ) return VisualId;

  procedure WMGeometry(
       dpy       : in     Display;
       screen    : in     Int;
       user_geom : in     String;
       def_geom  : in     String;
       bwidth    : in     UnsignedInt;
       hints     : in     SizeHints;
       x         :    out Int;
       y         :    out Int;
       width     :    out Int;
       height    :    out Int;
       gravity   :    out Int;
       success   :    out Int );

  procedure WarpPointer(
```

```
      dpy        : in Display;
      src_w      : in Window;
      dest_w     : in Window;
      src_x      : in Int;
      src_y      : in Int;
      src_width  : in UnsignedInt;
      src_height : in UnsignedInt;
      dest_x     : in Int;
      dest_y     : in Int );

  procedure WindowEvent(
      dpy        : in     Display;
      w          : in     Window;
      event_mask : in     Long;
      rep        :    out Event );

  function WithdrawWindow(
      dpy           : in Display;
      w             : in Window;
      screen_number : in Int ) return Status;

  function WriteBitmapFile(
      dpy        : in Display;
      filename   : in String;
      bitmap     : in Pixmap;
      width      : in UnsignedInt;
      height     : in UnsignedInt;
      x_hot      : in Int;
      y_hot      : in Int ) return Int;

  procedure XorRegion(
      sra : in Region;
      srb : in Region;
      dr  : in Region );
```

Screen and Display Macros

```
  function AllPlanes return Int;

  function BlackPixel(
      dpy : in Display;
      scr : in Int ) return UnsignedLong;

  function BlackPixelOfScreen(
      s : in Screen_Ptr ) return UnsignedLong;

  function CellsOfScreen(
      s : in Screen_Ptr ) return Int;

  function ConnectionNumber(
      dpy : in Display ) return Int;

  function DefaultColormap(
      dpy : in Display;
```

```
      scr : in Int ) return Colormap;


function DefaultColormapOfScreen(
      s : in Screen_Ptr ) return Colormap;

 function DefaultDepth(
      dpy : in Display;
      scr : in Int ) return Int;

 function DefaultDepthOfScreen(
      s : in Screen_Ptr ) return Int;

 function DefaultGC(
      dpy : Display;
      scr : Int ) return GC;

 function DefaultGCOfScreen(
      s : in Screen_Ptr ) return GC;

 function DefaultRootWindow(
      dpy : in Display ) return Window;

 function DefaultScreen(
      dpy : in Display ) return Int;

 function DefaultScreenOfDisplay(
      dpy : in Display ) return Screen_Ptr;

 function DefaultVisual(
      dpy : in Display;
      scr : in Int ) return Visual_Ptr;

 function DefaultVisualOfScreen(
      s : in Screen_Ptr ) return Visual_Ptr;

 function DisplayCells(
      dpy : in Display;
      scr : in Int ) return Int;

 function DisplayHeight(
      dpy : in Display;
      scr : in Int ) return Int;

 function DisplayHeightMM(
      dpy : in Display;
      scr : in Int ) return Int;

 function DisplayMotionBufferSize(
      dpy : in Display ) return Int;

 function DisplayOfScreen(
      s : Screen_Ptr ) return Display;
```

```
function DisplayPlanes(
     dpy : in Display;
     scr : in Int ) return Int;

function DisplayString(
     dpy : in Display ) return String_Ptr;

function DisplayWidth(
     dpy : in Display;
     scr : in Int ) return Int;

function DisplayWidthMM(
     dpy : in Display;
     scr : in Int ) return Int;
function DoesBackingStore(
     s : in Screen_Ptr ) return Int;

function DoesSaveUnders(
     s : in Screen_Ptr ) return Boolean;

function EventMaskOfScreen(
     s : in Screen_Ptr ) return Long;

function HeightOfScreen(
     s : in Screen_Ptr ) return UnsignedInt;

function HeightMMOfScreen(
     s : in Screen_Ptr ) return UnsignedInt;

function LastKnownRequestProcessed(
     dpy : in Display ) return UnsignedLong;

function MaxRequestSize(
     dpy : in Display ) return Int;

function MaxCmapsOfScreen(
     s : in Screen_Ptr ) return Int;

function MinCmapsOfScreen(
     s : in Screen_Ptr ) return Int;

function NextRequest(
     dpy : in Display ) return UnsignedLong;

function PlanesOfScreen(
     s : in Screen_Ptr ) return Int;

function ProtocolRevision(
     dpy : in Display ) return Int;

function ProtocolVersion(
     dpy : in Display ) return Int;

function QLength(
```

```
      dpy : in Display ) return Int;

  function RootWindow( dpy : Display;
               scr : Int ) return Window;

  function RootWindowOfScreen(
      s : in Screen_Ptr ) return window;

  function ScreenCount(
      dpy : in Display ) return Int;

  function ScreenNumberOfScreen(
      scr_ptr : in Screen_Ptr ) return Int;

  function ScreenOfDisplay(
      dpy : in Display;
      scr : in Int ) return Screen_Ptr;

  function ServerVendor(
      dpy : in Display ) return String_Ptr;

  procedure ServerVendor(
      dpy  : in     Display;
      str  :    out String;
      stat :    out Int );
  function VendorRelease(
      dpy : in Display ) return Int;

  function WhitePixel(
      dpy : in Display;
      scr : in Int ) return UnsignedLong;

  function WhitePixelOfScreen(
      s : in Screen_Ptr ) return UnsignedLong;

  function WidthOfScreen(
      s : in Screen_Ptr ) return Int;

  function WidthMMOfScreen(
      s : in Screen_Ptr ) return Int;
```

Image Format Macros

```
  function BitmapBitOrder(
      dpy : in Display ) return Int;

  function BitmapPad(
      dpy : in Display ) return Int;

  function BitmapUnit(
      dpy : in Display ) return Int;

  function ImageByteOrder(
      dpy : in Display ) return Int;
```

Keysym Classification Macros

```
function IsCursorKey(
     key_sym : in KeySym ) return Boolean;

function IsFunctionKey(
     key_sym : in KeySym ) return Boolean;

function IsKeypadKey(
     key_sym : in KeySym ) return Boolean;

function IsMiscFunctionKey(
     key_sym : in KeySym ) return Boolean;

function IsModifierKey(
     key_sym : in KeySym ) return Boolean;

function IsPFKey(
     key_sym : in KeySym ) return Boolean;
```

Ada Convenience Functions

```
procedure BuildClassHint(
     class_hint :    out ClassHint;
     res_name   : in     String;
     res_class  : in     String );

procedure BuildTextItem(
     item  : in out TextItem;
     text  : in     String );


function CreateStringList(
     nstrings         : in Int;
     chars_per_string : in Int ) return StringList_Ptr;

procedure c_exit(
     status : in Int );

procedure FreeAdaString(
     name : in out String_Ptr );

procedure FreeIntList(
     name : in out IntList_ptr );

procedure FreeStringList(
     list : in out StringList_Ptr );

function GetArguments return StringList_Ptr;
```

X11 Release 5 Functions

```
function CloseIM(
     input_method : in IM ) return Status;
```

```
  procedure CreateFontSet(
       dpy                          : in     Display;
       base_font_name_list          : in     String;
       missing_charset_list_return  :    out StringList_Ptr;
       missing_charset_count_return :    out Int;
       def_string_return            :    out String_Ptr;
       font_set                     :    out FontSet );

  function CreateIC(
        input_method : IM ) return IC;

  function DefaultString return String_Ptr;

  procedure DestroyIC(
       input_context : in IC );

  function DisplayOfIM(
       input_method : IM ) return Display;

  procedure ExtentsOfFontSet(
       font_set : in     FontSet;
       extents  :    out FontSetExtents );

  function FilterEvent(
       ev  : in Event;
       w   : in Window ) return Boolean;

  procedure FlushGC(
       dpy : in Display;
       agc : in GC );

  procedure FontsOfFontSet(
       font_set                : in     FontSet;
       font_struct_list_return :    out FontStructList_Ptr;
       font_name_list_return   :    out StringList_Ptr );

  procedure FreeFontSet(
       dpy      : in Display;
       font_set : in FontSet );

  function IMOfIC(
       input_context : in IC ) return IM;

  function LocaleOfFontSet(
       font_set  : in FontSet ) return String_Ptr;

  function LocaleOfIM(
       input_method  : in IM ) return String_Ptr;

  function OpenIM(
       dpy       : in Display;
       db        : in Pointer;  -- This should really be Xrm.Database;
       res_name  : in String;
       res_class : in String ) return IM;
```

```
function ScreenResourceString(
     scr  : in Screen_Ptr ) return String_Ptr;

procedure SetICFocus(
     input_context : in IC );

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in Int ) return Boolean;

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in Pointer ) return Boolean;

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in Window ) return Boolean;

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in String ) return Boolean;

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in Rectangle ) return Boolean;

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in Point ) return Boolean;

function SetICValue(
     input_context : in IC;
     name          : in String;
     value         : in Atom ) return Boolean;

function SetLocaleModifiers(
     modifiers_list : in String ) return String_Ptr;

function SetLocaleModifiers return String_Ptr;

function SupportsLocale return Boolean;

procedure UnSetICFocus(
     input_context : in IC );
```

5

X Resource Manager

X Resource Manager Routines

```
function ClassToString(
     clss : in Class ) return Xlib.String_Ptr;

procedure DestroyDatabase(
     db : in Database  );

function GetFileDatabase(
     filename : in String ) return DataBase;

procedure GetResource(
     db        : in     Database;
     str_name  : in     String;
     str_class : in     String;
     str_type  :    out Xlib.String_Ptr;
     val       :    out Value;
     found     :    out Boolean );

function GetStringDatabase(
     data : in String ) return DataBase;

procedure Initialize;

procedure MergeDatabases(
     source_db : in     Database;
     target_db : in out Database );

function NameToString(
     nme : in Name ) return Xlib.String_Ptr;

procedure ParseCommand(
     db          : in out Database;
     table       : in     OptionDescList;
     table_count : in     Xlib.Int;
     name        : in     String;
     argv        : in out Xlib.StringList_Ptr );

procedure PutFileDatabase(
     db        : in Database;
     stored_db : in String );
```

```
  procedure PutLineResource(
       db    : in out Database;
       line  : in     String );
  procedure PutResource(
       db            : in out Database;
       specifier     : in     String;
       resource_type : in     String;
       val           : in     Value );

  procedure PutStringResource(
       db       : in out Database;
       resource : in     String;
       value    : in     String );

  procedure QGetResource(
       db          : in     Database;
       quark_name  : in     NameList;
       quark_class : in     ClassList;
       quark_type  :    out Representation;
       val         :    out Value;
       found       :    out Boolean );

  procedure QGetSearchList(
       db          : in     Database;
       names       : in     NameList;
       classes     : in     ClassList;
       search_list : in out SearchList;
       list_length : in     Xlib.Int;
       success     :    out Boolean );

  procedure QGetSearchResource(
       search_list : in     SearchList;
       res_name    : in     Name;
       res_class   : in     Class;
       res_type    :    out Representation;
       res_value   :    out Value;
       found       :    out Boolean );

  procedure QPutResource(
       db            : in Database;
       bindings      : in BindingList;
       quarks        : in QuarkList;
       resource_type : in Representation;
       val           : in Value );

  procedure QPutStringResource(
       db            : in out Database;
       bindings      : in     BindingList;
       quarks        : in     QuarkList;
       val           : in     String );

  function QuarkToString(
       qurk      : in Quark ) return Xlib.String_Ptr;
```

```
function RepresentationToString(
     rep : in Representation ) return Xlib.String_Ptr;

procedure StringToBindingQuarkList(
     str      : in     String;
     bindings : in out BindingList;
     quarks   : in out QuarkList );

function StringToClass(
     str : in String ) return Quark;

function StringToName(
     str : in String ) return Quark;

function StringToQuark(
     str : in String ) return Quark;

function StringToRepresentation(
     str : in String ) return Quark;

procedure StringToQuarkList(
     str    : in     String;
     quarks :    out QuarkList );

function UniqueQuark return Quark;
```

6

X Color Management System

XCMS Routines

```
procedure AllocColor(
     dpy           : in     Xlib.Display;
     cmap          : in     Xlib.Colormap;
     color_in_out  : in out Color;
     result_format : in     ColorFormat;
     return_status :    out Boolean );

procedure AllocNamedColor(
     dpy                 : in     Xlib.Display;
     cmap                : in     Xlib.Colormap;
     color_string        : in     String;
     color_screen_return :    out Color;
     color_exact_return  :    out Color;
     result_format       : in     ColorFormat;
     return_status       :    out Boolean );

function CCCOfColormap(
     dpy  : in     Xlib.Display;
     cmap : in     Xlib.Colormap ) return CCC;

procedure ConvertColors(
     ccc_in        : in     CCC;
     colors_in_out : in out ColorList;
     ncolors       : in     Xlib.UnsignedInt;
     target_format : in     ColorFormat;
     compression_flags_return :     out BooleanList;
     return_status            :     out Boolean );

procedure ConvertColors(
     ccc_in        : in     CCC;
     colors_in_out : in out ColorList;
     target_format : in     ColorFormat;
     compression_flags_return :     out BooleanList;
     return_status            :     out Boolean );

function DefaultCCC(
     dpy           : in Xlib.Display;
     screen_number : in Xlib.Int ) return CCC;
```

```
  procedure LookupColor(
       dpy                : in     Xlib.Display;
       cmap               : in     Xlib.Colormap;
       color_string       : in     String;
       color_screen_return  :    out Color;
       color_exact_return  :    out Color;
       result_format      : in     ColorFormat;
       return_status      :    out Xlib.Status );

  procedure QueryBlack(
       ccc_in        : in     CCC;
       target_format : in     ColorFormat;
       color_return  :    out Color;
       return_status :    out Boolean );

  procedure QueryBlue(
       ccc_in        : in     CCC;
       target_format : in     ColorFormat;
       color_return  :    out Color;
       return_status :    out Boolean );

  procedure QueryColor(
       dpy           : in     Xlib.Display;
       cmap          : in     Xlib.Colormap;
       color_in_out  : in out Color;
       result_format : in     ColorFormat;
       return_status :     out Boolean );

  procedure QueryColors(
       dpy            : in     Xlib.Display;
       cmap           : in     Xlib.Colormap;
       colors_in_out  : in out ColorList;
       ncolors        : in     Xlib.UnsignedInt;
       result_format  : in     ColorFormat;
       return_status  :    out Boolean );

  procedure QueryColors(
       dpy            : in     Xlib.Display;
       cmap           : in     Xlib.Colormap;
       colors_in_out  : in out ColorList;
       result_format  : in     ColorFormat;
       return_status  :    out Boolean );

  procedure QueryGreen(
       ccc_in        : in     CCC;
       target_format : in     ColorFormat;
       color_return  :    out Color;
       return_status :    out Boolean );

  procedure QueryRed(
       ccc_in        : in     CCC;
       target_format : in     ColorFormat;
       color_return  :    out Color;
       return_status :    out Boolean );
```

```
procedure QueryWhite(
     ccc_in        : in     CCC;
     target_format : in     ColorFormat;
     color_return  :    out Color;
     return_status :    out Boolean );

procedure StoreColor(
     dpy    : in     Xlib.Display;
     cmap   : in     Xlib.Colormap;
     clr    : in out Color;
     return_status : out Xlib.Status );

procedure StoreColors(
     dpy     : in     Xlib.Display;
     cmap    : in     Xlib.Colormap;
     colors  : in out ColorList;
     ncolors : in     Xlib.Int;
     compression_flags_return :    out BooleanList;
     return_status : out Xlib.Status );

procedure StoreColors(
     dpy     : in     Xlib.Display;
     cmap    : in     Xlib.Colormap;
     colors  : in out ColorList;
     compression_flags_return :    out BooleanList;
     return_status : out Xlib.Status );

procedure TekHVCQueryMaxC(
     ccc_in       : in     CCC;
     hue          : in     Float;
     value        : in     Float;
     color_return :    out Color;
     return_status : out Boolean );

procedure TekHVCQueryMaxV(
     ccc_in        : in     CCC;
     hue           : in     Float;
     chroma        : in     Float;
     color_return  :    out Color;
     return_status : out Boolean );

procedure TekHVCQueryMaxVC(
     ccc_in       : in     CCC;
     hue          : in     Float;
     color_return :    out Color;
     return_status : out Boolean );

procedure TekHVCQueryMaxVSamples(
     ccc_in        : in     CCC;
     hue           : in     Float;
     colors_return :    out ColorList;
     nsamples      : in     Xlib.UnsignedInt;
     return_status :    out Boolean );
```

```
  procedure TekHVCQueryMaxVSamples(
       ccc_in        : in     CCC;
       hue           : in     Float;
       colors_return :    out ColorList;
       return_status :    out Boolean );

  procedure TekHVCQueryMinV(
       ccc_in        : in     CCC;
       hue           : in     Float;
       chroma        : in     Float;
       color_return  :    out Color;
       return_status :    out Boolean );
```

7

X Wide Character Strings

Xwc Routines

```
procedure DrawImageString(
     dpy         : in Xlib.Display;
     drawable    : in Xlib.Window;
     font_set    : in FontSet;
     gc          : in Xlib.GC;
     x           : in Xlib.Int;
     y           : in Xlib.Int;
     str         : in wchar_t_String;
     num_wchars  : in Xlib.Int );

procedure DrawImageString(
     dpy         : in Xlib.Display;
     drawable    : in Xlib.Window;
     font_set    : in FontSet;
     gc          : in Xlib.GC;
     x           : in Xlib.Int;
     y           : in Xlib.Int;
     str         : in wchar_t_String );

procedure DrawString(
     dpy        : in Xlib.Display;
     drawable   : in Xlib.Window;
     font_set   : in FontSet;
     gc         : in Xlib.GC;
     x          : in Xlib.Int;
     y          : in Xlib.Int;
     str        : in wchar_t_String;
     num_wchars : in Xlib.Int );

procedure DrawString(
     dpy        : in Xlib.Display;
     drawable   : in Xlib.Window;
     font_set   : in FontSet;
     gc         : in Xlib.GC;
     x          : in Xlib.Int;
     y          : in Xlib.Int;
     str        : in wchar_t_String );

procedure DrawText(
     dpy        : in Xlib.Display;
```

```
      drawable   : in Xlib.Window;
      gc         : in Xlib.GC;
      x          : in Xlib.Int;
      y          : in Xlib.Int;
      items      : in TextItemList;
      nitems     : in Xlib.Int );

  procedure DrawText(
      dpy        : in Xlib.Display;
      drawable   : in Xlib.Window;
      gc         : in Xlib.GC;
      x          : in Xlib.Int;
      y          : in Xlib.Int;
      items      : in TextItemList );

  procedure FreeStringSet(
      list   : in out wchar_t_StringList_Ptr );

  procedure LookupString(
      ic_in          : in     IC;
      ev             : in     Xlib.Event;
      buffer_return  :    out wchar_t_String;
      wchars_buffer  : in     Xlib.Int;
      keysym_return  :    out Xlib.KeySym;
      status_return  :    out Xlib.Status;
      length         :    out Xlib.Int );

  procedure LookupString(
      ic_in          : in     IC;
      ev             : in     Xlib.Event;
      buffer_return  :    out wchar_t_String;
      keysym_return  :    out Xlib.KeySym;
      status_return  :    out Xlib.Status;
      length         :    out Xlib.Int );

  function ResetIC(
      ic_in : in IC ) return Xlib.Pointer;

  function TextEscapement(
      font_set  : in FontSet;
      str       : in wchar_t_String;
      num_bytes : in Xlib.Int ) return Xlib.Int;

  function TextEscapement(
      font_set  : in FontSet;
      str       : in wchar_t_String ) return Xlib.Int;

  procedure TextExtents(
      font_set               : in     FontSet;
      str                    : in     wchar_t_String;
      num_wchars             : in     Xlib.Int;
      overall_ink_return     :    out Xlib.Int;
      overall_logical_return :    out Xlib.Int;
      width                  :    out Xlib.Int );
```

```
procedure TextExtents(
     font_set               : in     FontSet;
     str                    : in     wchar_t_String;
     overall_ink_return     :    out Xlib.Int;
     overall_logical_return :    out Xlib.Int;
     width                  :    out Xlib.Int );

procedure TextListToTextProperty(
     dpy              : in     Xlib.Display;
     list             : in     wchar_t_StringList;
     count            : in     Xlib.Int;
     style            : in     ICCEncodingStyle;
     text_prop_return :    out Xlib.TextProperty;
     return_status    :    out Xlib.Status );

procedure TextListToTextProperty(
     dpy              : in     Xlib.Display;
     list             : in     wchar_t_StringList;
     style            : in     ICCEncodingStyle;
     text_prop_return :    out Xlib.TextProperty;
     return_status    :    out Xlib.Status );

procedure TextPerCharExtents(
     font_set               : in     FontSet;
     str                    : in     wchar_t_String;
     num_wchars             : in     Xlib.Int;
     ink_array_return       :    out Xlib.RectangleList;
     logical_array_return   :    out Xlib.RectangleList;
     array_size             : in     Xlib.Int;
     num_chars_return       :    out Xlib.Int;
     overall_ink_return     :    out Xlib.Rectangle;
     overall_logical_return :    out Xlib.Rectangle;
     return_status          :    out Xlib.Status );

procedure TextPerCharExtents(
     font_set               : in     FontSet;
     str                    : in     wchar_t_String;
     ink_array_return       :    out Xlib.RectangleList;
     logical_array_return   :    out Xlib.RectangleList;
     array_size             : in     Xlib.Int;
     num_chars_return       :    out Xlib.Int;
     overall_ink_return     :    out Xlib.Rectangle;
     overall_logical_return :    out Xlib.Rectangle;
     return_status          :    out Xlib.Status );

procedure TextPropertyToTextList(
     dpy          : in     Xlib.Display;
     text_prop    : in     Xlib.TextProperty;
     list_return  :    out wchar_t_StringList_Ptr;
     count_return :    out Xlib.Int;
     return_status :    out Xlib.Status );
```

8

X Server Extensions

MIT Miscellaneous Extension

```
function GetBugMode(
     dpy : in Xlib.Display ) return Boolean;

procedure QueryExtension(
     dpy         : in     Xlib.Display;
     event_basep :    out Xlib.Int;
     error_basep :    out Xlib.Int;
     success     :    out Boolean );

function SetBugMode(
     dpy    : in Xlib.Display;
     onOff  : in Boolean ) return Xlib.Int;
```

Shape Extension

```
procedure CombineMask(
     dpy       : in Xlib.Display;
     dest      : in Xlib.Window;
     dest_kind : in Xlib.Int;
     x_off     : in Xlib.Int;
     y_off     : in Xlib.Int;
     src       : in Xlib.Int;
     op        : in Xlib.Int );

procedure CombineRegion(
     dpy       : in Xlib.Display;
     dest      : in Xlib.Window;
     dest_kind : in Xlib.Int;
     x_off     : in Xlib.Int;
     y_off     : in Xlib.Int;
     reg       : in Xlib.Pointer;
     op        : in Xlib.Int );

procedure CombineShape(
     dpy       : in Xlib.Display;
     dest      : in Xlib.Window;
     dest_kind : in Xlib.Int;
     x_off     : in Xlib.Int;
     y_off     : in Xlib.Int;
     src       : in Xlib.Int;
```

```
      src_kind  : in Xlib.Int;
      op        : in Xlib.Int );
  procedure GetRectangles(
      dpy      : in     Xlib.Display;
      window   : in     Xlib.Window;
      kind     : in     Xlib.Int;
      ordering :    out Xlib.Int;
      rects    : in out Xlib.RectangleList_Ptr );

  function InputSelected(
      dpy    : in Xlib.Display;
      window : in Xlib.Window ) return Xlib.UnsignedLong;

  procedure OffsetShape(
      dpy       : in Xlib.Display;
      dest      : in Xlib.Window;
      dest_kind : in Xlib.Int;
      x_off     : in Xlib.Int;
      y_off     : in Xlib.Int );

  procedure QueryExtension(
      dpy         : in     Xlib.Display;
      event_basep :    out Xlib.Int;
      error_basep :    out Xlib.Int;
      success     :    out Boolean );

  procedure QueryExtents(
      dpy             : in     Xlib.Display;
      window          : in     Xlib.Window;
      bounding_shaped :    out Boolean;
      x_bounding      :    out Xlib.Int;
      y_bounding      :    out Xlib.Int;
      w_bounding      :    out Xlib.Int;
      h_bounding      :    out Xlib.Int;
      clip_shaped     :    out Boolean;
      x_clip          :    out Xlib.Int;
      y_clip          :    out Xlib.Int;
      w_clip          :    out Xlib.Int;
      h_clip          :    out Xlib.Int;
      return_status   :    out Xlib.Int );

  procedure QueryVersion(
      dpy           : in     Xlib.Display;
      major_version :    out Xlib.Int;
      minor_version :    out Xlib.Int;
      return_status :    out Xlib.Int );

  procedure SelectInput(
      dpy    : in Xlib.Display;
      window : in Xlib.Window;
      mask   : in Xlib.UnsignedLong );
```

```
function Attach(
     dpy      : in Xlib.Display;
     shminfo  : in SegmentInfo ) return Xlib.Status;

function CreateImage(
     dpy      : in Xlib.Display;
     visual   : in Xlib.Visual_Ptr;
     depth    : in Xlib.UnsignedInt;
     format   : in Xlib.Int;
     data     : in Xlib.Pointer;
     shminfo  : in SegmentInfo;
     width    : in Xlib.UnsignedInt;
     height   : in Xlib.UnsignedInt ) return Xlib.Image_Ptr;

function CreatePixmap(
     dpy     : in Xlib.Display;
     d       : in Xlib.Drawable;
     shminfo : in SegmentInfo;
     width   : in Xlib.UnsignedInt;
     height  : in Xlib.UnsignedInt;
     depth   : in Xlib.UnsignedInt ) return Xlib.Pixmap;

function Detach(
     dpy      : in Xlib.Display;
     shminfo  : in SegmentInfo ) return Xlib.Status;

function GetEventBase(
     dpy         : in Xlib.Display ) return Xlib.Int;

function GetImage(
     dpy        : in Xlib.Display;
     draw       : in Xlib.Drawable;
     img        : in Xlib.Image_Ptr;
     x          : in Xlib.Int;
     y          : in Xlib.Int;
     plane_mask : in Xlib.UnsignedLong ) return Xlib.Status;

function PutImage(
     dpy        : in Xlib.Display;
     draw       : in Xlib.Drawable;
     agc        : in Xlib.GC;
     img        : in Xlib.Image_Ptr;
     src_x      : in Xlib.Int;
     src_y      : in Xlib.Int;
     dest_x     : in Xlib.Int;
     dest_y     : in Xlib.Int;
     src_width  : in Xlib.UnsignedInt;
     src_height : in Xlib.UnsignedInt;
     send_event : in Boolean ) return Xlib.Status;

procedure QueryExtension(
     dpy         : in     Xlib.Display;
```

```
      event_basep :    out Xlib.Int;
      error_basep :    out Xlib.Int;
      success     :    out Boolean );

  procedure QueryVersion(
      dpy           : in     Xlib.Display;
      major_version :    out Xlib.Int;
      minor_version :    out Xlib.Int;
      sharedPixmaps :    out Boolean;
      success       :    out Boolean );
```

Multi Buffering Extension

```
  procedure ChangeBufferAttributes(
      dpy        : in Xlib.Display;
      buffer     : in MultiBuffer;
      valuemask  : in Xlib.UnsignedLong;
      attributes : in BufferAttributes );


 procedure ChangeWindowAttributes(
      dpy        : in Xlib.Display;
      window     : in Xlib.Window;
      valuemask  : in Xlib.UnsignedLong;
      attributes : in WindowAttributes );

  function CreateBuffers(
      dpy            : in Xlib.Display;
      window         : in Xlib.Window;
      count          : in Xlib.Int;
      update_action  : in Xlib.Int;
      update_hint    : in Xlib.Int;

  function CreateStereoWindow(
      dpy          : in Xlib.Display;
      parent       : in Xlib.Window;
      x            : in Xlib.Int;
      y            : in Xlib.Int;
      width        : in Xlib.UnsignedInt;
      height       : in Xlib.UnsignedInt;
      border_width : in Xlib.UnsignedInt;
      depth        : in Xlib.Int;
      visual       : in Xlib.Visual_Ptr;
      valuemask    : in Xlib.UnsignedLong;
      attributes   : in Xlib.SetWindowAttributes;
      left_return  : in Multibuffer;
      right_return : in Multibuffer ) return Xlib.Window;

  procedure DestroyBuffers(
      dpy     : in Xlib.Display;
      window  : in Xlib.Window );

  procedure DisplayBuffers(
      dpy       : in Xlib.Display;
```

```
        count     : in Xlib.Int;
        buffers   : in MultibufferList;
        min_delay : in Xlib.Int;
        max_delay : in Xlib.Int );

    procedure DisplayBuffers(
        dpy       : in Xlib.Display;
        buffers   : in MultibufferList;
        min_delay : in Xlib.Int;
        max_delay : in Xlib.Int );

    procedure GetBufferAttributes(
        dpy        : in Xlib.Display;
        buffer     : in MultiBuffer;
        attributes : in BufferAttributes );

    procedure GetVersion(
        dpy           : in     Xlib.Display;
        major_version :    out Xlib.Int;
        minor_version :    out Xlib.Int;
        return_status :    out Xlib.Int );

    procedure GetWindowAttributes(
        dpy           : in     Xlib.Display;
        window        : in     Xlib.Window;
        attributes    :    out WindowAttributes;
        return_status :    out Xlib.Status );

procedure QueryExtension(
        dpy         : in     Xlib.Display;
        event_basep :    out Xlib.Int;
        error_basep :    out Xlib.Int;
        success     :    out Boolean );
```

Compound Text Extension

```
    function Create(
        str     : in String;
        length  : in Xlib.Int;
        flgs    : in Flags ) return Data;

    function Create(
        str     : in String;
        flgs    : in Flags ) return Data;

    procedure Free(
        dat : Data );

    function NextItem(
        dat : Data ) return Result;

    procedure Reset(
        dat : Data );
```

9

X Toolkit Intrinsics Interface

X Toolkit Routines

```
procedure AddActions(
     actions      : in ActionList;
     num_actions  : in Cardinal );

procedure AddCallback(
     object        : in Widget;
     callback_name : in String;
     callback      : in CallbackProc;
     client_data   : in Pointer );

procedure AddCallback(
     object        : in Widget;
     callback_name : in String;
     callback      : in CallbackProc;
     client_data   : in Int );

procedure AddCallbacks(
     object        : in Widget;
     callback_name : in String;
     callbacks     : in CallbackList );

procedure AddConverter(
     from_type     : in String;
     to_type       : in String;
     conv          : in ConverterProc;
     convert_args  : in ConvertArgList;
     num_args      : in Cardinal );

procedure AddEventHandler(
     w           : in Widget;
     event_mask  : in EventMask;
     nonmaskable : in Boolean;
     proc        : in EventHandler;
     client_data : in Pointer );

procedure AddExposureToRegion(
     event : in Xlib.Event;
     reg   : in Xlib.Region );

procedure AddGrab(
```

```
      w             : in Widget;
      exclusive     : in Boolean;
      spring_loaded : in Boolean );
   function AddInput(
      source      : in Int;
      condition   : in Int;
      proc        : in InputCallbackProc;
      client_data : in Pointer ) return InputId;

  procedure AddRawEventHandler(
      w           : in Widget;
      event_mask  : in EventMask;
      nonmaskable : in Boolean;
      proc        : in EventHandler;
      client_data : in Pointer );

  procedure AddTimeOut(
      interval    : in UnsignedLong;
      proc        : in WorkProc;
      client_data : in Pointer );

  procedure AddWorkProc(
      proc        : in WorkProc;
      client_data : in Pointer );

  procedure AppAddActionHook(
      app         : in AppContext;
      proc        : in ActionHookProc;
      client_data : in Pointer );

  procedure AppAddActions(
      app          : in AppContext;
      actions      : in ActionList;
      num_actions  : in Cardinal );

  procedure AppAddActions(
      app          : in AppContext;
      actions      : in ActionList );

  procedure AppAddConverter(
      app          : in AppContext;
      from_type    : in String;
      to_type      : in String;
      conv         : in ConverterProc;
      convert_args : in ConvertArgList;
      num_args     : in Cardinal );

  function AppAddInput(
      app_context : in AppContext;
      source      : in Int;
      condition   : in Int;
      proc        : in InputCallbackProc;
      client_data : in Pointer ) return InputId;
```

```
function AppAddTimeOut(
     app_context : in AppContext;
     interval    : in UnsignedLong;
     proc        : in TimerCallbackProc;
     client_data : in Pointer ) return IntervalId;

function AppAddWorkProc(
     app_context : in AppContext;
     proc        : in WorkProc;
     client_data : in Pointer ) return WorkProcId;

function AppAddWorkProc(
     app_context : in AppContext;
     proc        : in WorkProc;
     client_data : in Int ) return WorkProcId;

function AppCreateShell(
     application_name  : in String;
     application_class : in String;
     widget_class      : in WidgetClass;
     dpy               : in Xlib.Display;
     args              : in ArgList ) return Widget;

function AppCreateShell(
     application_name  : in String;
     application_class : in String;
     widget_class      : in WidgetClass;
     dpy               : in Xlib.Display;
     args              : in ArgList;
     num_args          : in Cardinal ) return Widget;

procedure AppError(
     app_context : in AppContext;
     message     : in String );

procedure AppErrorMsg(
     app_context : in AppContext;
     name        : in String;
     error_type  : in String;
     class       : in String;
     default     : in String;
     params      : in StringList;
     num_params  : in Cardinal );

procedure AppErrorMsg(
     app_context : in AppContext;
     name        : in String;
     error_type  : in String;
     class       : in String;
     default     : in String;
     params      : in StringList );

function AppGetErrorDatabase(
     app_context  : in AppContext ) return Xrm.Database;
```

```
  procedure AppGetErrorDatabaseText(
       app_context   : in     AppContext;
       name          : in     String;
       error_type    : in     String;
       class         : in     String;
       default       : in     String;
       buffer_return : in out String;
       nbytes        : in     Int;
       database      : in     Xrm.Database );

  procedure AppGetErrorDatabaseText(
       app_context   : in     AppContext;
       name          : in     String;
       error_type    : in     String;
       class         : in     String;
       default       : in     String;
       buffer_return : in out String;
       database      : in     Xrm.Database );

  function AppGetSelectionTimeout(
       app_context : AppContext ) return UnsignedInt;

procedure AppInitialize(
       app_context_return :    out AppContext;
       application_class  : in     String;
       options            : in     Xrm.OptionDescList;
       num_options        : in     Cardinal;
       argv_in_out        : in out StringList_Ptr;
       fallback_resources : in out String;
       args               : in     ArgList;
       num_args           : in     Cardinal;
       w                  :    out Widget );

  procedure AppInitialize(
       app_context_return :    out AppContext;
       application_class  : in     String;
       options            : in     Xrm.OptionDescList;
       argv_in_out        : in out StringList_Ptr;
       fallback_resources : in out String;
       args               : in     ArgList;
       w                  :    out Widget );

  procedure AppMainLoop(
       app_context  : AppContext );

  procedure AppNextEvent(
       app_context  : in     AppContext;
       event_return :   out Xlib.Event );

  procedure AppPeekEvent(
       app_context  : in     AppContext;
       event_return :    out Xlib.Event;
       x_event      :    out Boolean );
```

```
function AppPending(
     app_context  : in AppContext ) return InputMask;

procedure AppProcessEvent(
      app_context : in AppContext;
  mask        : in InputMask );

procedure AppReleaseCacheRefs(
     app_context : in AppContext;
     refs        : in CacheRef );

procedure AppSetErrorHandler(
     app_context  : in AppContext;
     handler      : in ErrorHandler );

procedure AppSetErrorMsgHandler(
     app_context  : in AppContext;
     msg_handler  : in ErrorMsgHandler );

procedure AppSetFallbackResources(
     app_context        : in AppContext;
     specification_list : in String );

procedure AppSetSelectionTimeout(
     app_context : in AppContext;
     timeout     : in UnsignedLong );

procedure AppSetTypeConverter(
     app_context    : in AppContext;
     from_type      : in String;
     to_type        : in String;
     converter      : in TypeConverter;
     converter_args : in ConvertArgList;
     num_args       : in Cardinal;
     cache_type     : in CacheType;
     destruct       : in Destructor );

procedure AppSetTypeConverter(
     app_context    : in AppContext;
     from_type      : in String;
     to_type        : in String;
     converter      : in TypeConverter;
     converter_args : in ConvertArgList;
     cache_type     : in CacheType;
     destruct       : in Destructor );

procedure AppSetWarningHandler(
     app_context : in AppContext;
     handler     : in ErrorHandler );

procedure AppSetWarningMsgHandler(
     app_context : in AppContext;
     msg_handler : in ErrorMsgHandler );
```

```
  procedure AppWarning(
       app_context : in AppContext;
       message     : in String );

  procedure AppWarningMsg(
       app_context  : in AppContext;
       name         : in String;
       warning_type : in String;
       class        : in String;
       default      : in String;
       params       : in StringList;
       num_params   : in Cardinal );

  procedure AppWarningMsg(
       app_context  : in AppContext;
       name         : in String;
       warning_type : in String;
       class        : in String;
       default      : in String;
       params       : in StringList );

  procedure AugmentTranslations(
       w         : in Widget;
       translats : in Translations );

  function BuildEventMask(
       w : in Widget ) return EventMask;

  function CallAcceptFocus(
       w    : in Widget;
       time : in Xlib.Time ) return Boolean;

  procedure CallActionProc(
       w          : in Widget;
       action     : in String;
       event      : in Xlib.Event;
       params     : in StringList;
       num_params : in Cardinal );

  procedure CallActionProc(
       w          : in Widget;
       action     : in String;
       event      : in Xlib.Event;
       params     : in StringList );
  procedure CallbackExclusive(
       w           : in Widget;
       client_data : in Pointer;
       call_data   : in Pointer );

  procedure CallbackNone(
       w           : in Widget;
       client_data : in Pointer;
       call_data   : in Pointer );
```

```
procedure CallbackNonexclusive(
     w           : in Widget;
     client_data : in Pointer;
     call_data   : in Pointer );

procedure CallbackPopdown(
     w           : in Widget;
     client_data : in Pointer;
     call_data   : in Pointer );

procedure CallbackReleaseCacheRef(
     object      : in Widget;
     client_data : in Pointer;
     call_data   : in Pointer );

procedure CallbackReleaseCacheRefList(
     object      : in Widget;
     client_data : in Pointer;
     call_data   : in Pointer );

procedure CallCallbackList(
     object     : in Widget;
     callbacks  : in CallbackList;
     call_data  : in Pointer );

procedure CallCallbacks(
     object        : in Widget;
     callback_name : in String;
     call_data     : in Pointer );

procedure CallConverter(
     dpy              : in Xlib.Display;
     converter        : in    TypeConverter;
     args             : in     Xrm.ValueList;
     num_args         : in     Cardinal;
     from             : in     Xrm.Value;
     to_in_out        : in out Xrm.Value;
     cache_ref_return : in out CacheRef;
     success          :    out Boolean );

function Calloc(
     num  : UnsignedInt;
     size : UnsignedInt ) return Pointer;

procedure CheckSubclass(
     object       : in Widget;
     object_class : in WidgetClass;
     message      : in String );

function Class(
     object : in Widget ) return WidgetClass;

procedure CloseDisplay(
```

```
      dpy : in Xlib.Display );

  procedure ConfigureWidget(
      w            : in Widget;
      x            : in Position;
      y            : in Position;
      width        : in Dimension;
      height       : in Dimension;
      border_width : in Dimension );

  procedure Convert(
      object    : in     Widget;
      from_type : in     String;
      from      : in     Xrm.Value;
      to_type   : in     String;
      to_return : in out Xrm.Value );

  procedure ConvertAndStore(
      object    : in     Widget;
      from_type : in     String;
      from      : in     Xrm.Value;
      to_type   : in     String;
      to_in_out : in out Xrm.Value;
      success   :    out Boolean );

  procedure ConvertCase(
      display      : in     Xlib.Display;
      keysym       : in     Xlib.KeySym;
      lower_return :    out Xlib.KeySym;
      upper_return :    out Xlib.KeySym );

  function CreateApplicationContext return AppContext;

  function CreateApplicationShell(
      name         : in String;     -- unused
      widget_class : in WidgetClass;
      args         : in ArgList;
      num_args     : in Cardinal ) return Widget;

  function CreateApplicationShell(
      name         : in String;     -- unused
      widget_class : in WidgetClass;
      args         : in ArgList ) return Widget;

  function CreateManagedWidget(
      name         : in String;
      widget_class : in WidgetClass;
      parent       : in Widget;
      args         : in ArgList;
      nargs        : in Cardinal ) return Widget;

  function CreateManagedWidget(
      name         : in String;
      widget_class : in WidgetClass;
```

```
     parent       : in Widget;
     args         : in ArgList ) return Widget;

function CreatePopupShell(
     name         : in String;
     widget_class : in WidgetClass;
     parent       : in Widget;
     args         : in ArgList;
     num_args     : in Cardinal ) return Widget;

function CreatePopupShell(
     name         : in String;
     widget_class : in WidgetClass;
     parent       : in Widget;
     args         : in ArgList ) return Widget;

function CreateWidget(
     name         : in String;
     object_class : in WidgetClass;
     parent       : in Widget;
     args         : in ArgList;
     num_args     : in Cardinal ) return Widget;

function CreateWidget(
     name         : in String;
     object_class : in WidgetClass;
     parent       : in Widget;
     args         : in ArgList ) return Widget;

procedure CreateWindow(
     name         : in String;
     window_class : in UnsignedInt;
     vis          : in Xlib.Visual_Ptr;
     value_mask   : in ValueMask;
     attributes   : in Xlib.SetWindowAttributes );

procedure DestroyApplicationContext(
       app_context : in AppContext );

procedure DestroyGC(
     gc : in Xlib.GC );

procedure DestroyWidget(
     object : in Widget );

procedure DirectConvert(
     convrter  : in     Converter;
     args      : in     Xrm.ValueList;
     num_args  : in     Cardinal;
     from      : in     Xrm.Value;
     to_return : in out Xrm.Value );

procedure DirectConvert(
     convrter  : in     Converter;
```

```
      args      : in     Xrm.ValueList;
      from      : in     Xrm.Value;
      to_return : in out Xrm.Value );

  procedure DisownSelection(
      widg      : in Widget;
      selection : in Xlib.Atom;
      time      : in Xlib.Time );

  function DispatchEvent(
      event : in Xlib.Event ) return Boolean;

  procedure DisplayInitialize(
      app_context       : in     AppContext;
      display           : in     Xlib.Display;
      application_name  : in     String;
      application_class : in     String;
      options           : in     Xrm.OptionDescList;
      num_options       : in     Cardinal;
      argv              : in out StringList_Ptr );

  procedure DisplayInitialize(
      app_context       : in     AppContext;
      display           : in     Xlib.Display;
      application_name  : in     String;
      application_class : in     String;
      options           : in     Xrm.OptionDescList;
      argv              : in out StringList_Ptr );

  function GetDisplay(
      w : in Widget ) return Xlib.Display;

  function DisplayOfObject(
      object : in Widget ) return Xlib.Display;

  procedure DisplayStringConversionWarning(
      display    : in Xlib.Display;
      from_value : in String;
      to_value   : in String );

  function DisplayToApplicationContext(
      display  : in Xlib.Display ) return AppContext;

  procedure Error(
      message : String );

  procedure ErrorMsg(
      name        : in String;
      error_type  : in String;
      class       : in String;
      default     : in String;
      params      : in StringList;
      num_params  : in Cardinal );
```

```
 procedure ErrorMsg(
      name        : in String;
      error_type  : in String;
      class       : in String;
      default     : in String;
      params      : in StringList );

 procedure FindFile(
      path              : in String;
      substitutions     : in SubstitutionList;
      num_substitutions : in Cardinal;
      predicate         : in FilePredicate );

 procedure Free(
      ptr : Pointer );

 procedure GetActionKeysym(
      event            : in     Xlib.Event;
      modifiers_return :    out Modifiers;
      key_sym          :    out Xlib.KeySym );

 procedure GetApplicationNameAndClass(
      dpy          : in     Xlib.Display;
      name_return  :    out String;
      class_return :    out String );

 procedure GetApplicationResources(
      object        : in Widget;
      base          : in Pointer;
      resources     : in ResourceList;
      num_resources : in Cardinal;
      args          : in ArgList;
      num_args      : in Cardinal );


procedure GetApplicationResources(
      object        : in Widget;
      base          : in Pointer;
      resources     : in ResourceList;
      args          : in ArgList );

 procedure GetConstraintResourceList(
      object_class         : in     WidgetClass;
      resources_return     : in out ResourceList_Ptr );

 function GetConstraintResourceList(
      object_class : in     WidgetClass ) return ResourceList_Ptr;

 function GetDatabase(
        display : in Xlib.Display ) return Xrm.Database;

 function GetErrorDatabase return Xrm.Database;

 procedure GetErrorDatabaseText(
```

```
      name          : in     String;
      error_type    : in     String;
      class         : in     String;
      default       : in     String;
      buffer_return :    out String;
  nbytes        : in     Int );

procedure GetErrorDatabaseText(
      name          : in     String;
      error_type    : in     String;
      class         : in     String;
      default       : in     String;
      buffer_return :    out String );

function GetGC(
     object     : in Widget;
     value_mask : in GCMask;
     values     : in Xlib.GCValues ) return Xlib.GC;

function GetKeysymTable(
     display                    : in Xlib.Display;
     min_keycode_return         : in Int;
     keysyms_per_keycode_return : in Int ) return Pointer;

procedure GetKeysymTable(
     display                    : in     Xlib.Display;
     min_keycode_return         :    out Int;
     keysyms_per_keycode_return :    out Int;
     keysms                     :    out Xlib.KeySymList_Ptr );

function GetMultiClickTime( display  : in Xlib.Display) return Int ;

function GetResourceList(
     object_class : in WidgetClass ) return ResourceList_Ptr;

function GetSelectionRequest(
     w          : in Widget;
     selection  : in Xlib.Atom;
     request_id : in RequestId ) return Xlib.Event_Ptr;

function GetSelectionTimeout return UnsignedInt;

procedure GetSelectionValue(
     w           : in Widget;
     selection   : in Xlib.Atom;
     target      : in Xlib.Atom;
     callback    : in SelectionCallbackProc;
     client_data : in Pointer;
     time        : in Xlib.Time );

procedure GetSelectionValueIncremental(
     w                 : in Widget;
     selection         : in Xlib.Atom;
     target            : in Xlib.Atom;
```

```
     selection_callback : in SelectionCallbackProc;
     client_data        : in Pointer;
     time               : in Xlib.Time );

procedure GetSelectionValues(
     w           : in Widget;
     selection   : in Xlib.Atom;
     targets     : in Xlib.AtomList;
     count       : in Int;
     callback    : in SelectionCallbackProc;
     client_data : in Pointer;
     time        : in Xlib.Time );

procedure GetSelectionValues(
     w           : in Widget;
     selection   : in Xlib.Atom;
     targets     : in Xlib.AtomList;
     callback    : in SelectionCallbackProc;
     client_data : in Pointer;
     time        : in Xlib.Time );

procedure GetSelectionValuesIncremental(
     w           : in Widget;
     selection   : in Xlib.Atom;
     targets     : in Xlib.AtomList;
     count       : in Int;
     callback    : in SelectionCallbackProc;
     client_data : in Pointer;
     time        : in Xlib.Time );

procedure GetSelectionValuesIncremental(
     w           : in Widget;
     selection   : in Xlib.Atom;
     targets     : in Xlib.AtomList;
     callback    : in SelectionCallbackProc;
     client_data : in Pointer;
     time        : in Xlib.Time );

procedure GetSubresources(
     object        : in Widget;
     base          : in Pointer;
     name          : in String;
     class         : in String;
     resources     : in ResourceList;
     num_resources : in Cardinal;
     args          : in ArgList;
     num_args      : in Cardinal );

procedure GetSubresources(
     object        : in Widget;
     base          : in Pointer;
     name          : in String;
     class         : in String;
     resources     : in ResourceList;
```

```
       args          : in ArgList );


procedure GetSubvalues(
       base          : in Pointer;
       resources     : in ResourceList;
       num_resources : in Cardinal;
       args          : in ArgList;
       num_args      : in Cardinal );

  procedure GetSubvalues(
       base          : in Pointer;
       resources     : in ResourceList;
       args          : in ArgList );

  procedure GetValues(
       object   : in Widget;
       args     : in ArgList;
       num_args : in Cardinal );

  procedure GetValues(
       object : in Widget;
       args   : in ArgList );

  procedure GrabButton(
       widg          : in Widget;
       button        : in Int;
       modifs        : in Modifiers;
       owner_events  : in Boolean;
       event_mask    : in UnsignedInt;
       pointer_mode  : in Int;
       keyboard_mode : in Int;
       confine_to    : in Xlib.Window;
       cursor        : in Xlib.Cursor );

  procedure GrabKey(
       widg          : in Widget;
       keycode       : in Xlib.KeyCode;
       modifs        : in Modifiers;
       owner_events  : in Boolean;
       pointer_mode  : in Int;
       keyboard_mode : in Int );

  procedure GrabKeyboard(
       widg          : in Widget;
       owner_events  : in Boolean;
       pointer_mode  : in Int;
       keyboard_mode : in Int;
       time          : in Xlib.Time );

  procedure GrabPointer(
       widg          : in Widget;
       owner_events  : in Boolean;
       event_mask    : in UnsignedInt;
```

```
     pointer_mode  : in Int;
     keyboard_mode : in Int;
     confine_to    : in Xlib.Window;
     cursor        : in Xlib.Cursor;
     time          : in Xlib.Time );

function HasCallbacks(
     widg          : in Widget;
     callback_name : in String ) return CallbackStatus;

procedure Initialize(
     shell_name  : in     String;
     application_class : in     String;
     options           : in     Xrm.OptionDescList;
     num_options       : in     Cardinal;
     argv              : in out StringList_Ptr;
     toplevel          :    out Widget );

procedure Initialize(
     shell_name  : in     String;
     application_class : in     String;
     options           : in     Xrm.OptionDescList;
     argv              : in out StringList_Ptr;
     toplevel          :    out Widget );

procedure InsertEventHandler(
     w           : in Widget;
     event_mask  : in EventMask;
     nonmaskable : in Boolean;
     proc        : in EventHandler;
     client_data : in Pointer;
     position    : in ListPosition );

procedure InsertRawEventHandler(
     w           : in Widget;
     event_mask  : in EventMask;
     nonmaskable : in Boolean;
     proc        : in EventHandler;
     client_data : in Pointer;
     position    : in ListPosition );

procedure InstallAccelerators(
     destination : in Widget;
     source      : in Widget );

procedure InstallAllAccelerators(
     destination : in Widget;
     source      : in Widget );

function IsApplicationShell(
     object : in Widget ) return Boolean;

function IsComposite(
     object : in Widget ) return Boolean;
```

```
  function IsConstraint(
       object : in Widget ) return Boolean;

  function IsManaged(
       object : in Widget ) return Boolean;

  function IsObject(
       object : in Widget ) return Boolean;

  function IsOverrideShell(
       object : in Widget ) return Boolean;

  function IsRealized(
       object : in Widget ) return Boolean;

  function IsRectObj(
       object : in Widget ) return Boolean;

  function IsSensitive(
       object : in Widget ) return Boolean;

  function IsShell(
       object : in Widget ) return Boolean;

  function IsSubclass(
       object       : in Widget;
       object_class : in WidgetClass ) return Boolean;

  function IsTopLevelShell(
       object : in Widget ) return Boolean;

  function IsTransientShell(
       object : in Widget ) return Boolean;

  function IsVendorShell(
       object : in Widget ) return Boolean;

  function IsWidget(
       object : in Widget ) return Boolean;

  function IsWMShell(
       object : in Widget ) return Boolean;

  procedure KeysymToKeycodeList(
       display         : in     Xlib.Display;
       keysym          : in     Xlib.KeySym;
       keycodes_return :    out Xlib.KeyCodeList_Ptr;
       keycount_return :    out Cardinal );

  function LastTimestampProcessed(
       display         : in Xlib.Display ) return Xlib.Time;

  procedure MainLoop;
```

```
procedure MakeGeometryRequest(
     w            : in     Widget;
     request      : in     WidgetGeometry;
     reply_return :    out WidgetGeometry;
     result       :    out GeometryResult );

procedure MakeResizeRequest(
     w             : in     Widget;
     width         : in     Dimension;
     height        : in     Dimension;
     width_return  : in out Dimension;
     height_return : in out Dimension;
     result        :    out GeometryResult );

function Malloc(
     size : in UnsignedInt ) return Pointer;

procedure ManageChild(
     w : in Widget );

procedure ManageChildren(
     children     : in WidgetList;
     num_children : in Cardinal );

procedure ManageChildren(
     children     : in WidgetList );

procedure MapWidget(
     w : in Widget );

function MergeArgLists(
     args1     : in ArgList;
     num_args1 : in Cardinal;
     args2     : in ArgList;
     num_args2 : in Cardinal ) return ArgList_Ptr;
function MergeArgLists(
     args1     : in ArgList;
     args2     : in ArgList ) return ArgList_Ptr;

procedure MoveWidget(
     w   : in Widget;
     x   : in Position;
     y   : in Position );

function Name(
     object : in Widget ) return String_Ptr;

function NameToWidget(
     reference : in Widget;
     names     : in String ) return Widget;

function NewString(
     str            : in String;
```

```
      null_terminate : in Boolean := True ) return String_Ptr;

  function NewString(
      str            : in Pointer;
      null_terminate : in Boolean := True ) return String_Ptr;

  procedure NextEvent(
      event_return : in out Xlib.Event );

  procedure OpenDisplay(
      app_context       : in     AppContext;
      display_name      : in     String;
      application_name  : in     String;
      application_class : in     String;
      options           : in     Xrm.OptionDescList;
      num_options       : in     Cardinal;
      argv              : in out StringList_Ptr;
      dpy               :    out Xlib.Display );

    procedure OpenDisplay(
      app_context       : in     AppContext;
      display_name      : in     String;
      application_name  : in     String;
      application_class : in     String;
      options           : in     Xrm.OptionDescList;
      argv              : in out StringList_Ptr;
      dpy               :    out Xlib.Display );

    procedure OpenDisplay(
      app_context       : in     AppContext;
      display_name      : in     String;
      application_name  : in     String;
      application_class : in     String;
      argv              : in out StringList_Ptr;
      dpy               :    out Xlib.Display );

  procedure OverrideTranslations(
      w         : in Widget;
      translats : in Translations );

  function OwnSelection(
      w            : in Widget;
      selection    : in Xlib.Atom;
      time         : in Xlib.Time;
      convert_proc : in ConvertSelectionProc;
      lose_proc    : in LoseSelectionProc;
      done_proc    : in SelectionDoneProc ) return Boolean;
  function OwnSelectionIncremental(
      w                : in Widget;
      selection        : in Xlib.Atom;
      time             : in Xlib.Time;
      convert_callback : in ConvertSelectionProc;
      lose_callback    : in LoseSelectionProc;
      done_callback    : in SelectionDoneProc;
```

```
     cancel_callback  : in CancelConvertSelectionProc;
     client_data      : in Pointer ) return Boolean;

function Parent(
     w : in Widget ) return Widget;

function ParseAcceleratorTable(
     table : in String ) return Accelerators;

function ParseTranslationTable(
     table : in String ) return Translations;

procedure PeekEvent(
     event_return : out Xlib.Event;
     x_event      : out Boolean );

function Pending return InputMask;

procedure Popdown(
    popup_shell : in Widget );

procedure Popup(
    popup_shell : in Widget;
    grab_kind   : in GrabKind );

procedure PopupSpringLoaded(
    popup_shell : in Widget );

procedure ProcessEvent(
     mask : in InputMask );

procedure QueryGeometry(
     w                : in     Widget;
     intended         : in     WidgetGeometry;
     preferred_return :    out WidgetGeometry;
     result           :    out GeometryResult );

procedure QueryGeometry(
     w                : in     Widget;
     preferred_return :    out WidgetGeometry;
     result           :    out GeometryResult );

procedure RealizeWidget(
     w : in Widget );

function Realloc(
     ptr : in Pointer;
     num : in UnsignedInt ) return Pointer;

procedure RegisterCaseConverter(
     display  : in Xlib.Display;
     proc     : in CaseProc;
     start    : in Xlib.KeySym;
     stop     : in Xlib.KeySym );
```

```
  procedure RegisterGrabAction(
       action_proc   : in ActionProc;
       owner_events  : in Boolean;
       event_mask    : in UnsignedInt;
       pointer_mode  : in Int;
       keyboard_mode : in Int );

  procedure ReleaseGC(
       object   : in Widget;
       gcontext : in Xlib.GC );

  procedure RemoveActionHook(
       id : in ActionHookId );

  procedure RemoveAllCallbacks(
       object        : in Widget;
       callback_name : in String );

  procedure RemoveCallback(
       object        : in Widget;
       callback_name : in String;
       callback      : in CallbackProc;
       client_data   : in Pointer );

  procedure RemoveCallbacks(
       object        : in Widget;
       callback_name : in String;
       callbacks     : in CallbackList );

  procedure RemoveEventHandler(
       w           : in Widget;
       event_mask  : in EventMask;
       nonmaskable : in Boolean;
       proc        : in EventHandler;
       client_data : in Pointer );

  procedure RemoveGrab(
       widg : in Widget );

  procedure RemoveInput(
       id : in InputId );

  procedure RemoveRawEventHandler(
       w           : in Widget;
       event_mask  : in EventMask;
       nonmaskable : in Boolean;
       proc        : in EventHandler;
       client_data : in Pointer );

  procedure RemoveTimeOut(
       id : in IntervalId );

  procedure RemoveWorkProc(
```

```
     id : in WorkProcId );

procedure ResizeWidget(
     w            : in Widget;
     width        : in Dimension;
     height       : in Dimension;
     border_width : in Dimension );

procedure ResizeWindow(
     w    : in Widget );

function ResolvePathname(
     display    : in Xlib.Display;
     file_type  : in String;
     filename   : in String;
     suffix     : in String;
     path       : in String;
     substs     : in SubstitutionList;
     num_substs : in Cardinal;
     predicate  : in FilePredicate ) return String_Ptr;

function GetScreen(
     w : Widget ) return Xlib.Screen_Ptr;

function ScreenOfObject(
     object : Widget ) return Xlib.Screen_Ptr;

procedure SetArg(
     a             : in Pointer;
     resource_name : in Pointer;
     value         : in ArgVal );
  pragma Interface( C, SetArg );

procedure SetArg(
     a             : in out Arg;
     resource_name : in     NullString;
     value         : in     Pointer );

procedure SetArg(
     a             : in out Arg;
     resource_name : in     NullString;
     value         : in     Boolean );

procedure SetArg(
     a             : in out Arg;
     resource_name : in     NullString;
     value         : in     FontStruct_Ptr );

procedure SetArg(
     a             : in out Arg;
     resource_name : in     NullString;
     value         : in     NullString );

procedure SetArg(
```

```
      a             : in out Arg;
      resource_name : in     NullString;
      value         : in     ArgVal );

  procedure SetArg(
      a             : in out Arg;
      resource_name : in     NullString;
      value         : in     Int );

  procedure SetErrorHandler(
      handler : in ErrorHandler );

  procedure SetErrorMsgHandler(
      msg_handler : in ErrorMsgHandler );

  procedure SetKeyboardFocus(
      subtree    : in Widget;
      descendant : in Widget );

  procedure SetKeyTranslator(
      display  : in Xlib.Display;
      proc     : in KeyProc );

  procedure SetMappedWhenManaged(
      w               : in Widget;
      map_when_managed : in Boolean );
  procedure SetMultiClickTime(
      display  : in Xlib.Display;
      time     : in Int );

  procedure SetSelectionTimeout(
      timeout : in UnsignedLong );

  procedure SetSensitive(
      w         : in Widget;
      sensitive : in Boolean );

  procedure SetSubvalues(
      base          : in Pointer;
      resources     : in ResourceList;
      num_resources : in Cardinal;
      args          : in ArgList;
      num_args      : in Cardinal );

  procedure SetSubvalues(
      base         : in Pointer;
      resources    : in ResourceList;
      args         : in ArgList );

  procedure SetTypeConverter(
      from_type    : in String;
      to_type      : in String;
      converter    : in TypeConverter;
      convert_args : in ConvertArgList;
```

```
     num_args     : in Cardinal;
     cache_type   : in CacheType;
     destruct     : in Destructor );

procedure SetTypeConverter(
     from_type    : in String;
     to_type      : in String;
     converter    : in TypeConverter;
     convert_args : in ConvertArgList;
     cache_type   : in CacheType;
     destruct     : in Destructor );

procedure SetValues(
     object   : in Widget;
     args     : in ArgList;
     num_args : in Cardinal );

procedure SetValues(
     object  : in Widget;
     args    : in ArgList );

procedure SetWarningHandler(
     handler : in ErrorHandler );

procedure SetWarningMsgHandler(
     msg_handler : in ErrorMsgHandler );

procedure SetWMColormapWindows(
     widg  : in Widget;
     list  : in WidgetList;
     count : in Cardinal );

procedure SetWMColormapWindows(
     widg  : in Widget;
     list  : in WidgetList );


procedure StringConversionWarning(
     src      : in String;
     dst_type : in String );

function Superclass(
     object : in Widget ) return WidgetClass;

procedure ToolkitInitialize;

procedure TranslateCoords(
     w             : in     Widget;
     x             : in     Position;
     y             : in     Position;
     root_x_return :    out Position;
     root_y_return :    out Position );

procedure TranslateKey(
```

```
      display          : in     Xlib.Display;
      keycode          : in     Xlib.KeyCode;
      modifs           : in     Modifiers;
      modifiers_return :    out Modifiers;
      keysym_return    :    out Xlib.KeySym );

  procedure TranslateKeycode(
      display          : in     Xlib.Display;
      keycode          : in     Xlib.KeyCode;
      modifs           : in     Modifiers;
      modifiers_return :    out Modifiers;
      keysym_return    :    out Xlib.KeySym );

  procedure UngrabButton(
      widg   : in Widget;
      button : in UnsignedInt;
      modifs : in Modifiers );

  procedure UngrabKey(
      widg    : in Widget;
      keycode : in Xlib.KeyCode;
      modifs  : in Modifiers );

  procedure UngrabKeyboard(
      widg  : in Widget;
      time  : in Xlib.Time );

  procedure UngrabPointer(
      widg  : in Widget;
      time  : in Xlib.Time );

  procedure UninstallTranslations(
      w : in Widget );

  procedure UnmanageChild(
      w : in Widget );

  procedure UnmanageChildren(
      children     : in WidgetList;
      num_children : in Cardinal );

  procedure UnmanageChildren(
      children : in WidgetList );

  procedure UnmapWidget(
      w : in Widget );


  procedure UnrealizeWidget(
      w : in Widget );

  procedure Warning(
      message : in String );
```

```
procedure WarningMsg(
     name          : in String;
     warning_type  : in String;
     class         : in String;
     default       : in String;
     params        : in StringList;
     num_params    : in Cardinal );

procedure WarningMsg(
     name          : in String;
     warning_type  : in String;
     class         : in String;
     default       : in String;
     params        : in StringList );

function WidgetToApplicationContext(
     object : in Widget ) return AppContext;

function GetWindow(
     w   : in Widget ) return Xlib.Window;

function WindowOfObject(
     object : in Widget ) return Xlib.Window;

function WindowToWidget(
     dpy    : in Xlib.Display;
     window : in Xlib.Window ) return Widget;
```

X Toolkit Convenience Functions

```
procedure BuildAction(
     res  :    out ActionRec;
     str  : in     NullString );

procedure BuildResource(
     res            : in out Resource;
     resource_name  : in     NullString;
     resource_class : in     NullString;
     resource_type  : in     NullString;
```

X11 Release 5 Toolkit Routines

```
procedure SetLanguageProc(
     app_context : in AppContext;
     proc        : in LanguageProc;
     client_data : in Pointer );
     default_type   : in     NullString );
```

10

X Miscellaneous Utility Routines

Atom Functions

```
function GetAtomName(
     d    : in Xlib.Display;
     atom : in Xlib.Atom ) return Xlib.String_Ptr;

function InternAtom(
     d        : in Xlib.Display;
     atom_ptr : in AtomPtr ) return Xlib.Atom;

procedure InternStrings(
     d     : in     Xlib.Display;
     names : in     Xt.StringList;
     count : in     Xt.Cardinal;
     atoms :    out Xlib.AtomList );

procedure InternStrings(
     d     : in     Xlib.Display;
     names : in     Xt.StringList;
     atoms :    out Xlib.AtomList );

function MakeAtom(
     name  : in String ) return AtomPtr;

function NameOfAtom(
     atom_ptr : in AtomPtr ) return Xlib.String_Ptr;
```

Error Handler Functions

```
function PrintDefaultErrorMessage(
     dpy   : in Xlib.Display;
     event : in Xlib.Event_Ptr;
     fp    : in Xlib.Pointer ) return Xlib.Int;

function SimpleErrorHandler(
     dpy   : in Xlib.Display;
     event : in Xlib.Event_Ptr ) return Xlib.Int;
```

System Utility Functions

```
procedure GetHostname(
     buf    : in out String;
     length :    out Xlib.Int );
```

Window Utility Functions

```
function ClientWindow(
     dpy   : in Xlib.Display;
     win   : in Xlib.Window ) return Xlib.Window;

function ScreenOfWindow(
     dpy   : in Xlib.Display;
     win   : in Xlib.Window ) return Xlib.Screen_Ptr;

function UpdateMapHints(
     dpy   : in Xlib.Display;
     win   : in Xlib.Window;
     hints : in Xlib.SizeHints ) return Boolean;
```

Cursor Utility Functions

```
function CursorNameToIndex(
     name  : in String ) return Xlib.Int;
```

Draw Functions

```
function CreatePixmapFromBitmap(
     dpy     : in Xlib.Display;
     d       : in Xlib.Drawable;
     bitmap  : in Xlib.Pixmap;
     width   : in Xlib.UnsignedInt;
     height  : in Xlib.UnsignedInt;
     depth   : in Xlib.UnsignedInt;
     fore    : in Xlib.UnsignedLong;
     back    : in Xlib.UnsignedLong ) return Xlib.Pixmap;

function CreateStippledPixmap(
     dpy    : in Xlib.Display;
     fore   : in Xt.Pixel;
     back   : in Xt.Pixel;
     depth  : in Xlib.UnsignedInt ) return Xlib.Pixmap;

procedure DrawLogo(
     dpy      : in Xlib.Display;
     drawable : in Xlib.Drawable;
     gcFore   : in Xlib.GC;
     gcBack   : in Xlib.GC;
     x        : in Xlib.Int;
     y        : in Xlib.Int;
     width    : in Xlib.Int;
     height   : in Xlib.Int );

procedure DrawRoundedRectangle(
     dpy   : in Xlib.Display;
     draw  : in Xlib.Drawable;
     gc    : in Xlib.GC;
     x     : in Xlib.Int;
     y     : in Xlib.Int;
     w     : in Xlib.Int;
```

```
      h     : in Xlib.Int;
      ew    : in Xlib.Int;
      eh    : in Xlib.Int );

  procedure FillRoundedRectangle(
      dpy   : in Xlib.Display;
      draw  : in Xlib.Drawable;
      gc    : in Xlib.GC;
      x     : in Xlib.Int;
      y     : in Xlib.Int;
      w     : in Xlib.Int;
      h     : in Xlib.Int;
      ew    : in Xlib.Int;
      eh    : in Xlib.Int );

  procedure LocateBitmapFile(
      screen     : in     Xlib.Screen_Ptr;
      name       : in     String;
      srcname    :    out String;
      scrnamelen : in     Xlib.Int;
      width      :    out Xlib.Int;
      height     :    out Xlib.Int;
      xhot       :    out Xlib.Int;
      yhot       :    out Xlib.Int;
      pixmap     :    out Xlib.Pixmap );

  function ReadBitmapData(
      fstream : in Xlib.Pointer;
      width   : in Xlib.Pointer;
      height  : in Xlib.Pointer;
      datap   : in Xlib.Pointer;
      x_hot   : in Xlib.Pointer;
      y_hot   : in Xlib.Pointer ) return Xlib.Int;

  procedure ReleaseStippledPixmap(
      screen : in Xlib.Screen_Ptr;
      pixmap : in Xlib.Pixmap );
```

Selection Functions

```
  procedure ConvertStandardSelection(
      w           : in     Xt.Widget;
      time        : in     Xlib.Time;
      selection   :    out Xlib.Atom;
      target      :    out Xlib.Atom;
      select_type :    out Xlib.Atom;
      value       : in out Xlib.Pointer;
      length      :    out Xlib.UnsignedLong;
      format      :    out Xlib.Int;
      success     :    out Boolean );
```

Converter Functions

```
  procedure AddFunctionToCallbackConverter;
```

```
  procedure AddStringToBackingStoreConverter;

  procedure AddStringToBitmapConverter;

  procedure AddStringToCursorConverter;

  procedure AddStringToJustifyConverter;

  procedure AddStringToLongConverter;

  procedure AddStringToOrientationConverter;

  procedure AddStringToShapeStyleConverter;

  procedure AddStringToWidgetConverter;

  procedure AddAllConverters;

  function ReshapeWidget(
       w             : in Xt.Widget;
       shape_style   : in Xlib.Int;
       corner_width  : in Xlib.Int;
       corner_height : in Xlib.Int ) return Boolean;
```

Character Set Functions

```
  procedure CompareISOLatin1(
       first  : in String;
       second : in String );

  procedure CopyISOLatin1Lowered(
       dst : in out String;
       src : in     String );

  procedure CopyISOLatin1Uppered(
       dst : in out String;
       src : in     String );

  function LookupAPL(
       ev        : in Xlib.Event;
       buffer    : in String;
       num_bytes : in Xlib.Int;
       key_sym   : in Xlib.KeySym;
       stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupAPL(
       ev        : in Xlib.Event;
       buffer    : in String;
       key_sym   : in Xlib.KeySym;
       stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupArabic(
       ev        : in Xlib.Event;
```

```
     buffer    : in String;
     num_bytes : in Xlib.Int;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupArabic(
     ev        : in Xlib.Event;
     buffer    : in String;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupCyrillic(
     ev        : in Xlib.Event;
     buffer    : in String;
     num_bytes : in Xlib.Int;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupCyrillic(
     ev        : in Xlib.Event;
     buffer    : in String;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupGreek(
     ev        : in Xlib.Event;
     buffer    : in String;
     num_bytes : in Xlib.Int;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupGreek(
     ev        : in Xlib.Event;
     buffer    : in String;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupHebrew(
     ev        : in Xlib.Event;
     buffer    : in String;
     num_bytes : in Xlib.Int;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupHebrew(
     ev        : in Xlib.Event;
     buffer    : in String;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupJISX0201(
     ev        : in Xlib.Event;
     buffer    : in String;
     num_bytes : in Xlib.Int;
```

```
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupJISX0201(
      ev        : in Xlib.Event;
      buffer    : in String;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupKana(
      ev        : in Xlib.Event;
      buffer    : in String;
      num_bytes : in Xlib.Int;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupKana(
      ev        : in Xlib.Event;
      buffer    : in String;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupLatin1(
      ev        : in Xlib.Event;
      buffer    : in String;
      num_bytes : in Xlib.Int;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupLatin1(
      ev        : in Xlib.Event;
      buffer    : in String;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupLatin2(
      ev        : in Xlib.Event;
      buffer    : in String;
      num_bytes : in Xlib.Int;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupLatin2(
      ev        : in Xlib.Event;
      buffer    : in String;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;

  function LookupLatin3(
      ev        : in Xlib.Event;
      buffer    : in String;
      num_bytes : in Xlib.Int;
      key_sym   : in Xlib.KeySym;
      stat      : in Xlib.ComposeStatus ) return Xlib.Int;
```

```
function LookupLatin3(
     ev        : in Xlib.Event;
     buffer    : in String;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupLatin4(
     ev        : in Xlib.Event;
     buffer    : in String;
     num_bytes : in Xlib.Int;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;

function LookupLatin4(
     ev        : in Xlib.Event;
     buffer    : in String;
     key_sym   : in Xlib.KeySym;
     stat      : in Xlib.ComposeStatus ) return Xlib.Int;
```

Close Display Hook Functions

```
function AddCloseDisplayHook(
     dpy  : in Xlib.Display;
     func : in IntFunc;
     arg  : in Xlib.Pointer ) return CloseHook;

function LookupCloseDisplayHook(
     dpy    : in Xlib.Display;
     handle : in CloseHook;
     func   : in IntFunc;
     arg    : in Xlib.Pointer ) return Boolean;

function RemoveCloseDisplayHook(
     dpy    : in Xlib.Display;
     handle : in CloseHook;
     func   : in IntFunc;
     arg    : in Xlib.Pointer ) return Boolean;
```

Display Queue Functions

```
function DQAddDisplay(
     q    : in DisplayQueue_Ptr;
     dpy  : in Xlib.Display;
     data : Xlib.Pointer ) return DisplayQueueEntry_Ptr;


function DQCreate(
     closefunc : in IntFunc;
     freefunc  : in IntFunc;
     data      : in Xt.Pointer ) return DisplayQueue_Ptr;

function DQDestroy(
     q           : in DisplayQueue_Ptr;
     docallbacks : in Boolean ) return Boolean;
```

```
function DQLookupDisplay(
     q    : in DisplayQueue_Ptr;
     dpy  : in Xlib.Display ) return DisplayQueueEntry_Ptr;

function DQNDisplays(
     q    : in DisplayQueue_Ptr ) return Xt.Int;

function DQRemoveDisplay(
     q    : in DisplayQueue_Ptr;
     dpy  : in Xlib.Display ) return Boolean;
```

Toolkit Convenience Functions

```
procedure AddInitializer(
     func  : in VoidFunc;
     data  : in Xlib.Pointer );

procedure CallInitializers(
     app_con : Xt.AppContext );
```

Colormap Utility Functions

```
function AllStandardColormaps(
     dpy : in Xlib.Display ) return Xlib.Status;

function CreateColormap(
     dpy      : in Xlib.Display;
     colormap : in Xlib.StandardColormap ) return Xlib.Status;

procedure DeleteStandardColormap(
     dpy      : in Xlib.Display;
     screen   : in Xlib.Int;
     property : in Xlib.Atom );

function GetColormapAllocation(
     vinfo     : in Xlib.VisualInfo_Ptr;
     property  : in Xlib.Atom;
     red_max   : in Xlib.UnsignedLong;
     green_max : in Xlib.UnsignedLong;
     blue_max  : in Xlib.UnsignedLong ) return Xlib.Status;

procedure GetColormapAllocation(
     vinfo     : in     Xlib.VisualInfo_Ptr;
     property  : in     Xlib.Atom;
     red_max   :    out Xlib.UnsignedLong;
     green_max :    out Xlib.UnsignedLong;
     blue_max  :    out Xlib.UnsignedLong;
     status    :    out Xlib.Status );

function LookupStandardColormap(
     dpy      : in Xlib.Display;
     screen   : in Xlib.Int;
     visualid : in Xlib.VisualId;
     depth    : in Xlib.UnsignedInt;
```

```
     property : in Xlib.Atom;
     replace  : in Boolean;
     retain   : in Boolean ) return Xlib.Status;

function StandardColormap(
     dpy       : in Xlib.Display;
     screen    : in Xlib.Int;
     visualid  : in Xlib.VisualId;
     depth     : in Xlib.UnsignedInt;
     property  : in Xlib.Atom;
     cmap      : in Xlib.Colormap;
     red_max   : in Xlib.UnsignedLong;
     green_max : in Xlib.UnsignedLong;
     blue_max  : in Xlib.UnsignedLong ) return Xlib.StandardColormap_Ptr;

function VisualStandardColormaps(
     dpy      : in Xlib.Display;
     screen   : in Xlib.Int;
     visualid : in Xlib.VisualId;
     depth    : in Xlib.UnsignedInt;
     replace  : in Boolean;
     retain   : in Boolean ) return Xlib.Status;
```

X11 Release 5 Routines

```
 function DistinguishableColors(
     colors : in Xlib.ColorList;
     count  : in Xlib.Int ) return Boolean;

function DistinguishableColors(
     colors : in Xlib.ColorList ) return Boolean;

function DistinguishablePixels(
     dpy    : in Xlib.Display;
     cmap   : in Xlib.Colormap;
     pixels : in Xt.PixelList;
     count  : in Xlib.Int ) return Boolean;

function DistinguishablePixels(
     dpy    : in Xlib.Display;
     cmap   : in Xlib.Colormap;
     pixels : in Xlib.PixelList ) return Boolean;

procedure LocatePixmapFile(
     screen     : in     Xlib.Screen_Ptr;
     name       : in     String;
     fore       : in     Xlib.UnsignedLong;
     back       : in     Xlib.UnsignedLong;
     depth      : in     Xlib.UnsignedInt;
     srcname    :    out String;
     srcnamelen : in     Xlib.Int;
     widthp     :    out Xlib.Int;
     heightp    :    out Xlib.Int;
     xhotp      :    out Xlib.Int;
```

```
      yhotp      :    out Xlib.Int;
      pixmap_return : out Xlib.Pixmap );

  procedure LocatePixmapFile(
      screen     : in     Xlib.Screen_Ptr;
      name       : in     String;
      fore       : in     Xlib.UnsignedLong;
      back       : in     Xlib.UnsignedLong;
      depth      : in     Xlib.UnsignedInt;
      srcname    :    out String;
      widthp     :    out Xlib.Int;
      heightp    :    out Xlib.Int;
      xhotp      :    out Xlib.Int;
      yhotp      :    out Xlib.Int;
      pixmap_return : out Xlib.Pixmap );


  procedure ReshapeWidget(
      w             : in Xt.Widget;
      shape_style   : in Xt.Int;
      corner_width  : in Xt.Int;
      corner_height : in Xt.Int );

  function WnCountOwnedResources(
      node        : in WidgetNode;
      owner_node  : in WidgetNode;
      constraints : in Boolean ) return Xlib.Int;

  procedure WnFetchResources(
      node     : in WidgetNode;
      toplevel : in Xt.Widget;
      top_node : in WidgetNode );

  procedure WnInitializeNodes(
      node_array : in out WidgetNodeList;
      num_nodes  : in     Xlib.Int );

  procedure WnInitializeNodes(
      node_array : in out WidgetNodeList );

  function WnNameToNode(
      node_list : in WidgetNodeList;
      num_nodes : in Xlib.Int;
      name      : in String ) return WidgetNode_Ptr;

  function WnNameToNode(
      node_list : in WidgetNodeList;
      name      : in String ) return WidgetNode_Ptr;
```

11

Motif Interface

Motif Interface Routines

```
procedure ActivateProtocol(
     shell    : in Xt.Widget;
     property : in Xlib.Atom;
     protocol : in Xlib.Atom );

procedure ActivateWMProtocol(
     shell    : in Xt.Widget;
     protocol : in Xlib.Atom );

procedure AddProtocolCallback(
     shell    : in Xt.Widget;
     property : in Xlib.Atom;
     protocol : in Xlib.Atom;
     callback : in Xt.CallbackProc;
     closure  : in Xt.Pointer );

procedure AddProtocols(
     shell         : in Xt.Widget;
     property      : in Xlib.Atom;
     protocols     : in Xlib.AtomList;
     num_protocols : in Xt.Cardinal );

procedure AddProtocols(
     shell     : in Xt.Widget;
     property  : in Xlib.Atom;
     protocols : in Xlib.AtomList );

procedure AddTabGroup(
     tab_group : in Xt.Widget );

procedure AddWMProtocolCallback(
     shell    : in Xt.Widget;
     protocol : in Xlib.Atom;
     callback : in Xt.CallbackProc;
     closure  : in Xt.Pointer );

procedure AddWMProtocols(
     shell         : in Xt.Widget;
     protocols     : in Xlib.AtomList;
     num_protocols : in Xt.Cardinal );
```

```
  procedure AddWMProtocols(
       shell     : in Xt.Widget;
       protocols : in Xlib.AtomList );

  procedure CascadeButtonGadgetHighlight(
       button    : in Xt.Widget;
       highlight : in Boolean );

  procedure CascadeButtonHighlight(
       button    : in Xt.Widget;
       highlight : in Boolean );

  procedure ClipboardCancelCopy(
       dpy     : in Xlib.Display;
       w       : in Xlib.Window;
       item_id : in Long );

  function ClipboardCopy(
       dpy         : in Xlib.Display;
       win         : in Xlib.Window;
       item_id     : in Long;
       format_name : in String;
       buffer      : in String;
       length      : in UnsignedLong;
       private_id  : in Int;
       data_id     : in Int ) return Int;

  function ClipboardCopyByName(
       dpy         : in Xlib.Display;
       w           : in Xlib.Window;
       data_id     : in Int;
       buffer      : in String;
       length      : in UnsignedLong;
       private_id  : in Int ) return Int;

  function ClipboardEndCopy(
       dpy     : in Xlib.Display;
       w       : in Xlib.Window;
       item_id : in Long ) return Int;

  function ClipboardEndRetrieve(
       dpy     : in Xlib.Display;
       w       : in Xlib.Window ) return Int;

  function ClipboardInquireCount(
       dpy                   : in Xlib.Display;
       w                     : in Xlib.Window;
       count                 : in Int;
       max_format_name_length : in Int ) return Int;

  function ClipboardInquireFormat(
       display         : in Xlib.Display;
       win             : in Xlib.Window;
```

```
     index           : in Int;
     format_name_buf : in String;
     buffer_len      : in UnsignedLong;
     copied_len      : in UnsignedLong ) return Int;

function ClipboardInquireFormat(
     display         : in Xlib.Display;
     win             : in Xlib.Window;
     index           : in Int;
     format_name_buf : in String;
     copied_len      : in UnsignedLong ) return Int;



function ClipboardInquireLength(
     dpy         : in Xlib.Display;
     win         : in Xlib.Window;
     format_name : in String;
     length      : in UnsignedLong ) return Int;

function ClipboardInquirePendingItems(
     dpy      : in Xlib.Display;
     w           : in Xlib.Window;
     format_name : in String;
     item_list   : in ClipboardPendingList;
     length      : in UnsignedLong ) return Int;

function ClipboardLock(
     dpy : in Xlib.Display;
     w   : in Xlib.Window ) return Int;

function ClipboardRegisterFormat(
     dpy      : in Xlib.Display;
     format_name   : in String;
     format_length : in UnsignedLong ) return Int;

function ClipboardRetrieve(
     display    : in Xlib.Display;
     win         : in Xlib.Window;
     format_name : in String;
     buffer      : in String;
     length      : in UnsignedLong;
     num_bytes   : in UnsignedLong;
     private_id  : in Int ) return Int;

function ClipboardRetrieve(
     display    : in Xlib.Display;
     win         : in Xlib.Window;
     format_name : in String;
     buffer      : in String;
     num_bytes   : in UnsignedLong;
     private_id  : in Int ) return Int;

function ClipboardStartCopy(
```

```
      display    : in Xlib.Display;
      w   : in Xlib.Window;
      clip_label  : in XmString;
      timestamp   : in Xlib.Time;
      widget      : in Xt.Widget;
      callback    : in VoidProc;
      item_id     : in Long ) return Int;

  function ClipboardStartRetrieve(
      display   : in Xlib.Display;
      window    : in Xlib.Window;
      timestamp : in Xlib.Time ) return Int;

  function ClipboardUndoCopy(
      dpy : in Xlib.Display;
      w   : in Xlib.Window ) return Int;

  function ClipboardUnlock(
      dpy              : in Xlib.Display;
      win         : in Xlib.Window;
      remove_all_locks : in Boolean ) return Int;

  function ClipboardWithdrawFormat(
      dpy     : in Xlib.Display;
      w       : in Xlib.Window;
      data_id : in Xt.Cardinal ) return Int;

  procedure CommandAppendValue(
      widget  : in Xt.Widget;
      command : in Pointer );

  procedure CommandError(
      widget : in Xt.Widget;
      error  : in Pointer );

  function CommandGetChild(
      widget : Xt.Widget;
      child  : UnsignedChar ) return Xt.Widget;

  procedure CommandSetValue(
      widget  : in Xt.Widget;
      command : in Pointer );

  function ConvertUnits(
      widg           : in Xt.Widget;
      orientation    : in Int;
      from_unit_type : in Int;
      from_value     : in Int;
      to_unit_type   : in Int ) return Int;

  function CreateArrowButton(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
```

```
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateArrowButton(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateArrowButtonGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateArrowButtonGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateBulletinBoard(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateBulletinBoard(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateBulletinBoardDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;


function CreateBulletinBoardDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateCascadeButton(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateCascadeButton(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateCascadeButtonGadget(
     parent   : in Xt.Widget;
```

```
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateCascadeButtonGadget(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateCommand(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateCommand(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateDialogShell(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateDialogShell(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateDrawingArea(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateDrawingArea(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateDrawnButton(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;
  function CreateDrawnButton(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateErrorDialog(
      parent   : in Xt.Widget;
```

```
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateErrorDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateFileSelectionBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateFileSelectionBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateFileSelectionDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateFileSelectionDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateForm(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateForm(
     parent   : in Xt.Widget;
     name     : in String;
     arglist: in Xt.ArgList ) return Xt.Widget;

function CreateFormDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateFormDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateFrame(
```

```
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;
  function CreateFrame(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateInformationDialog(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateInformationDialog(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateLabel(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateLabel(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateLabelGadget(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateLabelGadget(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateList(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateList(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateMainWindow(
```

```
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateMainWindow(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateMenuBar(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;
function CreateMenuBar(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateMenuShell(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateMenuShell(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateMessageBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateMessageBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateMessageDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateMessageDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateOptionMenu(
```

```
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateOptionMenu(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreatePanedWindow(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreatePanedWindow(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreatePopupMenu(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;
  function CreatePopupMenu(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreatePromptDialog(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreatePromptDialog(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreatePulldownMenu(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreatePulldownMenu(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreatePushButton(
```

```
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreatePushButton(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreatePushButtonGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreatePushButtonGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateQuestionDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateQuestionDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateRadioBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;
function CreateRadioBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateRowColumn(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateRowColumn(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateScale(
```

```
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateScale(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateScrollBar(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateScrollBar(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateScrolledList(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateScrolledList(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateScrolledText(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateScrolledText(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateScrolledWindow(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList;
      argcount : in Xt.Cardinal ) return Xt.Widget;
  function CreateScrolledWindow(
      parent   : in Xt.Widget;
      name     : in String;
      arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateSelectionBox(
```

```
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateSelectionBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateSelectionDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateSelectionDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateSeparator(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateSeparator(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateSeparatorGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateSeparatorGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateSimpleCheckBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateSimpleCheckBox(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;
```

```
  function CreateSimpleMenuBar(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;
  function CreateSimpleMenuBar(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateSimpleOptionMenu(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateSimpleOptionMenu(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateSimplePopupMenu(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateSimplePopupMenu(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateSimplePulldownMenu(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateSimplePulldownMenu(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateSimpleRadioBox(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateSimpleRadioBox(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;
```

```
function CreateText(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateText(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateTextField(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;
function CreateTextField(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateToggleButton(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateToggleButton(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateToggleButtonGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateToggleButtonGadget(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;

function CreateWarningDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList;
     argcount : in Xt.Cardinal ) return Xt.Widget;

function CreateWarningDialog(
     parent   : in Xt.Widget;
     name     : in String;
     arglist  : in Xt.ArgList ) return Xt.Widget;
```

```
  function CreateWorkArea(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateWorkArea(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;

  function CreateWorkingDialog(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList;
       argcount : in Xt.Cardinal ) return Xt.Widget;

  function CreateWorkingDialog(
       parent   : in Xt.Widget;
       name     : in String;
       arglist  : in Xt.ArgList ) return Xt.Widget;

  function CvtCTToXmString(
       text  : in String ) return XmString;

  procedure CvtStringToUnitType(
       args     : in Xrm.ValueList;
       num_args : in Xt.Cardinal;
       from_val : in Xrm.Value;
       to_val   : in Xrm.Value );

  procedure CvtStringToUnitType(
       args     : in Xrm.ValueList;
       from_val : in Xrm.Value;
       to_val   : in Xrm.Value );

  function CvtXmStringToCT(
       str  : in XmString ) return String_Ptr;

  procedure DeactivateProtocol(
       shell    : in Xt.Widget;
       property : in Xlib.Atom;
       protocol : in Xlib.Atom );

  procedure DeactivateWMProtocol(
       shell    : in Xt.Widget;
       protocol : in Xlib.Atom );

  procedure DestroyPixmap(
       screen  : in Xlib.Screen_Ptr;
       pixmap  : in Xlib.Pixmap );

  function FileSelectionBoxGetChild(
       widg  : in Xt.Widget;
```

```
     child : in UnsignedChar ) return Xt.Widget;

procedure FileSelectionDoSearch(
      widg    : in Xt.Widget;
      dirmask : in XmString );

function FontListAdd(
     oldlist : FontList;
     font    : Xlib.FontStruct_Ptr;
     charset : StringCharSet ) return FontList;

function FontListCreate(
     font    : Xlib.FontStruct_Ptr;
     charset : StringCharSet ) return FontList;

function FontListCopy(
     font_list : FontList ) return FontList;

procedure FontListFree(
     list : in FontList );

procedure FontListFreeFontContext(
     context : in FontContext );

function FontListGetNextFont(
     context : in FontContext;
     charset : in StringCharSet;
     font    : in Xlib.FontStruct_Ptr ) return Boolean;

function FontListInitFontContext(
     context   : in FontContext;
     font_list : in FontList ) return Boolean;

function GetAtomName(
    dpy : in Xlib.Display;
    a   : in Xlib.Atom ) return String_Ptr;


function GetColorCalculation return ColorProc;

procedure GetColors(
     screen        : in     Xlib.Screen_Ptr;
     colormap      : in     Xlib.Colormap;
     background    : in     Xt.Pixel;
     foreground    : in out Xt.Pixel;
     top_shadow    : in out Xt.Pixel;
     bottom_shadow : in out Xt.Pixel;
     selct         : in out Xt.Pixel );

function GetDestination(
     dpy : in Xlib.Display ) return Xt.Widget;

function GetMenuCursor(
     dpy : in Xlib.Display ) return Xlib.Cursor;
```

```
  function GetPixmap(
       screen     : in Xlib.Screen_Ptr;
       image_name : in String;
       foreground : in Xt.Pixel;
       background : in Xt.Pixel ) return Xlib.Pixmap;

  function GetPostedFromWidget(
       menu  : in Xt.Widget ) return Xt.Widget;

  function InstallImage(
        image        : in Xlib.Image_Ptr;
        image_name : in String ) return Boolean;

  function InternAtom(
       dpy            : in Xlib.Display;
       name           : in String;
       only_if_exists : in Boolean ) return Xlib.Atom;

  function IsBulletinBoard(
       w  : in Xt.Widget ) return Boolean;

  function IsCascadeButton(
       w  : in Xt.Widget ) return Boolean;

  function IsCascadeButtonGadget(
       w  : in Xt.Widget ) return Boolean;

  function IsCommandBox(
       w  : in Xt.Widget ) return Boolean;

  function IsDialogShell(
       w  : in Xt.Widget ) return Boolean;

  function IsDrawnButton(
       w  : in Xt.Widget ) return Boolean;

  function IsExtObject(
       w  : in Xt.Widget ) return Boolean;

  function IsForm(
       w  : in Xt.Widget ) return Boolean;

  function IsGadget(
       w  : in Xt.Widget ) return Boolean;

  function IsLabel(
       w  : in Xt.Widget ) return Boolean;

  function IsLabelGadget(
       w  : in Xt.Widget ) return Boolean;

  function IsList(
       w  : in Xt.Widget ) return Boolean;
```

```
function IsMainWindow(
     w  : in Xt.Widget ) return Boolean;

function IsManager(
     w  : in Xt.Widget ) return Boolean;

function IsMenuShell(
     w  : in Xt.Widget ) return Boolean;

function IsMotifWMRunning(
     shell : in Xt.Widget ) return Boolean;

function IsPrimitive(
     w  : in Xt.Widget ) return Boolean;

function IsPushButton(
     w  : in Xt.Widget ) return Boolean;

function IsPushButtonGadget(
     w  : in Xt.Widget ) return Boolean;

function IsRowColumn(
     w  : in Xt.Widget ) return Boolean;

function IsScrolledWindow(
     w  : in Xt.Widget ) return Boolean;

function IsScrollBar(
     w  : in Xt.Widget ) return Boolean;

function IsSelectionBox(
     w  : in Xt.Widget ) return Boolean;

function IsSeparator(
     w  : in Xt.Widget ) return Boolean;

function IsSeparatorGadget(
     w  : in Xt.Widget ) return Boolean;

function IsText(
     w  : in Xt.Widget ) return Boolean;

function IsTextField(
     w  : in Xt.Widget ) return Boolean;

function IsToggleButton(
     w  : in Xt.Widget ) return Boolean;

function IsToggleButtonGadget(
     w  : in Xt.Widget ) return Boolean;

procedure ListAddItem(
     widg     : in Xt.Widget;
```

```
      item     : in XmString;
      position : in Int );

  procedure ListAddItems(
      widget     : in Xt.Widget;
      items      : in XmStringList;
      position   : in Int );

  procedure ListAddItems(
      widget     : in Xt.Widget;
      items      : in XmStringList;
      item_count : in Int;
      position   : in Int );

  procedure ListAddItemUnselected(
      widg     : in Xt.Widget;
      item     : in XmString;
      position : in Int );

  procedure ListDeleteAllItems(
      widget : in Xt.Widget );

  procedure ListDeleteItem(
      widget : in Xt.Widget;
      item   : in XmString );

  procedure ListDeleteItems(
      widget     : in Xt.Widget;
      items      : in XmStringList;
      item_count : in Int );

  procedure ListDeleteItems(
      widget     : in Xt.Widget;
      items      : in XmStringList );

  procedure ListDeleteItemsPos(
      widg       : in Xt.Widget;
      item_count : in Int;
      position   : in Int );

  procedure ListDeletePos(
      widg     : in Xt.Widget;
      position : in Int );

  procedure ListDeselectAllItems(
      widget : in Xt.Widget );

  procedure ListDeselectItem(
      widget : in Xt.Widget;
      item   : in XmString );

  procedure ListDeselectPos(
      widget   : in Xt.Widget;
      position : in Int );
```

```
procedure ListGetMatchPos(
     widget         : in     Xt.Widget;
     item           : in     XmString;
     position_list  :    out IntList;
     position_count :    out Int );

procedure ListGetSelectedPos(
     widget         : in     Xt.Widget;
     position_list  :    out IntList;
     position_count :    out Int );

function ListItemExists(
     widget : in Xt.Widget;
     item   : in XmString ) return Boolean;


function ListItemPos(
     widget : in Xt.Widget;
     item   : in XmString ) return Int;

procedure ListReplaceItems(
     widget     : in Xt.Widget;
     old_items  : in XmStringList;
     item_count : in Int;
     new_items  : in XmStringList );

procedure ListReplaceItems(
     widget     : in Xt.Widget;
     old_items  : in XmStringList;
     new_items  : in XmStringList );

procedure ListReplaceItemsPos(
     widget     : in Xt.Widget;
     new_items  : in XmStringList;
     item_count : in Int;
     position   : in Int );

procedure ListReplaceItemsPos(
     widget     : in Xt.Widget;
     new_items  : in XmStringList;
     position   : in Int );

procedure ListSelectItem(
     widget : in Xt.Widget;
     item   : in XmString;
     notify : in Boolean );

procedure ListSelectPos(
     widget   : in Xt.Widget;
     position : in Int;
     notify   : in Boolean );

procedure ListSetAddMode(
```

```
      widget : in Xt.Widget;
      mode   : in Boolean );

  procedure ListSetBottomItem(
      widget : in Xt.Widget;
      item   : in XmString );

  procedure ListSetBottomPos(
      widget   : in Xt.Widget;
      position : in Int );

  procedure ListSetHorizPos(
      widget   : in Xt.Widget;
      position : in Int );

  procedure ListSetItem(
      widget : in Xt.Widget;
      item   : in XmString );

  procedure ListSetPos(
      widg     : in Xt.Widget;
      position : in Int );

  function MainWindowSep1(
      widget : Xt.Widget ) return Xt.Widget;

  function MainWindowSep2(
      widget : Xt.Widget ) return Xt.Widget;
  function MainWindowSep3(
      widget : Xt.Widget ) return Xt.Widget;

  procedure MainWindowSetAreas(
      widget               : in Xt.Widget;
      menu_bar             : in Xt.Widget;
      command_window       : in Xt.Widget;
      horizontal_scrollbar : in Xt.Widget;
      vertical_scrollbar   : in Xt.Widget;
      work_region          : in Xt.Widget );

  procedure MenuPosition(
      widget : in Xt.Widget;
      event  : in Xlib.Event );

  function MessageBoxGetChild(
      widget : in Xt.Widget;
      child  : in UnsignedChar ) return Xt.Widget;

  function OptionButtonGadget(
      option_menu : in Xt.Widget ) return Xt.Widget;

  function OptionLabelGadget(
      option_menu : in Xt.Widget ) return Xt.Widget;

  function ProcessTraversal(
```

```
     widget    : in Xt.Widget;
     direction : in Int ) return Boolean;

procedure RegisterConverters;

procedure RemoveProtocolCallback(
     shell    : in Xt.Widget;
     property : in Xlib.Atom;
     protocol : in Xlib.Atom;
     callback : in Xt.CallbackProc;
     closure  : in Xt.Pointer );

procedure RemoveProtocols(
     shell         : in Xt.Widget;
     property      : in Xlib.Atom;
     protocols     : in Xlib.AtomList;
     num_protocols : in Xt.Cardinal );

procedure RemoveProtocols(
     shell     : in Xt.Widget;
     property  : in Xlib.Atom;
     protocols : in Xlib.AtomList );

procedure RemoveTabGroup(
     tab_group : in Xt.Widget );

procedure RemoveWMProtocolCallback(
     shell    : in Xt.Widget;
     protocol : in Xlib.Atom;
     callback : in Xt.CallbackProc;
     closure  : in Xt.Pointer );

procedure RemoveWMProtocols(
     shell         : in Xt.Widget;
     property      : in Xlib.Atom;
     protocols     : in Xlib.AtomList;
     num_protocols : in Xt.Cardinal );


procedure RemoveWMProtocols(
     shell     : in Xt.Widget;
     property  : in Xlib.Atom;
     protocols : in Xlib.AtomList );

procedure ResolveAllPartOffsets(
     widget_class      : in Xt.WidgetClass;
     offst             : in Offset;
     constraint_offset : in Offset );

procedure ResolvePartOffsets(
     widget_class : in Xt.WidgetClass;
     offst        : in Offset );

procedure ScaleGetValue(
```

```
     widget       : in     Xt.Widget;
     value_return :    out Int );

  procedure ScaleSetValue(
       widget : in Xt.Widget;
       value  : in Int );

  procedure ScrollBarGetValues(
       widget        : in     Xt.Widget;
       value         :    out Int;
       slider_size   :    out Int;
       increment     :    out Int;
       page_increment :   out Int );

  procedure ScrollBarSetValues(
       widget         : in Xt.Widget;
       value          : in Int;
       slider_size    : in Int;
       increment      : in Int;
       page_increment : in Int;
       notify         : in Boolean );

  procedure ScrolledWindowSetAreas(
       widget              : in Xt.Widget;
       horizonal_scrollbar : in Xt.Widget;
       vertical_scrollbar  : in Xt.Widget;
       work_region         : in Xt.Widget );

  function SelectionBoxGetChild(
       widget : Xt.Widget;
       child  : UnsignedChar ) return Xt.Widget;

  function SetColorCalculation(
       color_proc  : ColorProc ) return ColorProc;

  procedure SetFontUnit(
       dpy            : in Xlib.Display;
       font_unit_value : in Int );

  procedure SetFontUnits(
       dpy     : in Xlib.Display;
       h_value : in Int;
       v_value : in Int );

  procedure SetMenuCursor(
       dpy      : Xlib.Display;
       cursorId : Xlib.Cursor );

  procedure SetProtocolHooks(
       shell        : in Xt.Widget;
       property     : in Xlib.Atom;
       protocol     : in Xlib.Atom;
       prehook      : in Xt.CallbackProc;
       pre_closure  : in Xt.Pointer;
```

```
     posthook     : in Xt.CallbackProc;
     post_closure : in Xt.Pointer );

procedure SetWMProtocolHooks(
     shell        : in Xt.Widget;
     protocol     : in Xlib.Atom;
     prehook      : in Xt.CallbackProc;
     pre_closure  : in Xt.Pointer;
     posthook     : in Xt.CallbackProc;
     post_closure : in Xt.Pointer );

function StringBaseline(
     font_list : in FontList;
     string    : in XmString ) return Xt.Dimension;

function StringByteCompare(
     s1  : in XmString;
     s2  : in XmString ) return Boolean;

function StringCompare(
     s1  : in XmString;
     s2  : in XmString ) return Boolean;

function StringConcat(
     s1  : in XmString;
     s2  : in XmString ) return XmString;

function StringCopy(
     s1  : in XmString ) return XmString;

function StringCreate(
     text    : in String;
     charset : in StringCharSet ) return XmString;

function StringCreateLtoR(
     text    : in String;
     charset : in StringCharSet ) return XmString;

function StringCreateSimple(
     text    : in String ) return XmString;

function StringDirectionCreate(
     direction : in StringDirection ) return XmString;

procedure StringDraw(
     dpy              : in Xlib.Display;
     w     : in Xlib.Window;
     list         : in FontList;
     str    : in XmString;
     gcontext    : in Xlib.GC;
     x                : in Xt.Position;
     y                : in Xt.Position;
     width    : in Xt.Dimension;
     alignment   : in UnsignedChar;
```

```
      layout_direction : in UnsignedChar;
      clip             : in Xlib.Rectangle );

  procedure StringDrawImage(
      dpy              : in Xlib.Display;
      w                : in Xlib.Window;
      list             : in FontList;
      str              : in XmString;
      gcontext         : in Xlib.GC;
      x                : in Xt.Position;
      y                : in Xt.Position;
      width            : in Xt.Dimension;
      alignment        : in UnsignedChar;
      layout_direction : in UnsignedChar;
      clip             : in Xlib.Rectangle );

  procedure StringDrawUnderline(
      dpy              : in Xlib.Display;
      w    : in Xlib.Window;
      list       : in FontList;
      str    : in XmString;
      gcontext   : in Xlib.GC;
      x             : in Xt.Position;
      y             : in Xt.Position;
      width    : in Xt.Dimension;
      alignment  : in UnsignedChar;
      layout_direction : in UnsignedChar;
      clip             : in Xlib.Rectangle;
      underline        : in XmString );

  function StringEmpty(
      s1 : in XmString ) return Boolean;

  procedure StringExtent(
      font_list : in     FontList;
      str       : in     XmString;
      width     :    out Xt.Dimension;
      height    :    out Xt.Dimension );

  procedure StringFree(
      str : in XmString );

  procedure StringFreeContext(
      context : in StringContext );

  procedure StringGetLtoR(
      str     : in     XmString;
      charset : in     StringCharSet;
      text    : in out String_Ptr;
      success :    out Boolean );

  procedure StringGetNextComponent(
      context       : in     StringContext;
      text          : in out String_Ptr;
```

```
     charset       : in out StringCharSet;
     direction     :    out StringDirection;
     unkown_tag    :    out StringComponentType;
     unkown_length :    out UnsignedShort;
     unkown_value  : in out String_Ptr;
     comp_type     :    out StringComponentType );

procedure StringGetNextSegment(
     context     : in     StringContext;
     text        : in out String_Ptr;
     charset     : in     StringCharSet;
     direction   :    out StringDirection;
     separator   :    out Boolean;
     success     :    out Boolean );

function StringHasSubstring(
     string    : in XmString;
     substring : in XmString ) return Boolean;
function StringHeight(
     s1 : in XmString ) return Xt.Dimension;

function StringInitContext(
     context : in StringContext;
     string  : in XmString ) return Boolean;

function StringLength(
     s1 : in XmString ) return Int;

function StringLineCount(
     str : in XmString ) return Int;

function StringNConcat(
     s1        : in XmString;
     s2        : in XmString;
     num_bytes : in Int ) return XmString;

function StringNCopy(
     s1        : in XmString;
     num_bytes : in Int ) return XmString;

function StringPeekNextComponent(
     context : in StringContext ) return StringComponentType;

function StringSegmentCreate(
     text      : in String;
     charset   : in StringCharSet;
     direction : in StringDirection;
     separator : in Boolean ) return XmString;

function StringSeparatorCreate return XmString;

function StringWidth(
     font_list : in FontList;
     str       : in XmString ) return Xt.Dimension;
```

```
  procedure TextClearSelection(
       widget : in Xt.Widget;
       t      : in Xlib.Time );

  function TextCopy(
       widget : in Xt.Widget;
       time   : in Xlib.Time ) return Boolean;

  function TextCut(
       widget : in Xt.Widget;
       time   : in Xlib.Time ) return Boolean;

  procedure TextFieldClearSelection(
       widget : in Xt.Widget;
       t      : in Xlib.Time );

  function TextFieldCopy(
       widget : in Xt.Widget;
       time   : in Xlib.Time ) return Boolean;

  function TextFieldCut(
       widget : in Xt.Widget;
       time   : in Xlib.Time ) return Boolean;

  function TextFieldGetBaseline(
       widget : in Xt.Widget ) return Int;


  function TextFieldGetEditable(
       widget : in Xt.Widget ) return Boolean;

  function TextFieldGetInsertionPosition(
       widget : in Xt.Widget ) return TextPosition;

  function TextFieldGetLastPosition(
       widget : in Xt.Widget ) return TextPosition;

  function TextFieldGetMaxLength(
       widget : in Xt.Widget ) return Int;

  function TextFieldGetSelection(
       widget : in Xt.Widget ) return String_Ptr;

  function TextFieldGetSelectionPosition(
       widget : in Xt.Widget;
       left   : in TextPosition;
       right  : in TextPosition ) return Boolean;

  function TextFieldGetString(
       widget : in Xt.Widget ) return String_Ptr;

  procedure TextFieldInsert(
       widget   : in Xt.Widget;
```

```
     position : in TextPosition;
     value    : in String );

function TextFieldPaste(
     widget   : in Xt.Widget ) return Boolean;

function TextFieldPosToXY(
     widget   : in Xt.Widget;
     position : in TextPosition;
     x        : in Pointer;
     y        : in Pointer ) return Boolean;

function TextFieldRemove(
     widget   : in Xt.Widget ) return Boolean;

procedure TextFieldReplace(
     widget   : in Xt.Widget;
     from_pos : in TextPosition;
     to_pos   : in TextPosition;
     value    : in String );

procedure TextFieldSetAddMode(
     widget : in Xt.Widget;
     state  : in Boolean );

procedure TextFieldSetEditable(
     widget   : in Xt.Widget;
     editable : in Boolean );

procedure TextFieldSetHighlight(
     widget  : in Xt.Widget;
     left    : in TextPosition;
     right   : in TextPosition;
     mode    : in HighlightMode );

procedure TextFieldSetInsertPosition(
     widget   : in Xt.Widget;
     position : in TextPosition );


procedure TextFieldSetMaxLength(
     widget     : in Xt.Widget;
     max_length : in Int );

procedure TextFieldSetSelection(
     widget  : in Xt.Widget;
     first   : in TextPosition;
     last    : in TextPosition;
     time    : in Xlib.Time );

procedure TextFieldSetString(
     widget  : in Xt.Widget;
     value   : in String );
```

```
  procedure TextFieldShowPosition(
       widget   : in Xt.Widget;
       position : in TextPosition );


  function TextFieldXYToPos(
       widget   : in Xt.Widget;
       x        : in Xt.Position;
       y        : in Xt.Position ) return TextPosition;

  function TextGetBaseline(
       widget : in Xt.Widget ) return Int;

  function TextGetEditable(
       widget : in Xt.Widget ) return Boolean;

  function TextGetInsertionPosition(
       widget : in Xt.Widget ) return TextPosition;

  function TextGetLastPosition(
       widget : in Xt.Widget ) return TextPosition;

  function TextGetMaxLength(
       widget : in Xt.Widget ) return Int;

  function TextGetSelection(
       widget : in Xt.Widget ) return Pointer;

  function TextGetSelectionPosition(
       widget : in Xt.Widget;
       left   : in TextPosition;
       right  : in TextPosition ) return Boolean;

  function TextGetSource(
       widget : in Xt.Widget ) return TextSource;

  function TextGetString(
       widget : in Xt.Widget ) return String_Ptr;

  function TextGetTopCharacter(
       widget   : in Xt.Widget ) return TextPosition;

  procedure TextInsert(
       widget   : in Xt.Widget;
       position : in TextPosition;
       value    : in String );

  function TextPaste(
       widget   : in Xt.Widget ) return Boolean;


  function TextPosToXY(
       widget   : in Xt.Widget;
       position : in TextPosition;
```

```
     x        : in Pointer;
     y        : in Pointer ) return Boolean;

function TextRemove(
     widget   : in Xt.Widget ) return Boolean;

procedure TextReplace(
     widget   : in Xt.Widget;
     from_pos : in TextPosition;
     to_pos   : in TextPosition;
     value    : in String );

procedure TextScroll(
     widget : in Xt.Widget;
     lines  : in Int );

procedure TextSetAddMode(
     widget : in Xt.Widget;
     state  : in Boolean );

procedure TextSetEditable(
     widget   : in Xt.Widget;
     editable : in Boolean );

procedure TextSetHighlight(
     widget  : in Xt.Widget;
     left    : in TextPosition;
     right   : in TextPosition;
     mode    : in HighlightMode );

procedure TextSetInsertPosition(
     widget   : in Xt.Widget;
     position : in TextPosition );

procedure TextSetMaxLength(
     widget     : in Xt.Widget;
     max_length : in Int );

procedure TextSetSelection(
     widget  : in Xt.Widget;
     first   : in TextPosition;
     last    : in TextPosition;
     time    : in Xlib.Time );

procedure TextSetSource(
     widget          : in Xt.Widget;
     source          : in TextPosition;
     top_character   : in TextPosition;
     cursor_position : in Xlib.Time );

procedure TextSetString(
     widget  : in Xt.Widget;
     value   : in String );
```

```
  procedure TextSetTopCharacter(
       widget        : in Xt.Widget;
       top_character : in TextPosition );

  procedure TextShowPosition(
       widget   : in Xt.Widget;
       position : in TextPosition );

  function TextXYToPos(
       widget   : in Xt.Widget;
       x        : in Xt.Position;
       y        : in Xt.Position ) return TextPosition;

  function ToggleButtonGadgetGetState(
       widget : in Xt.Widget ) return Boolean;

  procedure ToggleButtonGadgetSetState(
       widget  : in Xt.Widget;
       state   : in Boolean;
       notify  : in Boolean );

  function ToggleButtonGetState(
       widget : in Xt.Widget ) return Boolean;

  procedure ToggleButtonSetState(
       widget  : in Xt.Widget;
       state   : in Boolean;
       notify  : in Boolean );

  procedure TrackingLocate(
       widget     : in Xt.Widget;
       cursor     : in Xlib.Cursor;
       confine_to : in Boolean );

  function UninstallImage(
       img : in Xlib.Image_Ptr ) return Boolean;

  procedure UpdateDisplay(
       widget : in Xt.Widget );
```

12

Motif Resource Manager

Motif Resource Manager Routines

```
procedure BuildRegisterArg(
     reg_arg :    out RegisterArg;
     name    : in     String;
     val     : in      Xm.Pointer );

function CloseHierarchy(
     hierarchy_id  : in Hierarchy ) return Xt.Cardinal;

function FetchColorLiteral(
     hierarchy_id  : in Hierarchy;
     index         : in String;
     display       : in Xlib.Display;
     colormap_id   : in Xlib.Colormap;
     pix           : in Xt.Pixel ) return Xm.Int;

function FetchIconLiteral(
     hierarchy_id  : in Hierarchy;
     index         : in String;
     display       : in Xlib.Display;
     screen        : in Xlib.Screen_Ptr;
     fgpix         : in Xt.Pixel;
     bgpix         : in Xt.Pixel;
     pixmap        : in Xlib.Pixmap ) return Xm.Int;

function FetchLiteral(
     hierarchy_id  : in Hierarchy;
     index         : in String;
     display       : in Xlib.Display;
     screen        : in Xlib.Screen_Ptr;
     value         : in Xm.Pointer;
     literal_type  : in Code ) return Xm.Int;

function FetchSetValues(
     hierarchy_id  : in Hierarchy;
     widget        : in Xt.Widget;
     args          : in Xt.ArgList;
     num_args      : in Xt.Cardinal ) return Xm.Int;

function FetchSetValues(
     hierarchy_id  : in Hierarchy;
```

```
      widget        : in Xt.Widget;
      args          : in Xt.ArgList ) return Xm.Int;

  procedure FetchWidget(
      hierarchy_id  : in     Hierarchy;
      index         : in     String;
      parent_widget : in     Xt.Widget;
      widget        : in out Xt.Widget;
      class         :    out MrmType;
      status        :    out Xm.Int );

  procedure FetchWidgetOverride(
      hierarchy_id      : in     Hierarchy;
      index             : in     String;
      parent_widget     : in     Xt.Widget;
      override_name     : in     String;
      override_args     : in     Xt.ArgList;
      override_num_args : in     Xt.Cardinal;
      widget            : in out Xt.Widget;
      class             :    out MrmType;
      status            :    out Xm.Int );

  procedure FetchWidgetOverride(
      hierarchy_id      : in     Hierarchy;
      index             : in     String;
      parent_widget     : in     Xt.Widget;
      override_name     : in     String;
      override_args     : in     Xt.ArgList;
      widget            : in out Xt.Widget;
      class             :    out MrmType;
      status            :    out Xm.Int );

  procedure Initialize;

  procedure OpenHierarchy(
      num_files                : in     Count;
      file_names_list          : in     Xm.StringList_Ptr;
      ancillary_structures_list : in     Xm.Pointer;
      hierarchy_id             :    out Hierarchy;
      status                   :    out Xm.Int );

  procedure OpenHierarchy(
      file_names_list          : in     Xm.StringList_Ptr;
      ancillary_structures_list : in     Xm.Pointer;
      hierarchy_id             :    out Hierarchy;
      status                   :    out Xm.Int );

  function RegisterClass(
      class_code   : in MrmType;
      class_name   : in String;
      create_name  : in String;
      create_proc  : in CreateProc;
      class_record : in Xt.WidgetClass ) return Xm.Int;
```

```
function RegisterNames(
     register_list  : in RegisterArgList;
     register_count : in Count ) return Xm.Int;

function RegisterNames(
     register_list  : in RegisterArgList ) return Xm.Int;

function RegisterNamesInHierarchy(
     hierarchy_id   : in Hierarchy;
     register_list  : in RegisterArgList;
     register_count : in Count ) return Xm.Int;

function RegisterNamesInHierarchy(
     hierarchy_id   : in Hierarchy;
     register_list  : in RegisterArgList ) return Xm.Int;
```

13

STARS Binding Routines

Package X_Lib

```
function  Copy_Visual_From_Parent
    return Visual;

function  Zero_Address
    return System.Address;

function  Get_Command_Line_Arguments
    return String_List;

function  X_Open_Display           (
        Name            : in String  )
    return Display;

procedure X_No_Op                  (
        Display_Id      : in Display  );

procedure X_Close_Display          (
        The_Display     : in out Display  );

function  Bitmap_Bit_Order         (
        Display_Id      : in Display  )
    return Order_Type;

function  Bitmap_Pad               (
        Display_Id      : in Display  )
    return Pixel;

function  Bitmap_Unit              (
        Display_Id      : in Display  )
    return Pixel;

function  Black_Pixel              (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number  )
    return Color_Value;

function  Connection_Number        (
        Display_Id      : in Display  )
    return X_Long_Integer;
```

```
    function  Default_Colormap         (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Colors.Color_Map;

    procedure X_List_Depths            (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number ;
            Count           : in out Depth_List  );

    function  Default_Depth            (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Depth_Type;

    function  Default_Gc               (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Graphic_Output.Graphic_Context;

    function  Default_Root_Window      (
            Display_Id      : in Display  )
        return Window;

    function  Default_Screen           (
            Display_Id      : in Display  )
        return Screen_Number;

    function  X_Screen_Number_Of_Screen(
            Screen_Id       : in Screen  )
        return Screen_Number;

    function  Default_Visual           (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Visual;

    function  Display_Cells            (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Natural;

    function  Display_Height           (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Width_Height;

    function  Display_Height_Mm        (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Millimeters;

    function  X_Display_Name           (
            Name            : in String  )
```

```
    return String;

function  Display_Planes           (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number  )
    return Depth_Type;

function  Display_String           (
        Display_Id      : in Display  )
    return String;

function  Display_Width            (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number  )
    return Width_Height;

function  Display_Width_Mm         (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number  )
    return Millimeters;

function  Image_Byte_Order         (
        Display_Id      : in Display  )
    return Order_Type;

function  Last_Known_Request_Processed(
        Display_Id      : in Display  )
    return X_Long_Integer;

function  Next_Request             (
        Display_Id      : in Display  )
    return X_Long_Integer;

function  Protocol_Version         (
        Display_Id      : in Display  )
    return Natural;

function  Protocol_Revision        (
        Display_Id      : in Display  )
    return Natural;

function  Q_Length                 (
        Display_Id      : in Display  )
    return Natural;

function  Root_Window              (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number  )
    return Window;

function  Screen_Count             (
        Display_Id      : in Display  )
    return Screen_Number;
```

```
    function  Server_Vendor            (
            Display_Id      : in Display  )
        return String;

    function  Vendor_Release           (
            Display_Id      : in Display  )
        return Natural;

    function  White_Pixel              (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Color_Value;

    function  Screen_Of_Display        (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number  )
        return Screen;

    function  Default_Screen_Of_Display(
            Display_Id      : in Display  )
        return Screen;

    function  Display_Of_Screen        (
            Screen_Id       : in Screen  )
        return Display;

    function  Root_Window_Of_Screen    (
            Screen_Id       : in Screen  )
        return Window;

    function  Black_Pixel_Of_Screen    (
            Screen_Id       : in Screen  )
        return Color_Value;

    function  White_Pixel_Of_Screen    (
            Screen_Id       : in Screen  )
        return Color_Value;

    function  Default_Colormap_Of_Screen(
            Screen_Id       : in Screen  )
        return Colors.Color_Map;

    function  Default_Depth_Of_Screen  (
            Screen_Id       : in Screen  )
        return Depth_Type;

    function  Default_Gc_Of_Screen     (
            Screen_Id       : in Screen  )
        return Graphic_Output.Graphic_Context;

    function  Default_Visual_Of_Screen (
            Screen_Id       : in Screen  )
        return Visual;
```

```
    function  Width_Of_Screen          (
            Screen_Id       : in Screen  )
        return Width_Height;

    function  Height_Of_Screen         (
            Screen_Id       : in Screen  )
        return Width_Height;

    function  Width_Mm_Of_Screen       (
            Screen_Id       : in Screen  )
        return Millimeters;

    function  Height_Mm_Of_Screen      (
            Screen_Id       : in Screen  )
        return Millimeters;

    function  Planes_Of_Screen         (
            Screen_Id       : in Screen  )
        return Natural;

    function  Cells_Of_Screen          (
            Screen_Id       : in Screen  )
        return Natural;

    function  Min_Cmaps_Of_Screen      (
            Screen_Id       : in Screen  )
        return Natural;

    function  Max_Cmaps_Of_Screen      (
            Screen_Id       : in Screen  )
        return Natural;

    function  Does_Save_Unders         (
            Screen_Id       : in Screen  )
        return Boolean;

    function  Does_Backing_Store       (
            Screen_Id       : in Screen  )
        return Backing_Store_Type;

    function  Event_Mask_Of_Screen     (
            Screen_Id       : in Screen  )
        return Events.Event_Mask_Type;

--  function  Min_Keycode              (
--          Display_Id      : in Display  )
--      return Keyboard.Keycode;
--  --* This function has been deleted because it accesses a
--  --* particular component of the display record in the
--  --* STARS implementation. This component is not accessed in
--  --* the C version of X11R4, and hence is not accessed in AXI.
--  --* While it could be accessed to complete the STARS binding,
--  --* we feel that the absence from the C interface indicates
--  --* an intended abstraction, so we have deleted the associated
```

```
--  --* STARS function.

--  function  Max_Keycode              (
--          Display_Id      : in Display  )
--      return Keyboard.Keycode;
--  --* This function has been deleted because it accesses a
--  --* particular component of the display record in the
--  --* STARS implementation. This component is not accessed in
--  --* the C version of X11R4, and hence is not accessed in AXI.
--  --* While it could be accessed to complete the STARS binding,
--  --* we feel that the absence from the C interface indicates
--  --* an intended abstraction, so we have deleted the associated
--  --* STARS function.

--  function  Server_Defaults          (
--          Display_Id      : Display   )
--      return String;
--  --* This function has been deleted because it accesses a
--  --* particular component of the display record in the
--  --* STARS implementation. This component is not accessed in
--  --* the C version of X11R4, and hence is not accessed in AXI.
--  --* While it could be accessed to complete the STARS binding,
--  --* we feel that the absence from the C interface indicates
--  --* an intended abstraction, so we have deleted the associated
--  --* STARS function.

    function  X_Max_Request_Size       (
            Display_Id      : in Display  )
        return X_Long_Integer;

    function  X_Get_Visual_Info        (
            Display_Id      : in Display ;
            Info_Mask       : in Visual_Mask_Type ;
            Info_Template   : in Visual_Info_Record  )
        return Visual_Info_List;

    function  X_Visual_Id_From_Visual  (
            Visual_Id       : in Visual  )
        return Visual_Id_Type;

    procedure X_Match_Visual_Info      (
            Display_Id      : in Display ;
            Screen_No       : in Screen_Number ;
            Depth_Of_Screen : in Depth_Type ;
            Class_Of_Screen : in Visual_Class_Type ;
            Visual_Info     : out Visual_Info_Record ;
            Success         : out Boolean  );

    type Text_Property_Record is
       record
          Value    : System.Address;
          Encoding : Atoms.Atom;
          Format   : Property_Format_Type;
          items   : X_Integer;
```

```
   end record;
procedure X_String_List_To_Text_Property(
        Strings         : in String_List ;
        Text_Prop       : in out Text_Property_Record ;
        Status          : out Boolean  );

procedure X_Text_Property_To_String_List(
        Text_Prop       : in Text_Property_Record ;
        Strings         : in out String_List ;
        Status          : out Boolean  );

procedure X_Free_String_List       (
        Strings         : in out String_List  );

procedure X_Set_Text_Property      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in Text_Property_Record ;
        Property        : in Atoms.Atom  );

procedure X_Get_Text_Property      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in out Text_Property_Record ;
        Property        : in Atoms.Atom ;
        Status          : out Boolean  );

procedure X_Set_Standard_Properties(
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        W_Name          : in String ;
        Icon_Name       : in String ;
        Icon            : in Pixmap ;
        Command         : in String_List ;
        Hints           : in Size_Hint_Record  );

procedure X_Store_Name             (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Name            : in String  );

procedure X_Fetch_Name             (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Name            : in out String_Pointer ;
        Status          : out Boolean  );

procedure X_Set_Icon_Name          (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Name            : in String  );

procedure X_Get_Icon_Name          (
        Display_Id      : in Display ;
```

```
        Window_Id       : in Window ;
        Icon_Name       : in out String_Pointer ;
        Status          : out Boolean  );

procedure X_Set_Wm_Name           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in Text_Property_Record  );

procedure X_Get_Wm_Name           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in out Text_Property_Record ;
        Status          : out Boolean  );

procedure X_Set_Wm_Icon_Name      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in Text_Property_Record  );

procedure X_Get_Wm_Icon_Name      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in out Text_Property_Record ;
        Status          : out Boolean  );

procedure X_Set_Command           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Command         : in String_List  );

procedure X_Get_Command           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Command         : in out String_List ;
        Status          : out Boolean  );

function  X_Alloc_Wm_Hints
    return Wm_Hint_Record;

procedure X_Set_Wm_Hints          (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in Wm_Hint_Record  );

function  X_Get_Wm_Hints          (
        Display_Id      : in Display ;
        Window_Id       : in Window  )
    return Wm_Hint_Record;

function  X_Alloc_Size_Hints
    return Size_Hint_Record;

procedure X_Set_Normal_Hints      (
```

```
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in Size_Hint_Record  );

procedure X_Get_Normal_Hints       (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in out Size_Hint_Record ;
        Hints_Found     : out Boolean  );

procedure X_Set_Wm_Normal_Hints    (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in Size_Hint_Record  );

procedure X_Get_Wm_Normal_Hints    (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in out Size_Hint_Record ;
        Supplied_Mask   : in out Size_Hint_Mask_Type ;
        Status          : out Boolean  );

procedure X_Set_Zoom_Hints         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in Size_Hint_Record  );

procedure X_Get_Zoom_Hints         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in out Size_Hint_Record ;
        Hints_Found     : out Boolean  );

procedure X_Set_Size_Hints         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Hints           : in Size_Hint_Record ;
        Property        : in Atoms.Atom  );

procedure X_Get_Size_Hints         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Property        : in Atoms.Atom ;
        Hints           : in out Size_Hint_Record ;
        Hints_Found     : out Boolean  );

function  X_Alloc_Icon_Size
    return Icon_Size_Record;

procedure X_Set_Icon_Sizes         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Size_List       : in Icon_Size_List  );
```

```
    procedure X_Get_Icon_Sizes         (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            size_list       : in out Icon_Size_List ;
            status          : in out boolean  );

    function  X_Alloc_Class_Hint
        return Wm_Class_Hint_Record;

    procedure X_Set_Class_Hint         (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Class_Hints     : in Wm_Class_Hint_Record  );

    procedure X_Get_Class_Hint         (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Class_Hints_Return : in out Wm_Class_Hint_Record ;
            Success         : out Boolean  );

    procedure X_Set_Transient_For_Hint (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Prop_Window     : in Window  );

    procedure X_Get_Transient_For_Hint (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Prop_Window_Return : in out Window ;
            Success         : out Boolean  );

    function  X_Set_Wm_Protocols       (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Protocols       : in Atoms.Atom_List  )
        return Boolean;

    procedure X_Get_Wm_Protocols       (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Protocols       : in out Atoms.Atom_List ;
            Status          : out Boolean  );

    function  X_Set_Wm_Colormap_Windows(
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Colormap_Windows : in Window_List  )
        return Boolean;

    procedure X_Get_Wm_Colormap_Windows(
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Colormap_Windows : in out Window_List ;
            Status          : out Boolean  );
```

```
procedure X_Set_Wm_Client_Machine  (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in Text_Property_Record  );

procedure X_Get_Wm_Client_Machine  (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Text_Prop       : in out Text_Property_Record ;
        Status          : out Boolean  );

procedure X_Set_Wm_Properties      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Window_Name     : in Text_Property_Record ;
        Icon_Name       : in Text_Property_Record ;
        Argv            : in String_List ;
        Argc            : in X_Long_Integer ;
        Size_Hints      : in Size_Hint_Record ;
        Wm_Hints        : in Wm_Hint_Record ;
        Class_Hints     : in Wm_Class_Hint_Record  );

procedure X_Wm_Geometry            (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number ;
        User_Geometry   : in String ;
        Default_Geometry : in String ;
        Border_Width    : in Width_Height ;
        Size_Hints      : in Size_Hint_Record ;
        X               : out Pixel ;
        Y               : out Pixel ;
        Width           : out Width_Height ;
        Height          : out Width_Height ;
        Gravity         : out Gravity_Type  );
--* This function syntax has been changed. The original syntax
--* was erroneous. The second parameter has been correctly changed
--* to screen number.

function  X_Create_Pixmap          (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Depth           : in Depth_Type  )
    return Pixmap;

procedure X_Free_Pixmap            (
        Display_Id      : in Display ;
        Pixmap_Id       : in out Pixmap  );

function  X_Create_Window          (
        Display_Id      : in Display ;
        Parent          : in Window ;
```

```
            X               : in Coordinate ;
            Y               : in Coordinate ;
            Width           : in Width_Height ;
            Height          : in Width_Height ;
            Border_Width    : in Width_Height ;
            Depth           : in Depth_Type ;
            Class           : in Window_Class ;
            Visuals         : in Visual ;
            Value_Mask      : in Wa_Mask_Type ;
            Attributes      : in Set_Window_Attributes_Record  )
        return Window;

    function  X_Create_Simple_Window   (
            Display_Id      : in Display ;
            Parent          : in Window ;
            X               : in Coordinate ;
            Y               : in Coordinate ;
            Width           : in Width_Height ;
            Height          : in Width_Height ;
            Border_Width    : in Width_Height ;
            Border          : in Color_Value ;
            Background      : in Color_Value  )
        return Window;

    procedure X_Destroy_Window         (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Destroy_Subwindows     (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Map_Window             (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Map_Raised             (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Map_Subwindows         (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Unmap_Window           (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Unmap_Subwindows       (
            Display_Id      : in Display ;
            Window_Id       : in Window  );

    procedure X_Configure_Window       (
            Display_Id      : in Display ;
```

```
        Window_Id       : in Window ;
        Value_Mask      : in Cw_Mask_Type ;
        Changes         : in Window_Changes_Record  );

function  X_Reconfigure_Wm_Window  (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Screen_No       : in Screen_Number ;
        Value_Mask      : in Cw_Mask_Type ;
        Changes         : in Window_Changes_Record  )
    return Boolean;

procedure X_Move_Window           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        X               : in Coordinate ;
        Y               : in Coordinate  );

procedure X_Resize_Window         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Width           : in Width_Height ;
        Height          : in Width_Height  );

procedure X_Move_Resize_Window    (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height  );

function  X_Iconify_Window        (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Screen_No       : in Screen_Number  )
    return Boolean;

function  X_Withdraw_Window       (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Screen_No       : in Screen_Number  )
    return Boolean;

procedure X_Set_Window_Border_Width(
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Width           : in Width_Height  );

procedure X_Raise_Window          (
        Display_Id      : in Display ;
        Window_Id       : in Window  );

procedure X_Lower_Window          (
```

```
        Display_Id      : in Display ;
        Window_Id       : in Window  );

    procedure X_Circulate_Subwindows   (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Direction       : in Direction_Type  );

    procedure X_Circulate_Subwindows_Up(
        Display_Id      : in Display ;
        Window_Id       : in Window  );

    procedure X_Circulate_Subwindows_Down(
        Display_Id      : in Display ;
        Window_Id       : in Window  );

    procedure X_Restack_Windows        (
        Display_Id      : in Display ;
        Windows         : in Window_List  );

    procedure X_Change_Window_Attributes(
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Value_Mask      : in Wa_Mask_Type ;
        Attributes      : in Set_Window_Attributes_Record  );

    procedure X_Set_Window_Background  (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Background      : in Color_Value  );

    procedure X_Set_Window_Background_Pixmap(
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Tile            : in Pixmap  );

    procedure X_Set_Window_Border      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Border          : in Color_Value  );

    procedure X_Set_Window_Border_Pixmap(
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Tile            : in Pixmap  );

    procedure X_Translate_Coordinates  (
        Display_Id      : in Display ;
        Source          : in Window ;
        Destination     : in Window ;
        Source_X        : in Coordinate ;
        Source_Y        : in Coordinate ;
        Destination_X   : out Coordinate ;
        Destination_Y   : out Coordinate ;
```

```
        Child           : out Window ;
        Same_Screen     : out Boolean  );

procedure X_Query_Tree            (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Root            : in out Window ;
        Parent          : in out Window ;
        Children        : in out Window_List ;
        Status          : out Boolean  );

procedure X_Get_Window_Attributes  (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Attributes      : in out Window_Attributes_Record ;
        Status          : out Boolean  );

procedure X_Get_Geometry          (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Root            : out Window ;
        X               : out Coordinate ;
        Y               : out Coordinate ;
        Width           : out Width_Height ;
        Height          : out Width_Height ;
        Border_Width    : out Width_Height ;
        Depths          : out Depth_Type ;
        Status          : out Boolean  );

procedure X_Parse_Geometry        (
        Parse_String    : in String ;
        X               : out Coordinate ;
        Y               : out Coordinate ;
        Width           : out Width_Height ;
        Height          : out Width_Height ;
        Values_Found    : out Geometry_Mask_Type  );

procedure X_Geometry              (
        Display_Id      : in Display ;
        Screen_No       : in Screen_Number ;
        Position        : in String ;
        Default         : in String ;
        Border_Width    : in Width_Height ;
        Font_Height     : in Width_Height ;
        Font_Width      : in Width_Height ;
        X_Padding       : in Coordinate ;
        Y_Padding       : in Coordinate ;
        X               : out Coordinate ;
        Y               : out Coordinate ;
        Width           : out Coordinate ;
        Height          : out Coordinate ;
        Values_Found    : out Geometry_Mask_Type  );

procedure X_Query_Pointer         (
```

```
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Root            : out Window ;
            Child           : out Window ;
            Root_X          : out Coordinate ;
            Root_Y          : out Coordinate ;
            Window_X        : out Coordinate ;
            Window_Y        : out Coordinate ;
            Keys_And_Buttons : out Events.Key_And_Button_Mask ;
            Same_Screen     : out Boolean  );

    procedure X_Get_Window_Property    (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Property        : in Atoms.Atom ;
            Long_Offset     : in X_Long_Integer ;
            Long_Length     : in X_Long_Integer ;
            Delete          : in Boolean ;
            Req_Type        : in Atoms.Atom ;
            Actual_Type     : out Atoms.Atom ;
            Actual_Form     : out X_Long_Integer ;
            Bytes_After     : out X_Long_Integer ;
            Data            : out Bytes  );

    function  X_List_Properties        (
            Display_Id      : in Display ;
            Window_Id       : in Window  )
        return Atoms.Atom_List;

    procedure X_Change_Property        (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Property        : in Atoms.Atom ;
            Kind            : in Atoms.Atom ;
            Format          : in Property_Format_Type ;
            Mode            : in Property_Mode ;
            Data            : in Bytes ;
            N_Items         : in Natural  );

    procedure X_Rotate_Window_Properties(
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Properties      : in Atoms.Atom_List ;
            N_Positions     : in Natural  );

    procedure X_Delete_Property        (
            Display_Id      : in Display ;
            Window_Id       : in Window ;
            Property        : in Atoms.Atom  );

    procedure X_Set_Selection_Owner    (
            Display_Id      : in Display ;
            Selection       : in Atoms.Atom ;
            Owner           : in Window ;
```

```
        Time_Stamp      : in Time  );

function  X_Get_Selection_Owner    (
        Display_Id      : in Display ;
        Selection       : in Atoms.Atom  )
    return Window;

procedure X_Convert_Selection      (
        Display_Id      : in Display ;
        Selection       : in Atoms.Atom ;
        Target          : in Atoms.Atom ;
        Property        : in Atoms.Atom ;
        Requestor       : in Window ;
        Time_Stamp      : in Time  );

procedure X_Reparent_Window        (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Parent          : in Window ;
        X               : in Coordinate ;
        Y               : in Coordinate  );

procedure X_Change_Save_Set        (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Mode            : in Set_Mode  );

procedure X_Add_To_Save_Set        (
        Display_Id      : in Display ;
        Window_Id       : in Window  );

procedure X_Remove_From_Save_Set   (
        Display_Id      : in Display ;
        Window_Id       : in Window  );

procedure X_Save_Context           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        The_Context     : in Context ;
        Data            : in System.Address ;
        Status          : out Context_Status_Type  );

procedure X_Find_Context           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        The_Context     : in Context ;
        Data            : in out System.Address ;
        Status          : out Context_Status_Type  );

procedure X_Delete_Context         (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        The_Context     : in Context ;
        Status          : out Context_Status_Type  );
```

```
function  X_Unique_Context
    return Context;

procedure X_Synchronize            (
        Display_Id      : in Display ;
        Mode            : in Synchronize_Mode := On );

function Create_Constant          (
        Value           : in X_Long_Integer  )
    return Pixmap_Format;

function Create_Constant          (
        Value           : in X_Long_Integer  )
    return Order_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Backing_Store_Type;

function Create_Constant          (
        Value           : in X_Long_Integer  )
    return Property_Format_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Visual_Class_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Window_Class;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Direction_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Property_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Set_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Gravity_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Map_State_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
```

```
        return Map_Status;

    function  Create_Constant          (
            Value           : in X_Long_Integer  )
        return Initial_State_Type;

    function  Create_Constant          (
            Value           : in X_Long_Integer  )
        return Context_Status_Type;
```

Package X_Lib.Atoms

```
        function  X_Intern_Atom            (
                Display_Id      : in Display ;
                Atom_Name       : in String ;
                Only_If_Exists  : in Boolean := True )
            return Atom;

        function  X_Get_Atom_Name          (
                Display_Id      : in Display ;
                Atom_Id         : in Atom  )
            return String;
```

Package X_Lib.Fonts

```
        function  X_Load_Font              (
                Display_Id      : in Display ;
                Font_Name       : in String  )
            return Font;

        function  X_Query_Font             (
                Display_Id      : in Display ;
                Font_Id         : in Font  )
            return Font_Record_Pointer;
        --* This function syntax has been changed. We felt that the
        --* original STARS syntax would lead to code that would be
        --* subject to possible memory leaks. The current implementation
        --* returns an access value rather than the record itself.

        procedure X_List_Fonts_With_Info   (
                Display_Id      : in Display ;
                Pattern         : in String ;
                Font_Info_List  : in out Font_Record_List ;
                Names_List      : in out String_List  );

        procedure X_Free_Font_Info         (
                Names_List      : in out String_List ;
                Font_Info_List  : in out Font_Record_List  );

        function  X_Load_Query_Font        (
                Display_Id      : in Display ;
                Font_Name       : in String  )
            return Font_Record_Pointer;
        --* This function syntax has been changed. We felt that the
        --* original STARS syntax would lead to code that would be
```

```
--* subject to possible memory leaks. The current implementation
--* returns an access value rather than the record itself.

procedure X_Free_Font              (
        Display_Id      : in Display ;
        Font_Info       : in out Font_Record_Pointer  );
--* This function syntax has been changed. We felt that the
--* original STARS syntax would lead to code that would be
--* subject to possible memory leaks. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

procedure X_Get_Font_Property      (
        Font_Info       : in Font_Record_Pointer ;
        Property        : in Atoms.Atom ;
        Value           : out X_Long_Integer ;
        Defined         : out Boolean  );
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

procedure X_Unload_Font            (
        Display_Id      : in Display ;
        Font_Id         : in Font  );

procedure X_List_Fonts             (
        Display_Id      : in Display ;
        Pattern         : in String ;
        Names           : in out String_List  );

procedure X_Free_Font_Names        (
        Display_Id      : in Display ;
        Directories     : in out String_List  );

procedure X_Set_Font_Path          (
        Display_Id      : in Display ;
        Directories     : in String_List  );

function  X_Get_Font_Path          (
        Display_Id      : in Display  )
    return String_List;

procedure X_Free_Font_Path         (
        Directories     : in out String_List  );

function  X_Text_Width             (
        Font_Info       : in Font_Record_Pointer ;
        Text            : in String  )
    return Pixel;
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

function  X_Text_Width             (
```

```
        Font_Info       : in Font_Record_Pointer ;
        Text            : in String_8  )
    return Pixel;
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

function  X_Text_Width_16          (
        Font_Info       : in Font_Record_Pointer ;
        Text            : in String_16  )
    return Pixel;
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

procedure X_Text_Extents           (
        Font_Info       : in Font_Record_Pointer ;
        Text            : in String ;
        Direction       : out Font_Direction ;
        Ascent          : out Pixel ;
        Descent         : out Pixel ;
        Overall         : out Character_Record  );
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

procedure X_Text_Extents           (
        Font_Info       : in Font_Record_Pointer ;
        Text            : in String_8 ;
        Direction       : out Font_Direction ;
        Ascent          : out Pixel ;
        Descent         : out Pixel ;
        Overall         : out Character_Record  );
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

procedure X_Text_Extents_16        (
        Font_Info       : in Font_Record_Pointer ;
        Text            : in String_16 ;
        Direction       : out Font_Direction ;
        Ascent          : out Pixel ;
        Descent         : out Pixel ;
        Overall         : out Character_Record  );
--* This function syntax has been changed. The current implementation
--* expects an access value as a parameter rather than the
--* record itself.

procedure X_Query_Text_Extents     (
        Display_Id      : in Display ;
        Font_Id         : in Font ;
        Text            : in String ;
        Direction       : out Font_Direction ;
        Ascent          : out Pixel ;
```

```
        Descent         : out Pixel ;
        Overall         : out Character_Record  );

    procedure X_Query_Text_Extents     (
        Display_Id      : in Display ;
        Font_Id         : in Font ;
        Text            : in String_8 ;
        Direction       : out Font_Direction ;
        Ascent          : out Pixel ;
        Descent         : out Pixel ;
        Overall         : out Character_Record  );

    procedure X_Query_Text_Extents_16  (
        Display_Id      : in Display ;
        Font_Id         : in Font ;
        Text            : in String_16 ;
        Direction       : out Font_Direction ;
        Ascent          : out Pixel ;
        Descent         : out Pixel ;
        Overall         : out Character_Record  );

    function  Create_Constant          (
        Value           : in X_Long_Integer  )
      return Font_Direction;
```

Package X_Lib.Colors

```
    function  X_Create_Colormap        (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Visual_Id       : in Visual ;
        Allocate        : in Color_Map_Allocator  )
      return Color_Map;

    procedure X_Copy_Colormap_And_Free (
        Display_Id      : in Display ;
        Source          : in out Color_Map ;
        Target          : out Color_Map  );

    procedure X_Free_Colormap          (
        Display_Id      : in Display ;
        Map_Id          : in out Color_Map  );

    procedure X_Set_Window_Colormap    (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Map_Id          : in Color_Map  );

    procedure X_Set_Rgb_Colormaps      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Standard_Colormaps : in Standard_Colormap_List ;
        Property        : in Atoms.Atom  );
```

```
procedure X_Get_Rgb_Colormaps      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Standard_Colormaps : in out Standard_Colormap_List ;
        Property        : in Atoms.Atom ;
        Status          : out Boolean  );

procedure X_Allocate_Color         (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Definition      : in out Color_Record ;
        Success         : out Boolean  );

procedure X_Allocate_Named_Color   (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Color_Name      : in String ;
        Screen_Def      : out Color_Record ;
        Exact_Def       : out Color_Record ;
        Success         : out Boolean  );

procedure X_Lookup_Color           (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Color_Name      : in String ;
        Screen_Def      : out Color_Record ;
        Exact_Def       : out Color_Record ;
        Success         : out Boolean  );

procedure X_Store_Colors           (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Colors          : in out Color_Array  );

procedure X_Store_Color            (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Screen_Def      : in out Color_Record  );

procedure X_Allocate_Color_Cells   (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Contiguous      : in Boolean ;
        Planes          : in out Plane_Mask_Array ;
        Pixel_Values    : in out Pixel_Array ;
        Success         : out Boolean  );

procedure X_Allocate_Color_Planes  (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Contiguous      : in Boolean ;
        Red_Shades      : in Natural ;
        Green_Shades    : in Natural ;
        Blue_Shades     : in Natural ;
```

```
        Pixel_Values    : in out Pixel_List ;
        Red_Mask        : out Plane_Mask ;
        Green_Mask      : out Plane_Mask ;
        Blue_Mask       : out Plane_Mask ;
        Success         : out Boolean  );

procedure X_Store_Named_Color      (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Color_Name      : in String ;
        Map_Entry       : in Color_Value ;
        Flags           : in Color_Flag  );

procedure X_Free_Colors            (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Pixels_To_Free  : in Pixel_Array ;
        Planes          : in Plane_Mask  );

procedure X_Query_Color            (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        The_Color       : in out Color_Record  );

procedure X_Query_Colors           (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        The_Colors      : in out Color_Array  );

procedure X_Install_Colormap       (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map  );

procedure X_Uninstall_Colormap     (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map  );

function  X_List_Installed_Colormaps(
        Display_Id      : in Display ;
        Window_Id       : in Window  )
    return Color_Map_List;

procedure X_Parse_Color            (
        Display_Id      : in Display ;
        Map_Id          : in Color_Map ;
        Color_Name      : in String ;
        Screen_Def      : out Color_Record ;
        Success         : out Boolean  );

function  X_Alloc_Standard_Colormap
    return Standard_Colormap_Record;

procedure X_Get_Standard_Colormap  (
        Display_Id      : in Display ;
```

```
        Window_Id       : in Window ;
        Property        : in Atoms.Atom ;
        Cmap_Return     : out Standard_Colormap_Record ;
        Success         : out Boolean  );

procedure X_Set_Standard_Colormap  (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Cmap            : in Standard_Colormap_Record ;
        Property        : in Atoms.Atom  );

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Color_Map_Allocator;
```

Package X_Lib.Graphic_Output

```
procedure X_Get_Gc_Values          (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Value_Mask      : in Gc_Mask_Type ;
        Values          : in out Gc_Value_Record ;
        Status          : out Boolean  );

function  X_Create_Gc              (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Value_Mask      : in Gc_Mask_Type ;
        Values          : in Gc_Value_Record  )
    return Graphic_Context;

procedure X_Copy_Gc                (
        Display_Id      : in Display ;
        Value_Mask      : in Gc_Mask_Type ;
        Source          : in Graphic_Context ;
        Destination     : in Graphic_Context  );

procedure X_Change_Gc              (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Value_Mask      : in Gc_Mask_Type ;
        Values          : in Gc_Value_Record  );

procedure X_Free_Gc                (
        Display_Id      : in Display ;
        Gc              : in out Graphic_Context  );

procedure X_Set_State              (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Foreground      : in Color_Value ;
        Background      : in Color_Value ;
        Gx_Function     : in Gx_Function_Code ;
        Plane           : in Plane_Mask  );
```

```
procedure X_Set_Function           (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Gx_Function     : in Gx_Function_Code  );

procedure X_Set_Plane_Mask         (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Plane           : in Plane_Mask  );

procedure X_Set_Foreground         (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Foreground      : in Color_Value  );

procedure X_Set_Background         (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Background      : in Color_Value  );

procedure X_Set_Line_Attributes    (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Line_Width      : in Width_Height ;
        Line_Style      : in Line_Style_Type ;
        Cap_Style       : in Cap_Style_Type ;
        Join_Style      : in Join_Style_Type  );

procedure X_Set_Dashes             (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Dash_Offset     : in Pixel ;
        Dash_List       : in Bits  );

procedure X_Set_Fill_Style         (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Fill_Style      : in Fill_Style_Type  );

procedure X_Set_Fill_Rule          (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Fill_Rule       : in Fill_Rule_Type  );

procedure X_Query_Best_Size        (
        Display_Id      : in Display ;
        Shape_Class     : in Shape_Class_Type ;
        Which_Screen    : in Drawable ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Ret_Width       : out Width_Height ;
        Ret_Height      : out Width_Height ;
        Status          : out Boolean  );
```

```
procedure X_Query_Best_Tile        (
        Display_Id      : in Display ;
        Which_Screen    : in Drawable ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Ret_Width       : out Width_Height ;
        Ret_Height      : out Width_Height ;
        Status          : out Boolean  );

procedure X_Query_Best_Stipple     (
        Display_Id      : in Display ;
        Which_Screen    : in Drawable ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Ret_Width       : out Width_Height ;
        Ret_Height      : out Width_Height ;
        Status          : out Boolean  );

procedure X_Set_Tile               (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Tile            : in Pixmap  );

procedure X_Set_Stipple            (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Stipple         : in Pixmap  );

procedure X_Set_Ts_Origin          (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Ts_X_Origin     : in Coordinate ;
        Ts_Y_Origin     : in Coordinate  );

procedure X_Set_Font               (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Font_Id         : in Fonts.Font  );

procedure X_Set_Clip_Origin        (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Clip_X_Origin   : in Coordinate ;
        Clip_Y_Origin   : in Coordinate  );

procedure X_Set_Clip_Mask          (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Clip_Mask       : in Pixmap  );

procedure X_Set_Clip_Rectangles    (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
```

```
        Clip_X_Origin   : in Coordinate ;
        Clip_Y_Origin   : in Coordinate ;
        Rectangles      : in Rectangle_Array ;
        Ordering        : in Ordering_Type  );

procedure X_Set_Arc_Mode          (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Arc_Mode        : in Arc_Mode_Type  );

procedure X_Set_Subwindow_Mode    (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Subwindow_Mode  : in Subwindow_Mode_Type  );

procedure X_Set_Graphics_Exposures (
        Display_Id      : in Display ;
        Gc              : in Graphic_Context ;
        Graphics_Exposures : in Boolean  );

procedure X_Clear_Area            (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Exposures       : in Boolean  );

procedure X_Clear_Window          (
        Display_Id      : in Display ;
        Window_Id       : in Window  );

procedure X_Copy_Area             (
        Display_Id      : in Display ;
        Source          : in Drawable ;
        Destination     : in Drawable ;
        Gc              : in Graphic_Context ;
        Source_X        : in Coordinate ;
        Source_Y        : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Destination_X   : in Coordinate ;
        Destination_Y   : in Coordinate  );

procedure X_Copy_Plane            (
        Display_Id      : in Display ;
        Source          : in Drawable ;
        Destination     : in Drawable ;
        Gc              : in Graphic_Context ;
        Source_X        : in Coordinate ;
        Source_Y        : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
```

```
        Destination_X   : in Coordinate ;
        Destination_Y   : in Coordinate ;
        Plane           : in Plane_Mask  );

procedure X_Draw_Point            (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate  );

procedure X_Draw_Points           (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Points          : in Point_Array ;
        Coordinate_Mode : in Coordinate_Mode_Type  );

procedure X_Draw_Line             (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X_1             : in Coordinate ;
        Y_1             : in Coordinate ;
        X_2             : in Coordinate ;
        Y_2             : in Coordinate  );

procedure X_Draw_Lines            (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Points          : in Point_Array ;
        Coordinate_Mode : in Coordinate_Mode_Type  );

procedure X_Draw_Segments         (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Segments        : in Segment_Array  );

procedure X_Draw_Rectangle        (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height  );

procedure X_Draw_Rectangles       (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Bounds          : in Rectangle_Array  );
```

```
procedure X_Draw_Arc              (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : Coordinate  ;
        Y               : Coordinate  ;
        Width           : Width_Height  ;
        Height          : Width_Height  ;
        Angle_1         : Angle  ;
        Angle_2         : Angle   );

procedure X_Draw_Arcs             (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Bounds          : in Arc_Array  );

procedure X_Fill_Rectangle        (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height  );

procedure X_Fill_Rectangles       (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Rectangles      : in Rectangle_Array  );

procedure X_Fill_Polygon          (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        Points          : in Point_Array ;
        Shape           : in Shape_Type ;
        Coordinate_Mode : in Coordinate_Mode_Type  );

procedure X_Fill_Arc              (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Angle_1         : in Angle ;
        Angle_2         : in Angle  );

procedure X_Fill_Arcs             (
        Display_Id      : in Display ;
```

```
                Drawable_Id     : in Drawable ;
                Gc              : in Graphic_Context ;
                Arcs            : in Arc_Array  );

        procedure X_Draw_Text             (
                Display_Id      : in Display ;
                Drawable_Id     : in Drawable ;
                Gc              : in Graphic_Context ;
                X               : in Coordinate ;
                Y               : in Coordinate ;
                Text_Items      : in Text_Item_Array  );

--      procedure X_Draw_Text             (
--              Display_Id      : in Display ;
--              Drawable_Id     : in Drawable ;
--              Gc              : in Graphic_Context ;
--              X               : in Coordinate ;
--              Y               : in Coordinate ;
--              Text_Items      : in Text_Item_8_Array  );
--      --* This function has been deleted in this release because
--      --* we felt that its implementation would be very prone to
--      --* memory leaks.

        procedure X_Draw_Text_16          (
                Display_Id      : in Display ;
                Drawable_Id     : in Drawable ;
                Gc              : in Graphic_Context ;
                X               : in Coordinate ;
                Y               : in Coordinate ;
                Text_Items      : in Text_Item_16_Array  );

        procedure X_Draw_String           (
                Display_Id      : in Display ;
                Drawable_Id     : in Drawable ;
                Gc              : in Graphic_Context ;
                X               : in Coordinate ;
                Y               : in Coordinate ;
                Text            : in String  );

        procedure X_Draw_String           (
                Display_Id      : in Display ;
                Drawable_Id     : in Drawable ;
                Gc              : in Graphic_Context ;
                X               : in Coordinate ;
                Y               : in Coordinate ;
                Text            : in String_8  );

        procedure X_Draw_String_16        (
                Display_Id      : in Display ;
                Drawable_Id     : in Drawable ;
                Gc              : in Graphic_Context ;
                X               : in Coordinate ;
                Y               : in Coordinate ;
                Text            : in Fonts.String_16  );
```

```
procedure X_Draw_Image_String      (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Text            : in String  );

procedure X_Draw_Image_String      (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Text            : in String_8  );

procedure X_Draw_Image_String_16   (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Text            : in Fonts.String_16  );

procedure Set_Image                (
        X_Image         : in out Image ;
        Values          : in Set_Image_Record  );

procedure X_Put_Image              (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Gc              : in Graphic_Context ;
        X_Image         : in Image ;
        Source_X        : in Coordinate ;
        Source_Y        : in Coordinate ;
        Destination_X   : in Coordinate ;
        Destination_Y   : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height  );

function  X_Get_Image              (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Planes          : in Plane_Mask ;
        Format          : in Pixmap_Format  )
    return Image;

function  X_Get_Subimage           (
        Display_Id      : in Display ;
```

```
        Drawable_Id     : in Drawable ;
        Source_X        : in Coordinate ;
        Source_Y        : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Planes          : in Plane_Mask ;
        Format          : in Pixmap_Format ;
        Destination_X   : in Coordinate ;
        Destination_Y   : in Coordinate ;
        Destination     : in Image  )
    return Image;

function  X_Create_Image          (
        Display_Id      : in Display ;
        Visual_Id       : in Visual ;
        Depth           : in Depth_Type ;
        Format          : in Pixmap_Format ;
        Offset          : in Pixel ;
        Data            : in Bits ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Scanline_Pad    : in Pixel ;
        Scanline_Length : in Pixel  )
    return Image;

function  X_Get_Pixel             (
        X_Image         : in Image ;
        X               : in Coordinate ;
        Y               : in Coordinate  )
    return Color_Value;

function  X_Put_Pixel             (
        X_Image         : in Image ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Value           : in Color_Value  )
    return Boolean;

function  X_Sub_Image             (
        X_Image         : in Image ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Subimage_Width  : in Width_Height ;
        Subimage_Height : in Width_Height  )
    return Image;

function  X_Add_Pixel             (
        X_Image         : in Image ;
        Value           : in Color_Value  )
    return Boolean;

procedure X_Destroy_Image         (
        X_Image         : in out Image ;
        Status          : out Boolean  );
```

```
procedure X_Read_Bitmap_File       (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Filename        : in String ;
        Width           : out Width_Height ;
        Height          : out Width_Height ;
        Bitmap          : out Pixmap ;
        X_Hot           : out Coordinate ;
        Y_Hot           : out Coordinate ;
        Status          : out Bitmap_Status_Type  );

function  X_Write_Bitmap_File      (
        Display_Id      : in Display ;
        Filename        : in String ;
        Bitmap          : in Pixmap ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        X_Hot           : in Coordinate ;
        Y_Hot           : in Coordinate  )
    return Bitmap_Status_Type;

function  X_Create_Bitmap_From_Data(
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Data            : in Bits ;
        Width           : in Width_Height ;
        Height          : in Width_Height  )
    return Pixmap;

function  X_Create_Pixmap_From_Bitmap_Data(
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Data            : in Bits ;
        Width           : in Width_Height ;
        Height          : in Width_Height ;
        Foreground      : in Color_Value ;
        Background      : in Color_Value ;
        Depth           : in Depth_Type  )
    return Pixmap;

procedure X_List_Pixmap_Formats    (
        Display_Id      : in Display ;
        Formats         : in out Pixmap_Format_Value_List  );

function  X_Resource_Gc_From_Graphic_Context(
        Gc              : in Graphic_Context  )
    return Resource_Gc;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Bitmap_Status_Type;

function  Create_Constant          (
```

```
        Value           : in X_Long_Integer  )
    return Gx_Function_Code;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Line_Style_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Cap_Style_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Join_Style_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Fill_Style_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Fill_Rule_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Shape_Class_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Ordering_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Arc_Mode_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Subwindow_Mode_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Coordinate_Mode_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Shape_Type;
```

Package X_Lib.Cursors

```
function  X_Create_Font_Cursor     (
        Display_Id      : in Display ;
        Shape           : in Cursor_Shape  )
    return Cursor;
```

```
function  X_Create_Pixmap_Cursor   (
        Display_Id      : in Display ;
        Source          : in Pixmap ;
        Mask            : in Pixmap ;
        Foreground      : in Colors.Color_Record ;
        Background      : in Colors.Color_Record ;
        X_Hot           : in Coordinate ;
        Y_Hot           : in Coordinate  )
    return Cursor;

function  X_Create_Glyph_Cursor    (
        Display_Id      : in Display ;
        Source_Font     : in Fonts.Font ;
        Mask_Font       : in Fonts.Font ;
        Source_Glyph    : in X_Character ;
        Mask_Glyph      : in X_Character ;
        Foreground      : in Colors.Color_Record ;
        Background      : in Colors.Color_Record  )
    return Cursor;

procedure X_Recolor_Cursor         (
        Display_Id      : in Display ;
        Cursor_Id       : in Cursor ;
        Foreground      : in Colors.Color_Record ;
        Background      : in Colors.Color_Record  );

procedure X_Free_Cursor            (
        Display_Id      : in Display ;
        Cursor_Id       : in out Cursor  );

procedure X_Query_Best_Cursor      (
        Display_Id      : in Display ;
        Drawable_Id     : in Drawable ;
        Width           : in out Width_Height ;
        Height          : in out Width_Height  );

procedure X_Define_Cursor          (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Cursor_Id       : in Cursor  );

procedure X_Undefine_Cursor        (
        Display_Id      : in Display ;
        Window_Id       : in Window  );
```

Package X_Lib.Cut_And_Paste

```
procedure X_Store_Buffer           (
        Display_Id      : in Display ;
        Data            : in Bytes ;
        Number_Of_Bytes : in Positive ;
        To_Buffer       : in Natural := 0 );

function  X_Fetch_Buffer           (
```

```
        Display_Id      : in Display ;
        From_Buffer     : in Natural := 0 )
    return Bytes;

procedure X_Rotate_Buffers         (
        Display_Id      : in Display ;
        By              : in Integer  );
```

Package X_Lib.Regions

```
function  X_Polygon_Region         (
        Points          : in Point_Array ;
        Rule            : in Graphic_Output.Fill_Rule_Type  )
    return Region;

procedure X_Clip_Box               (
        Region_Id       : in Region ;
        Bounds          : in out Rectangle  );

function  X_Create_Region
    return Region;

procedure X_Set_Region             (
        Display_Id      : in Display ;
        Gc              : in Graphic_Output.Graphic_Context ;
        Region_Id       : in Region  );

procedure X_Destroy_Region         (
        Region_Id       : in out Region  );

procedure X_Offset_Region          (
        Region_Id       : in Region ;
        Delta_X         : in Coordinate ;
        Delta_Y         : in Coordinate  );

procedure X_Shrink_Region          (
        Region_Id       : in Region ;
        Delta_X         : in Coordinate ;
        Delta_Y         : in Coordinate  );

procedure X_Intersect_Region       (
        Source_A        : in Region ;
        Source_B        : in Region ;
        Destination     : in Region  );

procedure X_Union_Region           (
        Source_A        : in Region ;
        Source_B        : in Region ;
        Destination     : in Region  );

procedure X_Union_Rect_With_Region (
        Source_A        : in Rectangle ;
        Source_B        : in Region ;
```

```
        Destination     : in Region  );

    procedure X_Subtract_Region        (
        Source_A        : in Region ;
        Source_B        : in Region ;
        Destination     : in Region  );

    procedure X_Xor_Region             (
        Source_A        : in Region ;
        Source_B        : in Region ;
        Destination     : in Region  );

    function  X_Empty_Region           (
        Region_Id       : in Region  )
      return Boolean;

    function  X_Equal_Region           (
        Region_1        : in Region ;
        Region_2        : in Region  )
      return Boolean;

    function  X_Point_In_Region        (
        Region_Id       : in Region ;
        X               : in Coordinate ;
        Y               : in Coordinate  )
      return Boolean;

    function  X_Rect_In_Region         (
        Region_Id       : in Region ;
        X               : in Coordinate ;
        Y               : in Coordinate ;
        Width           : in Width_Height ;
        Height          : in Width_Height  )
      return Region_Results;

    function  Create_Constant          (
        Value           : in X_Long_Integer  )
      return Region_Results;
      return Region_Results;
```

Package X_Lib.Keyboard

```
    procedure X_Rebind_Keysym          (
        Display_Id      : in Display ;
        The_Symbol      : in Key_Sym ;
        Modifiers       : in Key_Sym_List ;
        Text            : in String  );

    function  X_String_To_Keysym       (
        Name            : in String  )
      return Key_Sym;

    function  X_Keysym_To_String       (
        The_Symbol      : in Key_Sym  )
```

```
    return String;

procedure X_Keycode_To_Keysym      (
        Display_Id      : in Display ;
        Key_Code        : in Keycode ;
        Index           : in Keycode ;
        The_Symbol      : out Key_Sym  );

function  X_Keysym_To_Keycode      (
        Display_Id      : in Display ;
        The_Symbol      : in Key_Sym  )
    return Keycode;

procedure X_Change_Keyboard_Control(
        Display_Id      : in Display ;
        Value_Mask      : in Keyboard_Control_Mask ;
        Values          : in Keyboard_Control_Record  );

function  X_Get_Keyboard_Control   (
        Display_Id      : in Display  )
    return Keyboard_State_Record;

procedure X_Auto_Repeat_On         (
        Display_Id      : in Display  );

procedure X_Auto_Repeat_Off        (
        Display_Id      : in Display  );

procedure X_Bell                   (
        Display_Id      : in Display ;
        Volume          : in Bell_Volume_Type  );

procedure X_Query_Keymap           (
        Display_Id      : in Display ;
        Keys            : out String  );

procedure X_Get_Keyboard_Mapping   (
        Display_Id      : in Display ;
        First_Keycode   : in X_Integer ;
        Keycode_Count   : in X_Integer ;
        Keysyms_Per_Keycode : out X_Integer ;
        Keysyms         : out Key_Sym_List  );
--* This function syntax has been changed. The old STARS data
--* type Keyboard_Encoding_Array has been discontinued in this
--* implementation. We felt that the original STARS syntax would
--* lead to code that would be subject to possible memory leaks.

procedure X_Change_Keyboard_Mapping(
        Display_Id      : in Display ;
        First_Keycode   : in X_Integer ;
        Keysyms_Per_Keycode : in X_Integer ;
        Keysyms         : in Key_Sym_Array ;
        Num_Keycodes    : in X_Integer  );
--* This function syntax has been changed. The old STARS data
```

```
--* type Keyboard_Encoding_Array has been discontinued in this
--* implementation. We felt that the original STARS syntax would
--* lead to code that would be subject to possible memory leaks.

procedure X_Display_Keycodes       (
        Display_Id      : in Display ;
        Min_Keycode_Return : out Keyboard.Keycode ;
        Max_Keycode_Return : out Keyboard.Keycode  );

function  X_New_Modifier_Map       (
        Max_Keys_Per_Mod : in X_Long_Integer  )
    return Modifier_Keymap;

function  X_Set_Modifier_Mapping   (
        Display_Id      : in Display ;
        Modifiers       : in Modifier_Keymap  )
    return Map_Status;

function  X_Insert_Modifier_Map_Entry(
        Map             : in Modifier_Keymap ;
        Key             : in Keycode ;
        Modifier        : in Modifier_Type  )
    return Modifier_Keymap;

function  X_Delete_Modifier_Map_Entry(
        Map             : in Modifier_Keymap ;
        Key             : in Keycode ;
        Modifier        : in Modifier_Type  )
    return Modifier_Keymap;

function  X_Get_Modifier_Mapping   (
        Display_Id      : in Display  )
    return Modifier_Keymap;

function  Is_Keypad_Key            (
        Sym             : in Key_Sym  )
    return Boolean;

function  Is_Cursor_Key            (
        Sym             : in Key_Sym  )
    return Boolean;

function  Is_Pf_Key                (
        Sym             : in Key_Sym  )
    return Boolean;

function  Is_Function_Key          (
        Sym             : in Key_Sym  )
    return Boolean;

function  Is_Misc_Function_Key     (
        Sym             : in Key_Sym  )
    return Boolean;
```

```
function  Is_Modifier_Key          (
        Sym             : in Key_Sym  )
    return Boolean;

function Create_Constant          (
        Value           : in X_Long_Integer  )
    return Led_Mode_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Auto_Repeat_Mode_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Modifier_Type;
    return Modifier_Type;
```

Package X_Lib.Events

```
procedure X_Select_Input           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Mask            : in Event_Mask_Type  );

procedure X_Flush                  (
        Display_Id      : in Display  );

procedure X_Sync                   (
        Display_Id      : in Display ;
        Discard         : in Boolean := False );

function  X_Events_Queued          (
        Display_Id      : in Display ;
        Mode            : in Event_Queue_Mode_Type  )
    return Natural;

function  X_Pending                (
        Display_Id      : in Display  )
    return Natural;

procedure X_Next_Event             (
        Display_Id      : in Display ;
        The_Event       : in out Event  );

procedure X_Peek_Event             (
        Display_Id      : in Display ;
        The_Event       : in out Event  );

procedure X_Put_Back_Event         (
        Display_Id      : in Display ;
        The_Event       : in Event  );

procedure X_Window_Event           (
        Display_Id      : in Display ;
```

```
        Window_Id       : in Window ;
        Mask            : in Event_Mask_Type ;
        The_Event       : in out Event  );

procedure X_Check_Window_Event     (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Mask            : in Event_Mask_Type ;
        The_Event       : in out Event ;
        Event_Found     : in out Boolean  );

procedure X_Check_Typed_Event      (
        Display_Id      : in Display ;
        Event_Kind      : in Event_Type ;
        Event_Return    : in out Event ;
        Event_Found     : in out Boolean  );

procedure X_Check_Typed_Window_Event(
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Event_Kind      : in Event_Type ;
        Event_Return    : in out Event ;
        Event_Found     : in out Boolean  );

procedure X_Mask_Event             (
        Display_Id      : in Display ;
        Mask            : in Event_Mask_Type ;
        The_Event       : in out Event  );

procedure X_Check_Mask_Event       (
        Display_Id      : in Display ;
        Mask            : in Event_Mask_Type ;
        The_Event       : in out Event ;
        Event_Found     : in out Boolean  );

function  X_Get_Motion_Events      (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Start           : in Time ;
        Stop            : in Time  )
    return Time_Coord_List;

function  X_Display_Motion_Buffer_Size(
        Display_Id      : in Display  )
    return Natural;

procedure X_Send_Event             (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Propagate       : in Boolean ;
        Mask            : in Event_Mask_Type ;
        The_Event       : in Event  );

function  X_Lookup_Keysym          (
```

```
        The_Event       : in Event ;
        Index           : in Natural  )
    return Keyboard.Key_Sym;

procedure X_Refresh_Keyboard_Mapping(
        The_Event       : in Event  );

procedure X_Lookup_String          (
        The_Event       : in Event ;
        Buffer          : in out String_Pointer ;
        The_Symbol      : out Keyboard.Key_Sym ;
        Status          : out Compose_Status_Record  );

generic
  with procedure Predicate (Dpy       : Display;
         The_Event : in out Event;
             Arg       : in String;
         Status    : out Boolean);
procedure X_Peek_If_Event          (
        Dpy             : in Display ;
        The_Event       : in out Event ;
        Arg             : in String  );
--* This function syntax has been changed. The generic procedure
--* now does not define any generic formal parameters other than
--* the subprogram Predicate. The parameter Arg is now of type
--* String.

generic
  with procedure Predicate (Dpy       : Display;
         The_Event : in out Event;
             Arg       : in String;
         Status    : out Boolean);
procedure X_Check_If_Event         (
        Dpy             : Display  ;
        The_Event       : in out Event ;
        Arg             : in String ;
        Status          : out Boolean  );
--* This function syntax has been changed. The generic procedure
--* now does not define any generic formal parameters other than
--* the subprogram Predicate. The parameter Arg is now of type
--* String.

function Create_Constant          (
        Value           : in X_Long_Integer  )
    return Event_Queue_Mode_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Button_Name_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Notify_Mode_Type;
```

```
function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Notify_Detail_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Mapping_Request_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Colormap_State_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Property_State_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Placement_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Visibility_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Graphic_Expose_Code_Type;
```

Package X_Lib.Window_Manager

```
procedure X_Grab_Button            (
        Display_Id      : in Display ;
        Button          : in Events.Button_Name_Type ;
        Modifiers       : in Events.Key_And_Button_Mask ;
        Window_Id       : in Window ;
        Owner_Events    : in Boolean ;
        Grab_Events     : in Events.Event_Mask_Type ;
        Pointer_Mode    : in Grab_Mode ;
        Keyboard_Mode   : in Grab_Mode ;
        Confine_To      : in Window ;
        Cursor_Id       : in Cursors.Cursor  );

procedure X_Ungrab_Button          (
        Display_Id      : in Display ;
        Button          : in Events.Button_Name_Type ;
        Modifiers       : in Events.Key_And_Button_Mask ;
        Window_Id       : in Window  );

function  X_Grab_Pointer           (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Owner_Events    : in Boolean ;
        Grab_Events     : in Events.Event_Mask_Type ;
        Pointer_Mode    : in Grab_Mode ;
```

```
        Keyboard_Mode   : in Grab_Mode ;
        Confine_To      : in Window ;
        Cursor_Id       : in Cursors.Cursor ;
        Grab_Time       : in Time  )
    return Grab_Reply_Status;

procedure X_Ungrab_Pointer         (
        Display_Id      : in Display ;
        Ungrab_Time     : in Time  );

procedure X_Change_Active_Pointer_Grab(
        Display_Id      : in Display ;
        Grab_Events     : in Events.Event_Mask_Type ;
        Cursor_Id       : in Cursors.Cursor ;
        Grab_Time       : in Time  );

function  X_Grab_Keyboard          (
        Display_Id      : in Display ;
        Window_Id       : in Window ;
        Owner_Events    : in Boolean ;
        Pointer_Mode    : in Grab_Mode ;
        Keyboard_Mode   : in Grab_Mode ;
        Grab_Time       : in Time  )
    return Grab_Reply_Status;

procedure X_Ungrab_Keyboard        (
        Display_Id      : in Display ;
        Ungrab_Time     : in Time  );

procedure X_Grab_Key               (
        Display_Id      : in Display ;
        Key             : in Keyboard.Keycode ;
        Modifiers       : in Events.Key_And_Button_Mask ;
        Owner_Events    : in Boolean ;
        Window_Id       : in Window ;
        Pointer_Mode    : in Grab_Mode ;
        Keyboard_Mode   : in Grab_Mode  );

procedure X_Ungrab_Key             (
        Display_Id      : in Display ;
        Key             : in Keyboard.Keycode ;
        Modifiers       : in Events.Key_And_Button_Mask ;
        Window_Id       : in Window  );

procedure X_Allow_Events           (
        Display_Id      : in Display ;
        Mode            : in Allow_Event_Mode ;
        Event_Time      : in Time  );

procedure X_Grab_Server            (
        Display_Id      : in Display  );

procedure X_Ungrab_Server          (
        Display_Id      : in Display  );
```

```
procedure X_Warp_Pointer           (
        Display_Id      : in Display ;
        Source          : in Window ;
        Destination     : in Window ;
        Source_X        : in Coordinate ;
        Source_Y        : in Coordinate ;
        Source_Width    : in Width_Height ;
        Source_Height   : in Width_Height ;
        Destination_X   : in Coordinate ;
        Destination_Y   : in Coordinate  );

procedure X_Set_Input_Focus        (
        Display_Id      : in Display ;
        Focus           : in Window ;
        Mode            : in Revert_Mode ;
        Grab_Time       : in Time  );

procedure X_Get_Input_Focus        (
        Display_Id      : in Display ;
        Focus           : in out Window ;
        Mode            : in out Revert_Mode  );

procedure X_Change_Pointer_Control (
        Display_Id      : in Display ;
        Do_Accel        : in Boolean ;
        Do_Threshold    : in Boolean ;
        Numerator       : in Integer ;
        Denominator     : in Integer ;
        Threshold       : in Integer  );

procedure X_Get_Pointer_Control    (
        Display_Id      : in Display ;
        Numerator       : out Integer ;
        Denominator     : out Integer ;
        Threshold       : out Integer  );

procedure X_Set_Pointer_Mapping    (
        Display_Id      : in Display ;
        Map             : in Map_Array ;
        Status          : out Map_Status  );

procedure X_Get_Pointer_Mapping    (
        Display_Id      : in Display ;
        Map             : in out Map_Array ;
        Status          : out Map_Status  );

procedure X_Set_Close_Down_Mode    (
        Display_Id      : in Display ;
        Mode            : in Close_Mode  );

procedure X_Kill_Client            (
        Display_Id      : in Display ;
        Resource_Id     : in Resource  );
```

```
procedure X_Set_Screen_Saver       (
        Display_Id      : in Display ;
        Time_Out        : in Time ;
        Interval        : in Time ;
        Blanking        : in Blanking_Type ;
        Exposures       : in Exposure_Type  );

procedure X_Force_Screen_Saver     (
        Display_Id      : in Display ;
        Mode            : in Screen_Saver_Mode  );

procedure X_Activate_Screen_Saver  (
        Display_Id      : in Display  );

procedure X_Reset_Screen_Saver     (
        Display_Id      : in Display  );

procedure X_Get_Screen_Saver       (
        Display_Id      : in Display ;
        Time_Out        : out Time ;
        Interval        : out Time ;
        Blanking        : out Blanking_Type ;
        Exposures       : out Exposure_Type  );

procedure X_Add_Host               (
        Display_Id      : in Display ;
        Host            : in Host_Address_Record  );

procedure X_Add_Hosts              (
        Display_Id      : in Display ;
        Hosts           : in Host_Array  );

procedure X_List_Hosts             (
        Display_Id      : in Display ;
        States          : out Boolean ;
        Hosts           : in out Host_List  );
--* This function syntax has been changed. The original syntax
--* was erroneous. The second parameter has been correctly changed
--* to a Boolean status which indicates the state of access to the
--* control list at connection setup.

procedure X_Remove_Host            (
        Display_Id      : in Display ;
        Host            : in Host_Address_Record  );

procedure X_Remove_Hosts           (
        Display_Id      : in Display ;
        Hosts           : in Host_Array  );

procedure X_Set_Access_Control     (
        Display_Id      : in Display ;
        Mode            : in Access_Control_Mode  );
```

```
procedure X_Enable_Access_Control  (
        Display_Id      : in Display  );

procedure X_Disable_Access_Control (
        Display_Id      : in Display  );

function X_Get_Default            (
        Display_Id      : in Display ;
        Program         : in String ;
        Option          : in String ) return String;

function  X_Resource_Manager_String(
        Display_Id      : in Display  )
    return String_Pointer;

function Create_Constant          (
        Value           : in X_Long_Integer  )
    return Grab_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Grab_Reply_Status;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Notify_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Notify_Detail;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Visibility_Notify_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Circulation_Request;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Protocol_Family;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Property_Notification;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Colormap_Notification;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Allow_Event_Mode;
```

```
function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Revert_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Close_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Screen_Saver_Mode;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Exposure_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Blanking_Type;

function  Create_Constant          (
        Value           : in X_Long_Integer  )
    return Access_Control_Mode;
```

Package X_Lib.Resource Manager

```
function  Xrm_Get_File_Database    (
        Filename        : in String  )
    return Xrm_Database;

procedure Xrm_Get_Resource         (
        Database        : in Xrm_Database ;
        Str_Name        : in String ;
        Str_Class       : in String ;
        Str_Type        : out String_Pointer ;
        Db_Value        : out Xrm_Value ;
        Status          : out Boolean  );

function  Xrm_Get_String_Database  (
        Data            : in String  )
    return Xrm_Database;

procedure Xrm_Initialize           ;

procedure Xrm_Merge_Databases      (
        Source_Db       : in Xrm_Database ;
        Target_Db       : in out Xrm_Database  );

procedure Xrm_Parse_Command        (
        Db              : in out Xrm_Database ;
        Table           : in Xrm_Option_Desc_Array ;
        Table_Count     : in X_Integer ;
        Name            : in String ;
```

```
            Argc            : in out X_Long_Integer ;
            Argv            : in out String_List  );

        procedure Xrm_Put_File_Database    (
            Database        : in Xrm_Database ;
            Filename        : in String  );

        procedure Xrm_Put_Line_Resource    (
            Database        : in out Xrm_Database ;
            Line            : in String  );

        procedure Xrm_Put_Resource         (
            Database        : in out Xrm_Database ;
            Specifier       : in String ;
            Res_Type        : in String ;
            Res_Value       : in Xrm_Value  );

        procedure Xrm_Put_String_Resource  (
            Database        : in out Xrm_Database ;
            Resource        : in String ;
            Res_Value       : in String  );

        procedure Xrm_Q_Get_Resource       (
            Database        : in Xrm_Database ;
            Quark_Name      : in Xrm_Name_Array ;
            Quark_Class     : in Xrm_Class_Array ;
            Quark_Type      : out Xrm_Representation ;
            Db_Value        : out Xrm_Value ;
            Status          : out Boolean  );

        procedure Xrm_Q_Get_Resource       (
            Database        : in Xrm_Database ;
            Quark_Name      : in Xrm_Name_List ;
            Quark_Class     : in Xrm_Class_List ;
            Quark_Type      : out Xrm_Representation ;
            Db_Value        : out Xrm_Value ;
            Status          : out Boolean  );

        procedure Xrm_Q_Get_Search_List    (
            Database        : in Xrm_Database ;
            Names           : in Xrm_Name_List ;
            Classes         : in Xrm_Class_List ;
            Search_List     : in out Xrm_Searchlist ;
            List_Length     : in X_Integer ;
            Status          : out Boolean  );

        procedure Xrm_Q_Get_Search_List    (
            Database        : in Xrm_Database ;
            Names           : in Xrm_Name_Array ;
            Classes         : in Xrm_Class_Array ;
            Search_List     : in out Xrm_Searchlist ;
            List_Length     : in X_Integer ;
            Status          : out Boolean  );
```

```
procedure Xrm_Q_Get_Search_Resource(
        Search_List     : in Xrm_Searchlist ;
        Name            : in Xrm_Name ;
        Class           : in Xrm_Class ;
        Data_Type       : out Xrm_Representation ;
        Db_Value        : in out Xrm_Value ;
        Status          : out Boolean  );

procedure Xrm_Q_Put_Resource       (
        Database        : in out Xrm_Database ;
        Bindings        : in Xrm_Binding_Array ;
        Quarks          : in Xrm_Quark_Array ;
        Res_Type        : in Xrm_Representation ;
        Res_Value       : in Xrm_Value  );

procedure Xrm_Q_Put_Resource       (
        Database        : in out Xrm_Database ;
        Bindings        : in Xrm_Binding_List ;
        Quarks          : in Xrm_Quark_List ;
        Res_Type        : in Xrm_Representation ;
        Res_Value       : in Xrm_Value  );

procedure Xrm_Q_Put_String_Resource(
        Database        : in out Xrm_Database ;
        Bindings        : in Xrm_Binding_List ;
        Quarks          : in Xrm_Quark_List ;
        Str_Value       : in String  );

procedure Xrm_Q_Put_String_Resource(
        Database        : in out Xrm_Database ;
        Bindings        : in Xrm_Binding_Array ;
        Quarks          : in Xrm_Quark_Array ;
        Str_Value       : in String  );

function  Xrm_Quark_To_String      (
        Quark           : in Xrm_Quark  )
    return String;

function Xrm_Quark_To_String      (
        Quark           : in Xrm_Quark  )
    return Xrm_String;

function  Xrm_Name_To_String       (
        Name            : in Xrm_Name  )
    return String

function  Xrm_Name_To_String       (
        Name            : in Xrm_Name  )
    return Xrm_String

function  Xrm_Class_To_String      (
        Class           : in Xrm_Class  )
    return String
```

```
function  Xrm_Class_To_String      (
        Class           : in Xrm_Class  )
    return Xrm_String

function  Xrm_Representation_To_String(
        Representation  : in Xrm_Representation  )
    return String

function  Xrm_Representation_To_String(
        Representation  : in Xrm_Representation  )
    return Xrm_String

procedure Xrm_String_To_Binding_Quark_List(
        Str             : in String ;
        Bindings        : in out Xrm_Binding_Array ;
        Quarks          : in out Xrm_Quark_Array  );

procedure Xrm_String_To_Binding_Quark_List(
        Str             : in String ;
        Bindings        : in out Xrm_Binding_List ;
        Quarks          : in out Xrm_Quark_List  );

function  Xrm_String_To_Quark      (
        Str             : in String  )
    return Xrm_Quark;

function  Xrm_String_To_Quark      (
        Str             : in Xrm_String  )
    return Xrm_Quark;

function  Xrm_String_To_Name       (
        Str             : in String  )
    return Xrm_Name

function  Xrm_String_To_Name       (
        Str             : in Xrm_String  )
    return Xrm_Name

function  Xrm_String_To_Class      (
        Str             : in String  )
    return Xrm_Class

function  Xrm_String_To_Class      (
        Str             : in Xrm_String  )
    return Xrm_Class

function  Xrm_String_To_Representation(
        Str             : in String  )
    return Xrm_Representation

function  Xrm_String_To_Representation(
        Str             : in Xrm_String  )
    return Xrm_Representation
```

```
        procedure Xrm_String_To_Quark_List (
                Str             : in String ;
                Quarks          : out Xrm_Quark_Array  );

        procedure Xrm_String_To_Quark_List (
                Str             : in Xrm_String ;
                Quarks          : out Xrm_Quark_List  );

        procedure Xrm_String_To_Name_List  (
                Str             : in String ;
                Quarks          : out Xrm_Name_Array  )

        procedure Xrm_String_To_Name_List  (
                Str             : in Xrm_String ;
                Quarks          : out Xrm_Name_List  )

        procedure Xrm_String_To_Class_List (
                Str             : in String ;
                Quarks          : out Xrm_Class_Array  )

        procedure Xrm_String_To_Class_List (
                Str             : in Xrm_String ;
                Quarks          : out Xrm_Class_List  )

        function  Xrm_Unique_Quark
            return Xrm_Quark;

--      function  New_Database
--          return Xrm_Database;
--      --* This function has been deleted because it is no longer
--      --* meaningful in this implementation which treats the data
--      --* Xrm_Database as an opaque data type as does the C
--      --* version of X11R4.

        procedure Xrm_Destroy_Database     (
                Db              : in out Xrm_Database  );

--      procedure Xrm_Destroy_Database     (
--              The_Display     : in out Display  );
--      --* This function has been deleted because it accesses a
--      --* particular component of the display record in the
--      --* STARS implementation. This component is not accessed in
--      --* the C version of X11R4, and hence is not accessed in AXI.
--      --* While it could be accessed to complete the STARS binding,
--      --* we feel that the absence from the C interface indicates
--      --* an intended abstraction, so we have deleted the associated
--      --* STARS function.

      function  Xrm_Null_Db
          return Xrm_Database;

--      function  Get_Display_Resource_Db  (
--              Display_Id      : in Display  )
--          return Xrm_Database;
```

```
--      --* This function has been deleted because it accesses a
--      --* particular component of the display record in the
--      --* STARS implementation. This component is not accessed in
--      --* the C version of X11R4, and hence is not accessed in AXI.
--      --* While it could be accessed to complete the STARS binding,
--      --* we feel that the absence from the C interface indicates
--      --* an intended abstraction, so we have deleted the associated
--      --* STARS function.

--      procedure Set_Display_Resource_Db  (
--              Display_Id      : in out Display ;
--              Database        : in Xrm_Database  );
--      --* This function has been deleted because it accesses a
--      --* particular component of the display record in the
--      --* STARS implementation. This component is not accessed in
--      --* the C version of X11R4, and hence is not accessed in AXI.
--      --* While it could be accessed to complete the STARS binding,
--      --* we feel that the absence from the C interface indicates
--      --* an intended abstraction, so we have deleted the associated
--      --* STARS function.

--      function  Xrm_Create_Searchlist    (
--              Size            : in Positive  )
--          return Xrm_Searchlist;
--      --* This function has been deleted because it is no longer
--      --* meaningful in this implementation which treats the data
--      --* Xrm_Searchlist as an opaque data type as does the C
--      --* version of X11R4.

--      procedure Xrm_Free_Searchlist      (
--              List            : in out Xrm_Searchlist  );
--      --* This function has been deleted because it is no longer
--      --* meaningful in this implementation which treats the data
--      --* Xrm_Searchlist as an opaque data type as does the C
--      --* version of X11R4.
--      --* version of X11R4.
```

A

Data Types

This appendix contains a listing of some of the complex data type definitions, included in packages Xlib, Xt and Xm. For a complete description of these packages see the package specifications provided with the Ada libraries.

Relevant portions of package Xlib

```
--************************************************************
--
--  Start of the actual X Interface package.
--************************************************************

subtype Address               is BasicTypes.Address;
subtype Char                  is BasicTypes.Char;
subtype Char_Ptr              is BasicTypes.Char_Ptr;
subtype CharList              is BasicTypes.CharList;
subtype CharList_Ptr          is BasicTypes.CharList_Ptr;
subtype Double                is BasicTypes.Double;
subtype Double_Ptr            is BasicTypes.Double_Ptr;
subtype DoubleList            is BasicTypes.DoubleList;
subtype DoubleList_Ptr        is BasicTypes.DoubleList_Ptr;
subtype Real                  is BasicTypes.Real;
subtype Real_Ptr              is BasicTypes.Real_Ptr;
subtype RealList              is BasicTypes.RealList;
subtype RealList_Ptr          is BasicTypes.RealList_Ptr;
subtype Int                   is BasicTypes.Int;
subtype IntList               is BasicTypes.IntList;
subtype Int_Ptr               is BasicTypes.Int_Ptr;
subtype IntList_Ptr           is BasicTypes.IntList_Ptr;
subtype Long                  is BasicTypes.Long;
subtype Long_Ptr              is BasicTypes.Long_Ptr;
subtype LongList              is BasicTypes.LongList;
subtype LongList_Ptr          is BasicTypes.LongList_Ptr;
subtype Pointer               is BasicTypes.Pointer;
subtype Pointer_Ptr           is BasicTypes.Pointer_Ptr;
subtype PointerList           is BasicTypes.PointerList;
subtype PointerList_Ptr       is BasicTypes.PointerList_Ptr;
subtype Short                 is BasicTypes.Short;
subtype Short_Ptr             is BasicTypes.Short_Ptr;
subtype ShortList             is BasicTypes.ShortList;
subtype ShortList_Ptr         is BasicTypes.ShortList_Ptr;
subtype StringList            is BasicTypes.StringList;
subtype StringList_Ptr        is BasicTypes.StringList_Ptr;
subtype String_Ptr            is BasicTypes.String_Ptr;
subtype NullString            is BasicTypes.NullString;
```

```
  subtype UnsignedChar          is BasicTypes.UnsignedChar;
  subtype UnsignedChar_Ptr      is BasicTypes.UnsignedChar_Ptr;
  subtype UnsignedCharList      is BasicTypes.UnsignedCharList;
  subtype UnsignedCharList_Ptr  is BasicTypes.UnsignedCharList_Ptr;
  subtype UnsignedInt           is BasicTypes.UnsignedInt;
  subtype UnsignedIntList       is BasicTypes.UnsignedIntList;
  subtype UnsignedLong          is BasicTypes.UnsignedLong;
  subtype UnsignedLong_Ptr      is BasicTypes.UnsignedLong_Ptr;
  subtype UnsignedLongList      is BasicTypes.UnsignedLongList;
  subtype UnsignedLongList_Ptr  is BasicTypes.UnsignedLongList_Ptr;
  subtype UnsignedShort         is BasicTypes.UnsignedShort;
  subtype UnsignedShort_Ptr     is BasicTypes.UnsignedShort_Ptr;
  subtype UnsignedShortList     is BasicTypes.UnsignedShortList;
  subtype UnsignedShortList_Ptr is BasicTypes.UnsignedShortList_Ptr;

  function "="( left, right : UnsignedShort ) return Boolean renames BasicTypes."=";
  function "<"( left, right : UnsignedShort ) return Boolean renames BasicTypes."<";
  function ">"( left, right : UnsignedShort ) return Boolean renames BasicTypes.">";
  function "-"( left, right : UnsignedShort ) return UnsignedShort
      renames BasicTypes."-";
  function "+"( left, right : UnsignedShort ) return UnsignedShort
      renames BasicTypes."+";
  function "*"( left, right : UnsignedShort ) return UnsignedShort
      renames BasicTypes."*";
  function "/"( left, right : UnsignedShort ) return UnsignedShort
      renames BasicTypes."/";

  function "="( left, right : System.Address ) return Boolean renames BasicTypes."=";
  function "-"( left, right : Address ) return Int renames BasicTypes."-";
  function "="( left, right : String_Ptr ) return Boolean renames BasicTypes."=";
  function "="( left, right : StringList_Ptr ) return Boolean renames BasicTypes."=";
  function "="( left, right : IntList_Ptr ) return Boolean renames BasicTypes."=";
  function "or"( left, right : UnsignedLong ) return UnsignedLong
    renames BasicTypes."or";
  function "and"( left, right : UnsignedLong ) return UnsignedLong
    renames BasicTypes."and";
  function "xor"( left, right : UnsignedLong ) return UnsignedLong
    renames BasicTypes."xor";

  function "or"( left, right : UnsignedChar ) return UnsignedChar
    renames BasicTypes."or";
  function "and"( left, right : UnsignedChar ) return UnsignedChar
    renames BasicTypes."and";
  function "xor"( left, right : UnsignedChar ) return UnsignedChar
    renames BasicTypes."xor";

  Address_Size  : constant := BasicTypes.Address_Size;
  Pointer_Size  : constant := BasicTypes.Pointer_Size;
  C_String_Size : constant := BasicTypes.C_String_Size;

  subtype Display is Address;
  type Bool       is new Int;
  subtype Status  is Int;

  type Pixel      is new UnsignedLong;
  type Pixel_Ptr  is access Pixel;
  type PixelList  is array ( Int range <> ) of Pixel;
  type PixelList_Ptr is access PixelList;
```

```
XNULL : constant Address := BasicTypes.C_NULL;

--=================================================
--
-- <X11/Xlib.h> Constants
--
type XID     is new UnsignedLong;
subtype Window    is XID;
subtype Drawable  is XID;
subtype Font      is XID;
subtype Pixmap    is XID;
subtype Cursor    is XID;
subtype Colormap  is XID;
subtype GContext  is XID;
subtype KeySym    is XID;

type PixmapList     is array ( Int range <> ) of Pixmap;
type PixmapList_Ptr is access PixmapList;

type FontList     is array ( Int range <> ) of Font;
type FontList_Ptr is access FontList;

type WindowList   is array ( Int range <> ) of Window;
type WindowList_Ptr is access WindowList;

type ColormapList is array ( Int range <> ) of Colormap;
type ColormapList_Ptr is access ColormapList;

type ModifierKeymap is new Address;
type ModifierKeymap_Ptr is new Address;

type KeySym_Ptr is access Keysym;
type KeySymList is array ( Int range <> ) of KeySym;
type KeySymList_Ptr is access KeySymList;

type Mask is new UnsignedLong;

type Atom is new UnsignedLong;
type Atom_Ptr is access Atom;
type AtomList  is array ( Int range <> ) of Atom;
type AtomList_Ptr is access AtomList;

type VisualId  is new UnsignedLong;
type Time is new UnsignedLong;
type Context   is new Int;

subtype Keycode   is UnsignedChar;
type Keycode_Ptr is access KeyCode;
type KeycodeList is array ( Int range <> ) of KeyCode;
type KeycodeList_Ptr is access KeyCodeList;

--=============================================================
-- X11 Constants from <X11/Xlib.h>
--
XTrue              : constant := 1;
XFalse             : constant := 0;

QueuedAlready      : constant:= 0;
QueuedAfterReading : constant:= 1;
```

```
  QueuedAfterFlush   : constant:= 2;


  --=============================================================
  -- RESERVED RESOURCE AND CONSTANT DEFINITIONS
  -- <X11/Xlib.h>
  --=============================================================

  None     : constant UnsignedLong := 0; -- universal null ressource or
                                 -- null ataom.
  ParentRelative  : constant UnsignedLong := 1; -- background pixmap in CreateWindow
                            -- and ChangeWindowAttributes.

  --
  --  Copyfrom Parent is now a function, and declared just before the macros.
  --
  --  CopyFromParent  : constant UnsignedLong := 0; -- border pixmap in CreateWindow
  --                                                -- and ChangeWindowAttributes
  --                                                -- special VisualID and special
window
  --                                                -- class passed to CreateWindow

  PointerWindow   : constant UnsignedLong := 1; -- destination window in SendEvent
  InputFocus      : constant UnsignedLong := 1; -- destination window in SendEvent
  PointerRoot     : constant UnsignedLong := 1; -- focus window in SetInputFocus
  AnyPropertyType : constant UnsignedLong := 0; -- special Atom, passed to GetProperty
  AnyKey          : constant UnsignedLong := 0; -- special Key Code, passed to GrabKey
  AnyButton       : constant UnsignedLong := 0; -- special Button Code, passed to
                                                -- GrabButton
  AllTemporary    : constant UnsignedLong := 0; -- special Resource ID passed to
                                                -- KillClient
  CurrentTime     : constant UnsignedLong := 0; -- special Time
  NoSymbol        : constant UnsignedLong := 0; -- special KeySym

  ------------------------------------------
  -- EVENT DEFINITIONS
  -- X11 Input Event Masks.
  -- Used as event-mask window attribute and as arguments
  -- to Grab requests.  Not to be confused with event names.
  ------------------------------------------

  NoEventMask      : constant := 16#0000000#;
  KeyPressMask     : constant := 16#0000001#;
  KeyReleaseMask   : constant := 16#0000002#;
  ButtonPressMask  : constant := 16#0000004#;
  ButtonReleaseMask: constant := 16#0000008#;
  EnterWindowMask   : constant := 16#0000010#;
  LeaveWindowMask   : constant := 16#0000020#;
  PointerMotionMask : constant := 16#0000040#;
  PointerMotionHINTMask   : constant := 16#0000080#;
  Button1MotionMask : constant := 16#0000100#;
  Button2MotionMask : constant := 16#0000200#;
  Button3MotionMask : constant := 16#0000400#;
  Button4MotionMask : constant := 16#0000800#;
  Button5MotionMask : constant := 16#0001000#;
  ButtonMotionMask  : constant := 16#0002000#;
  KeyMapStateMask   : constant := 16#0004000#;
  ExposureMask      : constant := 16#0008000#;
  VisibilityChangeMask   : constant := 16#0010000#;
```

```
StructureNotifyMask: constant := 16#0020000#;
ResizeRedirectMask : constant := 16#0040000#;
SubStructureNotifyMask   : constant := 16#0080000#;
SubStructureRedirectMask : constant := 16#0100000#;
FocusChangeMask   : constant := 16#0200000#;
PropertyChangeMask: constant := 16#0400000#;
ColorMapChangeMask: constant := 16#0800000#;
OwnerGrabButtonMask:constant := 16#1000000#;

-- Event names.  Used in “type” field in XEvent structures.  Not to be
-- confused with event masks above.  They start from 2 because 0 and 1
-- are reserved in the protocol for errors and replies.

NoEvent             : constant := 0;
KeyPress: constant := 2;
KeyRelease: constant := 3;
ButtonPress: constant := 4;
ButtonRelease: constant := 5;
MotionNotify: constant := 6;
EnterNotify: constant := 7;
LeaveNotify: constant := 8;
FocusIn: constant := 9;
FocusOut: constant := 10;
KeymapNotify: constant := 11;
Expose : constant := 12;
GraphicsExpose: constant := 13;
NoExpose: constant := 14;
VisibilityNotify: constant := 15;
CreateNotify: constant := 16;
DestroyNotify: constant := 17;
UnmapNotify: constant := 18;
MapNotify: constant := 19;
MapRequest: constant := 20;
ReparentNotify: constant := 21;
ConfigureNotify: constant := 22;
ConfigureRequest: constant := 23;
GravityNotify: constant := 24;
ResizeRequest: constant := 25;
CirculateNotify: constant := 26;
CirculateRequest: constant := 27;
PropertyNotify: constant := 28;
SelectionClear: constant := 29;
SelectionRequest: constant := 30;
SelectionNotify: constant := 31;
ColormapNotify: constant := 32;
ClientMessage: constant := 33;
MappingNotify: constant := 34;
LastEvent: constant := 35;  -- must be bigger than any event #

--
-- X11 Key masks and Button masks.
-- Used as modifiers to GrabButton and GrabKey,
-- results of QueryPointer,
-- state in various key-, mouse-, and button-related events.
--
ShiftMask : constant := 16#0001#;
LockMask: constant := 16#0002#;
ControlMask: constant := 16#0004#;
Mod1Mask: constant := 16#0008#;
```

```
 Mod2Mask: constant := 16#0010#;
 Mod3Mask: constant := 16#0020#;
 Mod4Mask: constant := 16#0040#;
 Mod5Mask: constant := 16#0080#;
--
-- modifier names.  Used to build a SetModifierMapping request or
-- to read a GetModifierMapping request.  These correspond to the
-- masks defined above.
--
 ShiftMapIndex: constant := 0;
 LockMapIndex: constant := 1;
 ControlMapIndex: constant := 2;
 Mod1MapIndex: constant := 3;
 Mod2MapIndex: constant := 4;
 Mod3MapIndex: constant := 5;
 Mod4MapIndex: constant := 6;
 Mod5MapIndex: constant := 7;
 --
 -- Button masks.  Used in same manner as key masks above.
 -- Not to be confused with button names below.
 --
 Button1Mask: constant := 16#0100#;   -- Left Button
 Button2Mask           : constant := 16#0200#;    -- Middle Button
 button3Mask : constant := 16#0400#;    -- Right Button
 Button4Mask: constant := 16#0800#;
 Button5Mask: constant := 16#1000#;
 AnyModifier: constant := 16#8000#;
 --
 -- button names. Used as arguments to GrabButton and as detail in ButtonPress
 -- and ButtonRelease events.  Not to be confused with button masks above.
 -- Note that 0 is already defined above as “AnyButton”.
 --
 Button1: constant := 1;
 Button2: constant := 2;
 Button3: constant := 3;
 Button4: constant := 4;
 Button5: constant := 5;
 --
 -- Notify modes
 --
 NotifyNormal: constant := 0;
 NotifyGrab: constant := 1;
 NotifyUngrab: constant := 2;
 NotifyWhileGrabbed: constant := 3;

 NotifyHint: constant := 1;  -- for MotionNotify events
 --
 -- Notify detail
 --
 NotifyAncestor : constant := 0;
 NotifyVirtual : constant := 1;
 NotifyInferior : constant := 2;
 NotifyNonlinear : constant := 3;
 NotifyNonlinearVirtual : constant := 4;
 NotifyPointer : constant := 5;
 NotifyPointerRoot : constant := 6;
 NotifyDetailNone : constant := 7;
 --
 -- Visibility notify
```

```
--
VisibilityUnobscured: constant := 0;
VisibilityPartiallyObscured: constant := 1;
VisibilityFullyObscured: constant := 2;
--
-- Circulation request
--
PlaceOnTop: constant := 0;
PlaceOnBottom: constant := 1;
--
-- Property notification
--
PropertyNewValue: constant := 0;
PropertyDelete: constant := 1;
--
-- Color Map notification
--
ColormapUninstalled: constant := 0;
ColormapInstalled: constant := 1;
--
-- GrabPointer, GrabButton, GrabKeyboard, GrabKey Modes
--
GrabModeSync: constant := 0;
GrabModeAsync: constant := 1;
--
-- GrabPointer, GrabKeyboard reply status
--
GrabSuccess: constant := 0;
AlreadyGrabbed: constant := 1;
GrabInvalidTime: constant := 2;
GrabNotViewable: constant := 3;
GrabFrozen: constant := 4;
--
-- AllowEvents modes
--
AsyncPointer: constant := 0;
SyncPointer: constant := 1;
ReplayPointer: constant := 2;
AsyncKeyboard: constant := 3;
SyncKeyboard: constant := 4;
ReplayKeyboard: constant := 5;
AsyncBoth: constant := 6;
SyncBoth: constant := 7;
--
-- Used in SetInputFocus, GetInputFocus
--
RevertToNone: constant INT := INT (None);
RevertToPointerRoot: constant INT := INT (PointerRoot);
RevertToParent: constant INT := 2;

--*****************************************************************
-- ERROR CODES
--*****************************************************************/

Success : constant Status := 0;  -- everything's okay
BadRequest : constant Status := 1;  -- bad request code
BadValue : constant Status := 2;  -- int parameter out of range
BadWindow : constant Status := 3;  -- parameter not a Window
BadPixmap : constant Status := 4;  -- parameter not a Pixmap
```

```
  BadAtom : constant Status := 5;  -- parameter not an Atom
  BadCursor : constant Status := 6;  -- parameter not a Cursor
  BadFont : constant Status := 7;  -- parameter not a Font
  BadMatch : constant Status := 8;  -- parameter mismatch
  BadDrawable : constant Status := 9;  -- parameter not a Pixmap or Window
  BadAccess : constant Status := 10; -- depending on context:
                    --   key/button already grabbed
                    --   attempt to free an illegal
                    --   cmap entry
                    --   attempt to store into a read-only
                    --   color map entry.
                    --   attempt to modify the access control
                    --   list from other than the local host.
  BadAlloc  : constant Status := 11;  -- insufficient resources
  BadColor        : constant Status := 12;  -- no such colormap
  BadGC       : constant Status := 13;  -- parameter not a GC
  BadIDChoice        : constant Status := 14;  -- choice not in range or already used
  BadName: constant Status := 15;  -- font or color name doesn't exist
  BadLength    : constant Status := 16;  -- Request length incorrect
  BadImplementation     : constant Status := 17;  -- server is defective

  FirstExtensionError: constant Status := 128;
  LastExtensionError: constant Status := 255;

  --****************************************************************
  --* WINDOW DEFINITIONS
  --****************************************************************/
  --
  -- Window Classes used by XCreateWindow
  -- Note that CopyFromParent is already defined as 0.
  --
  InputOutput       : constant := 1;
  InputOnly    : constant := 2;

  -- ------------------------------------------
  -- Window Attributes for CreateWindow and ChangeWindowAttributes.
  --
  CWBackPixmap: constant UnsignedLong := 16#0001#;
  CWBackPixel: constant UnsignedLong := 16#0002#;
  CWBorderPixmap: constant UnsignedLong := 16#0004#;
  CWBorderPixel         : constant UnsignedLong := 16#0008#;
  CWBitGravity: constant UnsignedLong := 16#0010#;
  CWWinGravity: constant UnsignedLong := 16#0020#;
  CWBackingStore        : constant UnsignedLong := 16#0040#;
  CWBackingPlanes: constant UnsignedLong := 16#0080#;
  CWBackingPixel: constant UnsignedLong := 16#0100#;
  CWOverrideRedirect: constant UnsignedLong := 16#0200#;
  CWSaveUnder: constant UnsignedLong := 16#0400#;
  CWEventMask: constant UnsignedLong := 16#0800#;
  CWDontPropagate: constant UnsignedLong := 16#1000#;
  CWColormap: constant UnsignedLong := 16#2000#;
  CWCursor        : constant UnsignedLong := 16#4000#;

  -- Bit Gravity

  ForgetGravity: constant := 0;
  NorthWestGravity: constant := 1;
  NorthGravity: constant := 2;
  NorthEastGravity: constant := 3;
```

```
WestGravity: constant := 4;
CenterGravity: constant := 5;
EastGravity: constant := 6;
SouthWestGravity: constant := 7;
SouthGravity: constant := 8;
SouthEastGravity: constant := 9;
StaticGravity: constant := 10;

--  Window gravity + bit gravity above

UnmapGravity: constant := 0;

-- Used in CreateWindow for backing-store hint

NotUseful             : constant := 0;
WhenMapped            : constant := 1;
Always                : constant := 2;

-- Used in GetWindowAttributes reply

IsUnmapped: constant := 0;
IsUnviewable: constant := 1;
IsViewable: constant := 2;

-- Used in ChangeSaveSet

SetModeInsert         : constant := 0;
SetModeDelete         : constant := 1;

-- Used in ChangeCloseDownMode

DestroyAll            : constant := 0;
RetainPermanent       : constant := 1;
RetainTemporary       : constant := 2;

-- Window stacking method (in configureWindow)

Above                 : constant := 0;
Below                 : constant := 1;
TopIf                 : constant := 2;
BottomIf              : constant := 3;
Opposite              : constant := 4;

-- Circulation direction

RaiseLowest           : constant := 0;
LowerHighest          : constant := 1;

-- Property modes

PropModeReplace       : constant := 0;
PropModePrepend       : constant := 1;
PropModeAppend        : constant := 2;

--****************************************************************
--* GRAPHICS DEFINITIONS
--****************************************************************/
--
-- GC Function values
```

```
--
GXclear: constant := 16#0000#;  -- 0
GXand               : constant := 16#0001#;  -- src AND dst
GXandReverse : constant := 16#0002#;  -- src AND NOT dst
GXcopy               : constant := 16#0003#;  -- src
GXandInverted: constant := 16#0004#;  -- NOT src AND dst
GXnoop : constant := 16#0005#;  -- dst
GXxor       : constant := 16#0006#;  -- src XOR dst
GXor        : constant := 16#0007#;  -- src OR dst
GXnor       : constant := 16#0008#;  -- NOT src AND NOT dst
GXequiv: constant := 16#0009#;  -- NOT src XOR dst
GXinvert: constant := 16#000A#;  -- NOT dst
GXXorReverse  : constant := 16#000B#;  -- src OR NOT dst
GXcopyInverted  : constant := 16#000C#;  -- NOT src
GXorInverted: constant := 16#000D#;  -- NOT src OR dst
GXnand : constant := 16#000E#;  -- NOT src OR NOT dst
GXset       : constant := 16#000F#;  -- 1
--
-- Line Styles (Polyline)
--
LineSolid          : constant := 0;
LineOnOffDash      : constant := 1;
LineDoubleDash     : constant := 2;
--
-- Cap Styles (Polyline)
--
CapNotLast         : constant := 0;
CapButt            : constant := 1;
CapRound           : constant := 2;
CapProjecting      : constant := 3;
--
-- Join Styles (Polyline)
--
JoinMiter          : constant := 0;
JoinRound          : constant := 1;
JoinBevel          : constant := 2;
--
-- Fill Styles (Fillarea)
--
FillSolid          : constant := 0;
FillTiled          : constant := 1;
FillStippled       : constant := 2;
FillOpaqueStippled : constant := 3;
--
-- Fill Rules
--
EvenOddRule        : constant := 0;
WindingRule        : constant := 1;
--
-- Subwindow mode
--
ClipByChildren     : constant := 0;
IncludeInferiors   : constant := 1;
--
-- SetClipRectangles ordering
--
Unsorted : constant INT := 0;
YSorted       : constant INT := 1;
YXSorted: constant INT := 2;
```

```
YXBanded: constant INT := 3;
--
-- Coordinate Mode for Drawing Primitives
--
CoordModeOrigin    : constant := 0; -- Relative to the origin.
CoordModePrevious  : constant := 1; -- Relative to previous point.
--
-- Fill Shapes
Complex     : constant := 0;  -- Paths may intersect
Nonconvex     : constant := 1;   -- No Paths interface, but not convex
Convex     : constant := 2;   -- Wholly convex
--
-- Arc modes for PolyFillArc
--
ArcChord     : constant := 0;   -- Join endpoints of arc.
ArcPieSlice     : constant := 1;   -- Joint endpoints to center of arc.
--
-- X11 GC components: masks used in
-- CreateGC, CopyGC, ChangeGC, OR'ed into
-- GC.stateChanges
--
GCFunction: constant UnsignedLong := 16#000001#;
GCPlaneMask: constant UnsignedLong := 16#000002#;
GCForeground: constant UnsignedLong := 16#000004#;
GCBackground: constant UnsignedLong := 16#000008#;
GCLineWidth: constant UnsignedLong := 16#000010#;
GCLineStyle : constant UnsignedLong := 16#000020#;
GCCapStyle: constant UnsignedLong := 16#000040#;
GCJoinStyle: constant UnsignedLong := 16#000080#;
GCFillStyle : constant UnsignedLong := 16#000100#;
GCFillRule  : constant UnsignedLong := 16#000200#;
GCTile     : constant UnsignedLong := 16#000400#;
GCStipple   : constant UnsignedLong := 16#000800#;
GCTileStipXOrigin: constant UnsignedLong := 16#001000#;
GCTileStipYOrigin: constant UnsignedLong := 16#002000#;
GCFont      : constant UnsignedLong := 16#004000#;
GCSubwindowMode: constant UnsignedLong := 16#008000#;
GCGraphicsExposures: constant UnsignedLong := 16#010000#;
GCClipXOrigin: constant UnsignedLong := 16#020000#;
GCClipYOrigin: constant UnsignedLong := 16#040000#;
GCClipMask : constant UnsignedLong := 16#080000#;
GCDashOffset: constant UnsignedLong := 16#100000#;
GCDashList  : constant UnsignedLong := 16#200000#;
GCArcMode  : constant UnsignedLong := 16#400000#;

GCLastBit: constant := 22;

--
--  Fonts
--
--
-- used in QueryFont -- draw direction
--
FontLeftToRight: constant := 0;
FontRightToLeft: constant := 1;

FontChange: constant := 255;

--
```

```
--  Imaging
--
--
--  ImageFormat -- PutImage, GetImage
--
XYBitmap: constant := 0;-- depth 1, XYFormat
XYPixmap: constant := 1;-- depth == drawable depth
ZPixmap: constant := 2;-- depth == drawable depth

--
--  Color Map Stuff
--
--
-- For CreateColormap
--
AllocNone: constant := 0;-- create map with no entries
AllocAll: constant := 1;-- allocate entire map writeable
--
-- Flags used in StoreNamedColor, StoreColors
--
DoRed  : constant UnsignedChar := 1;
DoGreen: constant UnsignedChar := 2;
DoBlue: constant UnsignedChar := 4;
DoAll        : constant UnsignedChar := 7;

--*****************************************************************
--* CURSOR STUFF
--*****************************************************************/
--
-- QueryBestSize Class
--
CursorShape: constant := 0;-- largest size that can be displayed
TileShape: constant := 1; -- size tiled fastest
StippleShape: constant := 2;-- size stippled fastest *

--*****************************************************************
--* KEYBOARD/POINTER STUFF
--*****************************************************************/

MappingSuccess      : constant := 0;
MappingBusy         : constant := 1;
MappingFailed  : constant := 2;

MappingModifier : constant := 0;
MappingKeyboard : constant := 1;
MappingPointer : constant := 2;

--*****************************************************************
--* SCREEN SAVER STUFF
--*****************************************************************/

DontPreferBlanking: constant := 0;
PreferBlanking: constant := 1;
DefaultBlanking: constant := 2;

DisableScreenSaver: constant := 0;
DisableScreenInterval: constant := 0;  -- I wonder if this is right.

DontAllowExposures: constant := 0;
```

```
AllowExposures: constant := 1;
DefaultExposures: constant := 2;
--
-- for ForceScreenSaver
--
ScreenSaverReset      : constant := 0;
ScreenSaverActive     : constant := 1;

--*****************************************************************
--* HOSTS AND CONNECTIONS
--*****************************************************************/
--
-- for ChangeHosts
--
HostInsert: constant := 0;
HostDelete: constant := 1;
--
-- for ChangeAccessControl
--
EnableAccess: constant := 1;
DisableAccess: constant := 0;
--
-- Display classes  used in opening the connection
-- Note that the statically allocated ones are even numbered and the
-- dynamically changeable ones are odd numbered
--
StaticGray: constant := 0;
GrayScale: constant := 1;
StaticColor: constant := 2;
PseudoColor: constant := 3;
TrueColor: constant := 4;
DirectColor: constant := 5;
--
-- Byte order  used in imageByteOrder and bitmapBitOrder
--
LSBFirst: constant := 0;
MSBFirst: constant := 1;

--===============================================================
-- <X11/Intrinsic.h>
--

--===============================================================
-- <X11/Xutil.h>
--
--
-- Bitmask returned by XParseGeometry().  Each bit tells if the corresponding
-- value (x, y, width, height) was found in the parsed string.
--
NoValue: constant := 16#0000#;
XValue  : constant := 16#0001#;
YValue: constant := 16#0002#;
WidthValue  : constant := 16#0004#;
HeightValue  : constant := 16#0008#;
AllValues : constant := 16#000F#;
XNegative : constant := 16#0010#;
YNegative : constant := 16#0020#;

-----------------------------------------
```

```
  type ClassHint is
    record
      res_name  : Pointer;
      res_class : Pointer;
    end record;

  type ClassHintList is array ( Int range <> ) of ClassHint;
  type ClassHint_Ptr is access ClassHint;
  type ClassHintList_Ptr is access ClassHintList;

  -------------------------------------------

  type KeyboardControl is
    record
      key_click_percent : Int;
      bell_percent      : Int;
      bell_pitch        : Int;
      bell_duration     : Int;
      led               : Int;
      led_mode          : Int;
      key               : Int;
      auto_repeat_mode  : Int;
    end record;

  type KeyboardControlList is array ( Int range <> ) of KeyboardControl;
  type KeyboardControl_Ptr is access KeyboardControl;
  type KeyboardControlList_Ptr is access KeyboardControlList;

  --
  -- masks for ChangeKeyboardControl
  --
  KBKeyClickPercent : constant := 16#0001#;
  KBBellPercent  : constant := 16#0002#;
  KBBellPitch : constant := 16#0004#;
  KBBellDuration : constant := 16#0010#;
  KBLed        : constant := 16#0020#;
  KBLedMode : constant := 16#0040#;
  KBKey        : constant := 16#0080#;
  KBAutoRepeatMode : constant := 16#0100#;

  LedModeOff : constant := 0;
  LedModeOn : constant := 1;

  AutoRepeatModeOff : constant := 0;
  AutoRepeatModeOn : constant := 1;
  AutoRepeatModeDefault  : constant := 2;

  -------------------------------------------

  type PixmapFormatValues is
    record
      depth          : Int;
      bits_per_pixel : Int;
      scanline_pad   : Int;
    end record;

  type PixmapFormatValues_Ptr is access PixmapFormatValues;
  type PixmapFormatValuesList is array ( Int range <> ) of PixmapFormatValues;
```

```
type PixmapFormatValuesList_Ptr is access PixmapFormatValuesList;

-------------------------------------------

type ComposeStatus is
  record
    compose_ptr   : Pointer;
    chars_matched : Int;
  end record;

type ComposeStatus_Ptr is access ComposeStatus;
type ComposeStatusList is array ( Int range <> ) of ComposeStatus;
type ComposeStatusList_Ptr is access ComposeStatusList;

-------------------------------------------


type KeyboardState is
  record
    key_click_percent  : Int;
    bell_percent       : Int;
    bell_pitch         : UnsignedInt;
    bell_duration      : UnsignedInt;
    led_mask           : UnsignedLong;
    global_auto_repeat : Int;
    auto_repeats       : UnsignedCharList( 1 .. 32 );
  end record;

type KeyboardStateList is array ( Int range <> ) of KeyboardState;
type KeyboardState_ptr is access KeyboardState;
type KeyboardStateList_ptr is access KeyboardStateList;

-------------------------------------------

type TimeCoord is
  record
    t : Time;
    x : Short;
    y : Short;
  end record;

type TimeCoord_Ptr is access TimeCoord;
type TimeCoordList is array ( Int range <> ) of TimeCoord;
type TimeCoordList_Ptr is access TimeCoordList;

-------------------------------------------

type Char2b is
  record
    byte1 : UnsignedChar;
    byte2 : UnsignedChar;
  end record;

type Char2b_Ptr is access Char2b;
type Char2bList is array ( Int range <> ) of Char2b;
type Char2bList_Ptr is access Char2bList;

-------------------------------------------
```

```
  type HostAddress is
    record
      family  : Int;
      length  : Int;
      addr    : Address;
    end record;

  type HostAddressList is array ( Int range <> ) of HostAddress;
  type HostAddress_Ptr is access HostAddress;
  type HostAddressList_Ptr is access HostAddressList;

  FamilyInternet : constant Int := 0;
  FamilyDECNet   : constant Int := 1;
  FamilyChaos    : constant Int := 2;

  -------------------------------------------

  type WindowChanges is
    record
      x            : Int;
      y            : Int;
      width        : Int;
      height       : Int;
      border_width : Int;
      sibling      : Window;
      stack_mode   : Int;
    end record;

  type WindowChangesList is array ( Int range <> ) of WindowChanges;
  type WindowChanges_Ptr is access WindowChanges;
  type WindowChangesList_Ptr is access WindowChangesList;

  -- ConfigureWindow structure

  CWX         : constant := 16#0001#;
  CWY         : constant := 16#0002#;
  CWWidth: constant := 16#0003#;
  CWHeight: constant := 16#0004#;
  CWBorderWidth: constant := 16#0010#;
  CWSibling: constant := 16#0020#;
  CWStackMode: constant := 16#0040#;

  -------------------------------------------

  type IconSize is
    record
      min_width  : Int;
      min_height : Int;
      max_width  : Int;
      max_height : Int;
      width_inc  : Int;
      height_inc : Int;
    end record;

  type IconSizeList is array ( Int range <> )  of IconSize;
  type IconSize_Ptr is access IconSize;
  type IconSizeList_Ptr is access IconSizeList;

  -------------------------------------------
```

```
type AspectRatio is
  record
    x   : Int;
    y   : Int;
  end record;

type AspectRatioList is array ( Int range <> ) of AspectRatio;
type AspectRatio_Ptr is access AspectRatio;
type AspectRatioList_Ptr is access AspectRatioList;

-------------------------------------------

type SizeHints is
  record
    flags        : Long;
    x            : Int;
    y            : Int;
    width        : Int;
    height       : Int;
    min_width    : Int;
    min_height   : Int;
    max_width    : Int;
    max_height   : Int;
    width_inc    : Int;
    height_inc   : Int;
    min_aspect   : AspectRatio;
    max_aspect   : AspectRatio;
    base_width   : Int;
    base_height  : Int;
    win_gravity  : Int;
  end record;

type SizeHintsList is array ( Int range <> ) of SizeHints;
type SizeHints_Ptr is access SizeHints;
type SizeHintsList_Ptr is access SizeHintsList;


-------------------------------------------
--
-- The next block of definitions are for window manager properties that
-- clients and applications use for communication.
--

--
-- flags argument in size hints
--
USPosition: constant Long := 16#0001#;  -- user specified x, y
USSize: constant Long := 16#0002#;  -- user specified width, height

PPosition: constant Long := 16#0004#;  -- program specified position
PSize  : constant Long := 16#0010#;  -- program specified size
PMinSize: constant Long := 16#0020#;  -- program specified minimum size
PMaxSize: constant Long := 16#0040#;  -- program specified maximum size
PResizeInc: constant Long := 16#0080#;  -- program specified resize increments
PAspect: constant Long := 16#0100#;  -- program specified min and max aspect ratios
PBaseSize: constant Long := 16#0200#;  -- program specified base for incrementing
PWinGravity: constant Long := 16#0400#;  -- program specified window gravity
PAllHints     : constant Long := 16#07ff#;
```

```
  ------------------------------------------

  type WMHints is
    record
      flags          : Long;
      input      : Bool;
      initial_state  : Int;
      icon_pixmap    : Pixmap;
      icon_window    : Pixmap;
      icon_x         : Int;
      icon_y         : Int;
      icon_mask      : Pixmap;
      window_group   : XID;
    end record;

  type WMHintsList is array ( Int range <> ) of WMHints;
  type WMHints_Ptr is access WMHints;
  type WMHintsList_Ptr is access WMHintsList;

  ------------------------------------------

  --
  -- definition for flags of WMHints
  -- <X11/Xutil.h>
  --
  InputHint    : constant UnsignedLong := 16#0001#;
  StateHint    : constant UnsignedLong := 16#0002#;
  IconPixmapHint   : constant UnsignedLong := 16#0004#;
  IconWindowHint   : constant UnsignedLong := 16#0008#;
  IconPositionHint : constant UnsignedLong := 16#0010#;
  IconMaskHint   : constant UnsignedLong := 16#0020#;
  WindowGroupHint  : constant UnsignedLong := 16#0040#;
  AllHints         : constant UnsignedLong :=
              InputHint + StateHint + IconPixmapHint +
              IconWindowHint + IconPositionHint + IconMaskHint +
              WindowGroupHint;
  --
  -- definitions for initial window state
  --
  WithdrawnState   : constant := 0; -- for windows that are not mapped
  NormalState      : constant := 1; -- most applications want to start this way
  IconicState      : constant := 3; -- application wants to start as an icon
  --
  -- Obsolete states no longer defined by ICCCM
  --
  DontCareState : constant := 0; -- don't know or care
  ZoomState     : constant := 2; -- application wants to start zoomed
  InactiveState : constant := 4; -- application believes it is seldom used;
                   -- some wm's may put it on inactive menu

  ------------------------------------------
  -- Used by the visual utility routines to find
  -- desired visual type from the many visuals a display may support.
  --
  VisualNoMask : constant := 16#000#;
  VisualIDMask  : constant := 16#001#;
  VisualScreenMask : constant := 16#002#;
  VisualDepthMask : constant := 16#004#;
```

```
VisualClassMask : constant := 16#008#;
VisualRedMaskMask : constant := 16#010#;
VisualGreenMaskMask : constant := 16#020#;
VisualBlueMaskMask : constant := 16#040#;
VisualColormapSizeMask : constant := 16#080#;
VisualBitsPerRGBMask : constant := 16#100#;
VisualAllMask : constant := 16#1FF#;

-----------------------------------------
-- This defines a window manager property that clients may use to
-- share shandard color maps of type RGP_COLOR_MAP;
--

type StandardColormap is
  record
    cmap       : Colormap;
    red_max    : UnsignedLong;
    red_mult   : UnsignedLong;
    green_max  : UnsignedLong;
    green_mult : UnsignedLong;
    blue_max   : UnsignedLong;
    blue_mult  : UnsignedLong;
    base_pixel : UnsignedLong;
    visual_id  : VisualId;
    killid     : XId;
  end record;

type StandardColormap_Ptr is access StandardColormap;
type StandardColormapList is array ( Int range <> ) of StandardColormap;
type StandardColormapList_ptr is access StandardColormapList;

--
-- return codes for XReadBitmapFile and XWriteBitmapFile
--
BitmapSuccess: constant := 0;
BitmapOpenFailed : constant := 1;
BitmapFileInvalid : constant := 2;
BitmapNoMemory: constant := 3;

--/*****************************************************************
--*
--* Context Management
--*
--*****************************************************************/
--
-- Associative lookup table return codes */
--
XCSUCCESS  : constant := 0;-- No error
XCNOMEM   : constant := 1;        -- Out of memory
XCNOENT   : constant := 2;   -- No entry in table

type REGION is new Address;

--============================================================
-- Pre-defined X11 Cursors
-- <X11/cursorfont.h>
--
C_num_glyphs  : constant := 154;
C_X_cursor  : constant := 0;
```

```
C_arrow  : constant := 2;
C_based_arrow_down    : constant := 4;
C_based_arrow_up      : constant := 6;
C_boat  : constant := 8;
C_bogosity  : constant := 10;
C_bottom_left_corner  : constant := 12;
C_bottom_right_corner : constant := 14;
C_bottom_side  : constant := 16;
C_bottom_tee  : constant := 18;
C_box_spiral  : constant := 20;
C_center_ptr  : constant := 22;
C_circle  : constant := 24;
C_clock  : constant := 26;
C_coffee_mug  : constant := 28;
C_cross  : constant := 30;
C_cross_reverse  : constant := 32;
C_crosshair  : constant := 34;
C_diamond_cross  : constant := 36;
C_dot   : constant := 38;
C_dotbox  : constant := 40;
C_double_arrow        : constant := 42;
C_draft_large  : constant := 44;
C_draft_small  : constant := 46;
C_draped_box  : constant := 48;
C_exchange  : constant := 50;
C_fleur  : constant := 52;
C_gobbler  : constant := 54;
C_gumby  : constant := 56;
C_hand1  : constant := 58;
C_hand2  : constant := 60;
C_heart  : constant := 62;
C_icon  : constant := 64;
C_iron_cross  : constant := 66;
C_left_ptr  : constant := 68;
C_left_side  : constant := 70;
C_left_tee  : constant := 72;
C_leftbutton  : constant := 74;
C_ll_angle   : constant := 76;
C_lr_angle  : constant := 78;
C_man   : constant := 80;
C_middlebutton  : constant := 82;
C_mouse  : constant := 84;
C_pencil  : constant := 86;
C_pirate  : constant := 88;
C_plus  : constant := 90;
C_question_arrow      : constant := 92;
C_right_ptr  : constant := 94;
C_right_side  : constant := 96;
C_right_tee   : constant := 98;
C_rightbutton  : constant := 100;
C_rtl_logo  : constant := 102;
C_sailboat  : constant := 104;
C_sb_down_arrow  : constant := 106;
C_sb_h_double_arrow  : constant := 108;
C_sb_left_arrow       : constant := 110;
C_sb_right_arrow      : constant := 112;
C_sb_up_arrow  : constant := 114;
C_sb_v_double_arrow   : constant := 116;
C_shuttle  : constant := 118;
```

```
C_sizing  : constant := 120;
C_spider  : constant := 122;
C_spraycan  : constant := 124;
C_star  : constant := 126;
C_target  : constant := 128;
C_tcross  : constant := 130;
C_top_left_arrow  : constant := 132;
C_top_left_corner  : constant := 134;
C_top_right_corner  : constant := 136;
C_top_side  : constant := 138;
C_top_tee  : constant := 140;
C_trek  : constant := 142;
C_ul_angle  : constant := 144;
C_umbrella  : constant := 146;
C_ur_angle  : constant := 148;
C_watch  : constant := 150;
C_xterm  : constant := 152;

--============================================================

--
--  Predefined X Windows Atoms.
--

A_PRIMARY    : constant Atom := 1;
A_SECONDARY  : constant Atom := 2;
A_ARC        : constant Atom := 3;
A_ATOM       : constant Atom := 4;
A_BITMAP     : constant Atom := 5;
A_CARDINAL   : constant Atom := 6;
A_COLORMAP   : constant Atom := 7;
A_CURSOR     : constant Atom := 8;
A_CUT_BUFFER0: constant Atom := 9;
A_CUT_BUFFER1: constant Atom := 10;
A_CUT_BUFFER2: constant Atom := 11;
A_CUT_BUFFER3: constant Atom := 12;
A_CUT_BUFFER4: constant Atom := 13;
A_CUT_BUFFER5: constant Atom := 14;
A_CUT_BUFFER6: constant Atom := 15;
A_CUT_BUFFER7: constant Atom := 16;
A_DRAWABLE   : constant Atom := 17;
A_FONT       : constant Atom := 18;
A_INTEGER     : constant Atom := 19;
A_PIXMAP     : constant Atom := 20;
A_POINT      : constant Atom := 21;
A_RECTANGLE  : constant Atom := 22;
A_RESOURCE_MANAGER: constant Atom := 23;
A_RGB_COLOR_MAP: constant Atom := 24;
A_RGB_BEST_MAP: constant Atom := 25;
A_RGB_BLUE_MAP: constant Atom := 26;
A_RGB_DEFAULT_MAP: constant Atom := 27;
A_RGB_GRAY_MAP: constant Atom := 28;
A_RGB_GREEN_MAP: constant Atom := 29;
A_RGB_RED_MAP: constant Atom := 30;
A_STRING      : constant Atom := 31;
A_VISUALID   : constant Atom := 32;
A_WINDOW             : constant Atom := 33;
A_WM_COMMAND  : constant Atom := 34;
A_WM_HINTS    : constant Atom := 35;
```

```
A_WM_CLIENT_MACHINE: constant Atom := 36;
A_WM_ICON_NAME: constant Atom := 37;
A_WM_ICON_SIZE: constant Atom := 38;
A_WM_NAME    : constant Atom := 39;
A_WM_NORMAL_HINTS: constant Atom := 40;
A_WM_SIZE_HINTS: constant Atom := 41;
A_WM_ZOOM_HINTS: constant Atom := 42;
A_MIN_SPACE  : constant Atom := 43;
A_NORM_SPACE  : constant Atom := 44;
A_MAX_SPACE  : constant Atom := 45;
A_END_SPACE     : constant Atom := 46;
A_SUPERSCRIPT_X: constant Atom := 47;
A_SUPERSCRIPT_Y: constant Atom := 48;
A_SUBSCRIPT_X: constant Atom := 49;
A_SUBSCRIPT_Y: constant Atom := 50;
A_UNDERLINE_POSITION: constant Atom := 51;
A_UNDERLINE_THICKNESS : constant Atom := 52;
A_STRIKEOUT_ASCENT: constant Atom := 53;
A_STRIKEOUT_DESCENT: constant Atom := 54;
A_ITALIC_ANGLE: constant Atom := 55;
A_X_HEIGHT   : constant Atom := 56;
A_QUAD_WIDTH : constant Atom := 57;
A_WEIGHT     : constant Atom := 58;
A_POINT_SIZE : constant Atom := 59;
A_RESOLUTION  : constant Atom := 60;
A_COPYRIGHT   : constant Atom := 61;
A_NOTICE     : constant Atom := 62;
A_FONT_NAME    : constant Atom := 63;
A_FAMILY_NAME: constant Atom := 64;
A_FULL_NAME  : constant Atom := 65;
A_CAP_HEIGHT  : constant Atom := 66;
A_WM_CLASS   : constant Atom := 67;
A_WM_TRANSIENT_FOR: constant Atom := 68;
A_LAST_PREDEFINED: constant Atom := 68;


-------------------------------------------

type GCValues is
  record
    func               : Int;
    plane_mask         : UnsignedLong;
    foreground         : UnsignedLong;
    background         : UnsignedLong;
    line_width         : Int;
    line_style         : Int;
    cap_style          : Int;
    join_style         : Int;
    fill_style         : Int;
    fill_rule          : Int;
    arc_mode           : Int;
    tile               : Pixmap;
    stipple            : Pixmap;
    ts_x_origin        : Int;
    ts_y_origin        : Int;
    font_id            : Font;
    subwindow_mode     : Int;
    graphics_exposures : Bool;
    clip_x_origin      : Int;
```

```
    clip_y_origin      : Int;
    clip_mask          : Pixmap;
    dash_offset        : Int;
    dashes             : Character;
  end record;

type GCValuesList is array ( Int range <> ) of GCValues;
type GCValues_Ptr is access GCValues;
type GCValuesList_Ptr is access GCValuesList;

type XGC is
  record
    xext_data : Address;
    gid       : GContext;
    rects     : Bool;
    dashes    : Bool;
    dirty     : UnsignedLong;
    values    : GCValues;
  end record;

type GC is new Address;
GC_Size : constant := BasicTypes.Address_Size;

-------------------------------------------

type TextProperty is
  record
    value     : Address;
    encoding  : Atom;
    format    : Int;
    nitems    : Int;
  end record;

type TextPropertyList is array ( Int range <> ) of TextProperty;
type TextProperty_Ptr is access TextProperty;
type TextPropertyList_Ptr is access TextPropertyList;

-------------------------------------------
-- Data structure for “image” data,
-- used by image manipulation routines.
--

type ImageFunctions is
  record
    create_image  : Address;
    destroy_image : Address;
    get_pixel     : Address;
    put_pixel     : Address;
    sub_image     : Address;
    add_pixel     : Address;
  end record;

-------------------------------------------

type Image is
  record
    width            : Int;-- Image Width
    height        : Int;  -- Image Height
    xoffset          : Int;-- Number of Pixels off set in X
```

```
    format         : Int;-- XYBitmap, XYPixmap, ZPixmap
    data          : Address;-- Pointer to Image Data
    byte_order       : Int;   -- LSBFirst, MSBFirst
    bitmap_unit      : Int;         -- Quant. of scanline 8, 16, 32
    bitmap_bit_order : Int;        -- LSBFirst, MSBFirst
    bitmap_pad       : Int;         -- 8, 16, 32 either XY or ZPixmap
    depth       : Int;           -- depth of image */
    bytes_per_line   : Int;        -- accelarator to next line */
    bits_per_pixel   : Int;        -- bits per pixel (ZPixmap) */
    red_mask       : UnsignedLong;
    green_mask       : UnsignedLong;
    blue_mask        : UnsignedLong;
    obdata       : Address;  -- hook for the object routines
    funcs            : ImageFunctions;
  end record;

type ImageList is array ( Int range <> ) of Image;
type Image_Ptr is access Image;
type ImageList_Ptr is access ImageList;

-------------------------------------------

type Point is
  record
    x  : Short;
    y  : Short;
  end record;

type PointList is array ( Int range <> ) of Point;
type Point_Ptr is access Point;
type PointList_Ptr is access PointList;

-------------------------------------------

type Rectangle is
  record
    x      : Short;
    y      : Short;
    width  : UnsignedShort;
    height : UnsignedShort;
  end record;

type RectangleList is array ( Int range <> ) of Rectangle;
type Rectangle_Ptr is access Rectangle;
type RectangleList_Ptr is access RectangleList;

-------------------------------------------

type Arc is
  record
    x      : Short;
    y      : Short;
    width  : UnsignedShort;
    height : UnsignedShort;
    angle1 : Short;
    angle2 : Short;
  end record;

type ArcList is array ( Int range <> ) of Arc;
```

```
type Arc_Ptr is access Arc;
type ArcList_Ptr is access ArcList;

-------------------------------------------

type Color is
  record
    pixel : UnsignedLong;
    red   : UnsignedShort;
    green : UnsignedShort;
    blue  : UnsignedShort;
    flags : UnsignedChar;
    pad   : UnsignedChar;
  end record;

type ColorList is array ( Int range <> ) of Color;
type Color_Ptr is access Color;
type ColorList_Ptr is access ColorList;

-------------------------------------------

type Segment is
  record
    x1  : Short;
    y1  : Short;
    x2  : Short;
    y2  : Short;
  end record;

type SegmentList is array ( Int range <> ) of Segment;
type Segment_Ptr is access Segment;
type SegmentList_Ptr is access SegmentList;

-------------------------------------------

--
-- Visual structure; contains information about
-- colomapping possible.
--

type Visual is
  record
    ext_data      : Pointer;
    visual_id     : VisualId;
    class         : Int;
    red_mask      : UnsignedLong;
    green_mask    : UnsignedLong;
    blue_mask     : UnsignedLong;
    bits_per_rgb  : Int;
    map_entries   : Int;
  end record;

type Visual_Ptr is access Visual;
type VisualList is array ( Int range <> ) of Visual;
type VisualList_Ptr is access VisualList;

-------------------------------------------

type VisualInfo is
```

```
    record
      vis           : Visual_Ptr;
      visual_id     : VisualId;
      screen        : Int;
      depth         : Int;
      class         : Int;
      red_mask      : UnsignedLong;
      green_mask    : UnsignedLong;
      blue_mask     : UnsignedLong;
      colormap_size : Int;
      bits_per_rgb  : Int;
    end record;

  type VisualInfo_Ptr is access Visual;
  type VisualInfoList is array ( Int range <> ) of VisualInfo;
  type VisualInfoList_Ptr is access VisualInfoList;

  -------------------------------------------
  --
  -- Depth structure; containns information for
  -- each possible depth.
  --

  type Depth is
    record
      depth       : Int;
      nvisuals    : Int;
      visual_list : Address;
    end record;

  type DepthList is array ( Int range <> ) of Depth;
  type Depth_Ptr is access Depth;
  type DepthList_Ptr is access DepthList;

  -------------------------------------------
  --
  -- Information about the screen.
  --
  type Screen is
    record
      ext_data        : Address;
      display_id      : Display;
      root            : Window;
      width           : Int;
      height          : Int;
      mwidth          : Int;
      mheight         : Int;
      ndepths         : Int;
      depths_list     : Address;
      root_depth      : Int;
      root_visual     : Visual_Ptr;
      default_gc      : GC;
      cmap            : Colormap;
      white_pixel     : UnsignedLong;
      black_pixel     : UnsignedLong;
      max_maps        : Int;
      min_maps        : Int;
      backing_store   : Int;
      save_unders     : Bool;
```

```
    root_input_mask : Long;
  end record;

type ScreenList is array ( Int range <> ) of Screen;
type Screen_Ptr is new Address;
type ScreenList_Ptr is access ScreenList;


-------------------------------------------

type SetWindowAttributes is
  record
    background_pixmap     : Pixmap;
    background_pixel      : UnsignedLong;
    border_pixmap         : Pixmap;
    border_pixel          : UnsignedLong;
    bit_gravity           : Int;
    win_gravity           : Int;
    backing_store         : Int;
    backing_planes        : UnsignedLong;
    backing_pixel         : UnsignedLong;
    save_under            : Bool;
    event_mask            : Long;
    do_not_propagate_mask : Long;
    override_redirect     : Bool;
    cmap                  : Colormap;
    window_cursor         : Cursor;
  end record;

type SetWindowAttributesList is array ( Int range <> ) of SetWindowAttributes;
type SetWindowAttributes_Ptr is access SetWindowAttributes;
type SetWindowAttributesList_Ptr is access SetWindowAttributesList;

-------------------------------------------

type WindowAttributes is
  record
    x                     : Int;
    y      : Int;
    width                 : Int;
    height                : Int;
    border_width    : Int;
    depth    : Int;
    vis       : Visual_Ptr;
    root    : Window;
    class    : Int;
    bit_gravity           : Int;
    win_gravity           : Int;
    backing_store         : Int;
    backing_planes        : UnsignedLong;
    backing_pixel         : UnsignedLong;
    save_under            : Bool;
    cmap                  : Colormap;
    map_installed    : Bool;
    map_state    : Int;
    all_event_masks       : Long;
    your_event_masks      : Long;
    do_not_propagate_mask : Long;
    override_redirect     : Bool;
```

```
    scr      : Screen_Ptr;
  end record;

type WindowAttributesList is array ( Int range <> ) of WindowAttributes;
type WindowAttributes_Ptr is access WindowAttributes;
type WindowAttributesList_Ptr is access WindowAttributesList;

------------------------------------------
--
--  Event Record for all Xlib events.
--  This record is declared as a union, the size of the structure
--  is the size of the largest element in the union.
--

type ButtonEvent is
  record
    root        : Window;
    subwindow   : Window;
    event_time  : Time;
    x           : Int;
    y           : Int;
    x_root      : Int;
    y_root      : Int;
    state       : UnsignedInt;
    button      : UnsignedInt;
    same_screen : Bool;
  end record;
type ButtonEvent_Ptr is access ButtonEvent;

subtype ButtonPressedEvent is ButtonEvent;
type ButtonPressedEvent_Ptr is access ButtonPressedEvent;

subtype ButtonReleasedEvent is ButtonEvent;
type ButtonReleaasedEvent_ptr is access ButtonReleasedEvent;

type CirculateEvent is
  record
    win       : Window;
    place     : Int;
  end record;
type CirculateEvent_Ptr is access CirculateEvent;

type CirculateRequestEvent is
  record
    win     : Window;
    place   : Int;
  end record;
type CirculateRequestEvent_Ptr is access CirculateRequestEvent;

type ClientMessageEvent is
  record
    message_type : Atom;
    format       : Int;
    s            : UnsignedCharList( 1 .. 20 );
  end record;
type ClientMessageEvent_Ptr is access ClientMessageEvent;

type ColormapEvent is
  record
```

```
    cmap     : Colormap;
    new_cmap : Bool;
    state    : Int;
  end record;
type ColormapEvent_Ptr is access ColormapEvent;

type ConfigureEvent is
  record
    win              : Window;
    x                : Int;
    y                : Int;
    width            : Int;
    height           : Int;
    border_width     : Int;
    above            : Window;
    override_redirect : Bool;
  end record;
type ConfigureEvent_Ptr is access ConfigureEvent;

type ConfigureRequestEvent is
  record
    win          : Window;
    x            : Int;
    y            : Int;
    width        : Int;
    height       : Int;
    border_width : Int;
    above        : Window;
    detail       : Int;
    value_mask   : UnsignedLong;
  end record;
type ConfigureRequestEvent_Ptr is access ConfigureRequestEvent;

type CreateWindowEvent is
  record
    win              : Window;
    x                : Int;
    y                : Int;
    width            : Int;
    height           : Int;
    border_width     : Int;
    override_redirect : Bool;
  end record;
type CreateWindowEvent_Ptr is access CreateWindowEvent;

type DestroyWindowEvent is
  record
    win       : Window;
  end record;
type DestroyWindowEvent_Ptr is access DestroyWindowEvent;

type CrossingEvent is
  record
    root        : Window;
    subwindow   : Window;
    event_time  : Time;
    x           : Int;
    y           : Int;
    x_root      : Int;
```

```
    y_root      : Int;
    mode        : Int;
    detail      : Int;
    same_screen : Bool;
    focus       : Bool;
    state       : UnsignedInt;
  end record;

subtype EnterWindowEvent is CrossingEvent;
subtype LeaveWindowEvent is CrossingEvent;
type CrossingEvent_Ptr is access CrossingEvent;

type ErrorEvent is
  record
    event_type   : Int;               -- Normally, XEvent discriminant
    dpy           : Display;        -- Display the event was read from
    resourceid   : XID;               -- resource id
    serial       : Long;              -- serial number of failed request
    error_code   : UnsignedChar;       -- error code of failed request
    request_code : UnsignedChar;       -- Major op-code of failed request
    minor_code   : UnsignedChar;       -- Minor op-code of failed request
  end record;
type ErrorEvent_Ptr is access ErrorEvent;

type ExposeEvent is
  record
    x        : Int;
    y        : Int;
    width   : Int;
    height  : Int;
    count   : Int;
  end record;
type ExposeEvent_Ptr is access ExposeEvent;

type FocusChangeEvent is
  record
    mode   : Int;
    detail : Int;
  end record;

subtype FocusInEvent is FocusChangeEvent;
subtype FocusOutEvent is FocusChangeEvent;
type FocusChangeEvent_Ptr is access FocusChangeEvent;

type GraphicsExposeEvent is
  record
    x          : Int;
    y          : Int;
    width      : Int;
    height     : Int;
    count      : Int;
    major_code : Int;
    minor_code : Int;
  end record;
type GraphicsExposeEvent_Ptr is access GraphicsExposeEvent;

type GravityNotifyEvent is
  record
    win  : Window;
```

```
    x    : Int;
    y    : Int;
  end record;
type GravityNotifyEvent_Ptr is access GravityNotifyEvent;

type KeymapEvent is
  record
    key_vector : String( 1 .. 32 );
  end record;
type KeymapEvent_Ptr is access KeymapEvent;

type KeyEvent is
  record
    root        : Window;
    subwindow   : Window;
    event_time  : Time;
    x           : Int;
    y           : Int;
    x_root      : Int;
    y_root      : Int;
    state       : UnsignedInt;
    keycode     : UnsignedInt;
    same_screen : Bool;
  end record;

subtype KeyPressedEvent is KeyEvent;
subtype KeyReleasedEvent is KeyEvent;
type KeyEvent_Ptr is access KeyEvent;

type MapEvent is
  record
    win               : Window;
    override_redirect : Bool;
  end record;
type MapEvent_Ptr is access MapEvent;

type MappingEvent is
  record
    request       : Int;
    first_keycode : Int;
    count         : Int;
  end record;
type MappingEvent_Ptr is access MappingEvent;

type MapRequestEvent is
  record
    win : Window;
  end record;
type MapRequestEvent_Ptr is access MapRequestEvent;

type MotionEvent is
  record
    root          : Window;
    subwindow     : Window;
    event_time    : Time;
    x             : Int;
    y             : Int;
    x_root        : Int;
    y_root        : Int;
```

```
    state         : UnsignedInt;
    is_hint       : UnsignedChar;
    same_screen   : Bool;
  end record;

subtype PointerMovedEvent is MotionEvent;
type MotionEvent_Ptr is access MotionEvent;

type NoExposeEvent is
  record
    major_code : Int;
    minor_code : Int;
  end record;
type NoExposeEvent_Ptr is access NoExposeEvent;

type PropertyEvent is
  record
    prop_atom  : Atom;
    event_time : Time;
    state      : Int;
  end record;
type PropertyEvent_Ptr is access PropertyEvent;

type ReparentEvent is
  record
    win              : Window;
    parent           : Window;
    x                : Int;
    y                : Int;
    override_redirect : Bool;
  end record;
type ReparentEvent_Ptr is access ReparentEvent;

type ResizeRequestEvent is
  record
    width   : Int;
    height  : Int;
  end record;
type ResizeRequestEvent_Ptr is access ResizeRequestEvent;

type SelectionClearEvent is
  record
    selection  : Atom;
    event_time : Time;
  end record;
type SelectionClearEvent_Ptr is access SelectionClearEvent;

type SelectionEvent is
  record
    selection  : Atom;
    target     : Atom;
    property   : Atom;
    event_time : Time;
  end record;
type SelectionEvent_Ptr is access SelectionEvent;

type SelectionRequestEvent is
  record
    requestor  : Window;
```

```
    selection  : Atom;
    target     : Atom;
    property   : Atom;
    event_time : Time;
  end record;
type SelectionRequestEvent_Ptr is access SelectionRequestEvent;

type UnMapEvent is
  record
    win             : Window;
    from_configure  : Bool;
  end record;
type UnMapEvent_Ptr is access UnMapEvent;

type VisibilityEvent is
  record
    state  : Int;
  end record;
type VisibilityEvent_Ptr is access VisibilityEvent;

type GenericEvent is
  record
    field1  : Int;
    field2  : Int;
    field3  : Int;
    field4  : Int;
    field5  : Int;
    field6  : Int;
    field7  : Int;
    field8  : Int;
    field9  : Int;
    field10 : Int;
    field11 : Int;
    field12 : Int;
    field13 : Int;
    field14 : Int;
    field15 : Int;
    field16 : Int;
    field17 : Int;
    field18 : Int;
    field19 : Int;
  end record;
type GenericEvent_Ptr is access GenericEvent;

type Event( event_type : Int := LastEvent ) is
  record
    serial_no  : UnsignedLong;
    send_event : Bool;
    dpy        : Display;
    win        : Window;
    case event_type is
      when NoEvent =>
     null;
      when ButtonPress | ButtonRelease =>
     xbutton : ButtonEvent;
   when CirculateNotify =>
     xcirculate : CirculateEvent;
   when CirculateRequest =>
     xcirculaterequest : CirculateRequestEvent;
```

```
    when ClientMessage =>
      xclient : ClientMessageEvent;
    when ColormapNotify =>
      xcolormap : ColormapEvent;
    when ConfigureNotify =>
      xconfigure : ConfigureEvent;
    when ConfigureRequest =>
      xconfigurerequest : ConfigureRequestEvent;
    when CreateNotify =>
      xcreatewindow : CreateWindowEvent;
    when DestroyNotify =>
      xdestroywindow : DestroyWindowEvent;
    when EnterNotify | LeaveNotify =>
      xcrossing : CrossingEvent;
       when Expose =>
      xexpose : ExposeEvent;
       when FocusIn | FocusOut =>
          xfocus : FocusChangeEvent;
       when NoExpose =>
          xnoexpose : NoExposeEvent;
       when GraphicsExpose =>
          xgraphicsexpose : GraphicsExposeEvent;
       when GravityNotify =>
          xgravity : GravityNotifyEvent;
       when KeymapNotify =>
          xkeymap : KeymapEvent;
       when KeyPress | KeyRelease =>
          xkey : KeyEvent;
       when MapNotify =>
          xmap : MapEvent;
       when UnMapNotify =>
          xunmap : UnMapEvent;
       when MappingNotify =>
          xmapping : MappingEvent;
       when MapRequest =>
          xmaprequest : MapRequestEvent;
       when MotionNotify =>
          xmotion : MotionEvent;
       when PropertyNotify =>
          xproperty : PropertyEvent;
       when ReparentNotify =>
          xreparent : ReparentEvent;
       when ResizeRequest =>
          xresizerequest : ResizeRequestEvent;
       when SelectionClear =>
          xselectionclear : SelectionClearEvent;
       when SelectionNotify =>
          xselection : SelectionEvent;
       when  SelectionRequest =>
          xselectionrequest : SelectionRequestEvent;
       when VisibilityNotify =>
          xvisibility : VisibilityEvent;
       when others =>
      xgeneric : GenericEvent;
     end case;
   end record;
 type Event_Ptr is access Event;

 -------------------------------------------
```

```
type AnyEvent is
  record
    event_type : Int;
    serial_no  : UnsignedLong;
    send_event : Bool;
    dpy        : Display;
    win        : Window;
    field1  : Int;
    field2  : Int;
    field3  : Int;
    field4  : Int;
    field5  : Int;
    field6  : Int;
    field7  : Int;
    field8  : Int;
    field9  : Int;
    field10 : Int;
    field11 : Int;
    field12 : Int;
    field13 : Int;
    field14 : Int;
    field15 : Int;
    field16 : Int;
    field17 : Int;
    field18 : Int;
    field19 : Int;
  end record;

-------------------------------------------

type CharStruct is
  record
    lbearing   : Short;
    rbearing   : Short;
    width      : Short;
    ascent     : Short;
    descent    : Short;
    attributes : UnsignedShort;
  end record;

type CharStructList is array ( Int range <> ) of CharStruct;
type CharStruct_Ptr is access CharStruct;
type CharStructList_Ptr is access CharStructList;

-------------------------------------------

type FontStruct is
  record
    ext_data          : Address;
    fid               : Font;
    direction         : UnsignedInt;
    min_char_or_byte2 : UnsignedInt;
    max_char_or_byte2 : UnsignedInt;
    min_byte1         : UnsignedInt;
    max_byte1         : UnsignedInt;
    all_chars_exist   : Bool;
    default_char      : UnsignedInt;
    n_properties      : Int;
```

```
    props             : Address;
    min_bounds        : CharStruct;
    max_bounds        : CharStruct;
    per_char          : Address;
    ascent     : Int;
    descent    : Int;
  end record;

type FontStruct_Ptr is access FontStruct;
type FontStructList is array ( Int range <> ) of FontStruct;
type FontStructList_Ptr is access FontStructList;

-------------------------------------------

type TextItem is
  record
    chars   : Address;
    nchars  : Int;
    delt    : Int;
    font_id : Font;
  end record;

type TextItemList is array ( Int range <> ) of TextItem;
type TextItem_Ptr is access TextItem;
type TextItemList_Ptr is access TextItemList;

type TextItem16 is
  record
    chars   : Address;
    nchars  : Int;
    delt    : Int;
    font_id : Font;
  end record;

type TextItem16List is array ( Int range <> ) of TextItem16;
type TextItem16_Ptr is access TextItem16;
type TextItem16List_Ptr is access TextItem16List;

-------------------------------------------

type SpecialCharacter is (
  UP_ARROW,
  DOWN_ARROW,
  LEFT_ARROW,
  RIGHT_ARROW,
  BACK_SPACE,
  LINEFEED,
  CARRIAGE_RETURN,
  INSERT_CHAR,
  INSERT_LINE,
  DELETE_CHAR,
  DELETE_LINE,
  CLEAR_LINE,
  ESCAPE,
  FK_1, FK_2,  FK_3,  FK_4,
  FK_5, FK_6,  FK_7,  FK_8,
  FK_9, FK_10, FK_11, FK_12);

-------------------------------------------
```

```
-- Some Font Names
--
F_6X10     : constant STRING := “6x10” & ASCII.NUL;
F_6X12     : constant STRING := “6x12” & ASCII.NUL;
F_6X13     : constant STRING := “6x13” & ASCII.NUL;
F_8X13     : constant STRING := “8x13” & ASCII.NUL;
F_8X13BOLD : constant STRING := “8x13bold” & ASCII.NUL;
F_FR3_25   : constant STRING := “fr3-25” & ASCII.NUL;
F_FR2_25   : constant STRING := “fr2-25” & ASCII.NUL;
F_VRI_40   : constant STRING := “vri-40” & ASCII.NUL;
F_FRI_33   : constant STRING := “fri-33” & ASCII.NUL;
F_FRI1_25  : constant STRING := “fri1-25” & ASCII.NUL;
F_FR1_25   : constant STRING := “fr1-25” & ASCII.NUL;
F_FR_33    : constant STRING := “fr-33” & ASCII.NUL;
F_FGI1_25  : constant STRING := “fgi1-25” & ASCII.NUL;
F_FGS_22   : constant STRING := “fgs-22” & ASCII.NUL;
F_FQXB_25  : constant STRING := “fqxb-25” & ASCII.NUL;
F_FR_25    : constant STRING := “fr-25” & ASCII.NUL;
F_FG_30    : constant STRING := “fg-30” & ASCII.NUL;
F_FG1_25   : constant STRING := “fg1-25” & ASCII.NUL;
F_FGB_13   : constant STRING := “fgb-13” & ASCII.NUL;
F_FGB_25   : constant STRING := “fgb-25” & ASCII.NUL;
F_FGB1_25  : constant STRING := “fgb1-25” & ASCII.NUL;
F_FGB1_30  : constant STRING := “fgb1-30” & ASCII.NUL;
F_FGI_20   : constant STRING := “fgi-20” & ASCII.NUL;
F_FG_25    : constant STRING := “fg-25” & ASCII.NUL;
F_FCOR_20  : constant STRING := “fcor-20” & ASCII.NUL;
F_FG_13    : constant STRING := “fg-13” & ASCII.NUL;
F_FG_16    : constant STRING := “fg-16” & ASCII.NUL;
F_FG_18    : constant STRING := “fg-18” & ASCII.NUL;
F_FG_20    : constant STRING := “fg-20” & ASCII.NUL;
F_FG_22    : constant STRING := “fg-22” & ASCII.NUL;
F_IPA_S25  : constant STRING := “ipa-s25” & ASCII.NUL;
F_MET25    : constant STRING := “met25” & ASCII.NUL;
F_R14      : constant STRING := “r14” & ASCII.NUL;
F_SANS12   : constant STRING := “sans12” & ASCII.NUL;
F_SANSB12  : constant STRING := “sansb12” & ASCII.NUL;
F_SANSI12  : constant STRING := “sansi12” & ASCII.NUL;
F_SERIF10  : constant STRING := “serif10” & ASCII.NUL;
F_SERIF12  : constant STRING := “serif12” & ASCII.NUL;
F_SERIFB10 : constant STRING := “serifb10” & ASCII.NUL;
F_SERIFB12 : constant STRING := “serifb12” & ASCII.NUL;
F_SERIFI10 : constant STRING := “serifi10” & ASCII.NUL;
F_SERIFI12 : constant STRING := “serifi12” & ASCII.NUL;
F_SUB      : constant STRING := “sub” & ASCII.NUL;
F_SUBSUB   : constant STRING := “subsub” & ASCII.NUL;
F_SUP      : constant STRING := “sup” & ASCII.NUL;
F_SUPSUP   : constant STRING := “supsup” & ASCII.NUL;
F_SWD_S30  : constant STRING := “swd-s30” & ASCII.NUL;
F_VBEE_36  : constant STRING := “vbee-36” & ASCII.NUL;
F_VCTL_25  : constant STRING := “vctl-25” & ASCII.NUL;
F_VG_13    : constant STRING := “vg-13” & ASCII.NUL;
F_VG_20    : constant STRING := “vg-20” & ASCII.NUL;
F_VG_25    : constant STRING := “vg-25” & ASCII.NUL;
F_VG_31    : constant STRING := “vg-31” & ASCII.NUL;
F_VGB_25   : constant STRING := “vgb-25” & ASCII.NUL;
F_VGB_31   : constant STRING := “vgb-31” & ASCII.NUL;
F_VGBC_25  : constant STRING := “vgbc-25” & ASCII.NUL;
F_VGH_25   : constant STRING := “vgh-25” & ASCII.NUL;
```

```
  F_VGI_20   : constant STRING := “vgi-20” & ASCII.NUL;
  F_VGI_25   : constant STRING := “vgi-25” & ASCII.NUL;
  F_VGI_31   : constant STRING := “vgi-31” & ASCII.NUL;
  F_VGL_40   : constant STRING := “vgl-40” & ASCII.NUL;
  F_VGVB_31  : constant STRING := “vgvb-31” & ASCII.NUL;
  F_VMIC_25  : constant STRING := “vmic-25” & ASCII.NUL;
  F_VR_20    : constant STRING := “vr-20” & ASCII.NUL;
  F_VR_25    : constant STRING := “vr-25” & ASCII.NUL;
  F_VR_27    : constant STRING := “vr-27” & ASCII.NUL;
  F_VR_30    : constant STRING := “vr-30” & ASCII.NUL;
  F_VR_31    : constant STRING := “vr-31” & ASCII.NUL;
  F_VR_40    : constant STRING := “vr-40” & ASCII.NUL;
  F_VRB_25   : constant STRING := “vrb-25” & ASCII.NUL;
  F_VRB_30   : constant STRING := “vrb-30” & ASCII.NUL;
  F_VRB_31   : constant STRING := “vrb-31” & ASCII.NUL;
  F_VRB_35   : constant STRING := “vrb-35” & ASCII.NUL;
  F_VRB_37   : constant STRING := “vrb-37” & ASCII.NUL;
  F_VRI_25   : constant STRING := “vri-25” & ASCII.NUL;
  F_VRI_30   : constant STRING := “vri-30” & ASCII.NUL;
  F_VRI_31   : constant STRING := “vri-31” & ASCII.NUL;
  F_9X15     : constant STRING := “9x15” & ASCII.NUL;
  F_VTBOLD   : constant STRING := “vtbold” & ASCII.NUL;
  F_VTSINGLE : constant STRING := “vtsingle” & ASCII.NUL;
  F_VXMS_37  : constant STRING := “vxms-37” & ASCII.NUL;
  F_VXMS_43  : constant STRING := “vxms-43” & ASCII.NUL;
  F_XIF_S25  : constant STRING := “xif-s25” & ASCII.NUL;

  --
  --  CopyFromParent must be defined as a function so that it can
  --  take on several different types.
  --
  function CopyFromParent return Int;
  function CopyFromParent return Visual_Ptr;

-------------------------------------------------
-- package XLibR5 data types
--
  subtype ICCEncodingStyle is Int;

  StringStyle : constant := 0;
  CompoundTextStyle : constant := 1;
  TextStyle : constant := 2;
  StdICCTextStyle : constant := 3;

  subtype FontSet is Pointer;

  type FontSetExtents is
    record
      max_ink_extent     : Rectangle;
      max_logical_extent : Rectangle;
    end record;

  for FontSetExtents use
    record
      max_ink_extent     at 0 range 0 .. Rectangle_Size - 1;
      max_logical_extent at Rectangle_Size / 8 range 0 .. Rectangle_Size - 1;
    end record;

  type FontSetExtentsList is array ( Int range <> ) of FontSetExtents;
```

```
  type FontSetExtents_Ptr is access FontSetExtents;
  type FontSetExtentsList_Ptr is access FontSetExtentsList;

-------------------------------------------------
-- package XMuR5 data types
--
  type WidgetNode is
    record
      label             : Xt.Pointer;
      widget_class_ptr  : Xt.Pointer;
      superclass        : Xt.Pointer;
      children          : Xt.Pointer;
      siblings          : Xt.Pointer;
      lowered_label     : Xt.Pointer;
      lowered_classname : Xt.Pointer;
      have_resources    : Xlib.Bool;
      resources         : Xt.Pointer;
      constraints       : Xt.Pointer;
      constraintwn      : Xt.Pointer;
      nconstraints      : Xt.Cardinal;
      data              : Xt.Pointer;
    end record;

  type WidgetNode_Ptr is access WidgetNode;
  type WidgetNodeList is array ( Xlib.Int range <> ) of WidgetNode;
  type WidgetNodeList_Ptr is access WidgetNodeList;

-------------------------------------------------
-- package Xcms data types
--
  subtype Float is Xlib.Double;
  subtype ColorFormat is Xlib.UnsignedLong;
  subtype CCC is Xlib.Pointer;

  type BooleanList is array ( Xlib.Int range <> ) of Boolean;

  UndefinedFormat : constant ColorFormat := 0;
  CIEXYZFormat    : constant ColorFormat := 1;
  CIEuvYFormat    : constant ColorFormat := 2;
  CIExyYFormat    : constant ColorFormat := 3;
  CIELabFormat    : constant ColorFormat := 4;
  CIELuvFormat    : constant ColorFormat := 5;
  TekHVCFormat    : constant ColorFormat := 6;
  RGBFormat       : constant ColorFormat := Xlib.Int'First;
  RGBiFormat      : constant ColorFormat := Xlib.Int'First + 1;

  --
  --   Xcms Status Return values
  --

  Failure : constant  := 0;
  Success : constant  := 1;
  SuccessWithCompress : constant  := 2;

  type RGB is
    record
      red   : Xlib.UnsignedShort;
      green : Xlib.UnsignedShort;
      blue  : Xlib.UnsignedShort;
```

```
    end record;

  type RGBi is
    record
      red   : Float;
      green : Float;
      blue  : Float;
    end record;

  type CIEXYZ is
    record
      X  : Float;
      Y  : Float;
      Z  : Float;
    end record;

  type CIEuvY is
    record
      u_prime : Float;
      v_prime : Float;
      Y       : Float;
    end record;

  type CIExyY is
    record
      X  : Float;
      Y  : Float;
      YY : Float;
    end record;

  type CIELab is
    record
      L_star  : Float;
      a_star  : Float;
      b_star  : Float;
    end record;

  type CIELuv is
    record
      L_star  : Float;
      u_star  : Float;
      v_star  : Float;
    end record;

  type TekHVC is
    record
      H  : Float;
      V  : Float;
      C  : Float;
    end record;

  type Pad is
    record
      pad0  : Float;
      pad1  : Float;
      pad2  : Float;
      pad3  : Float;
    end record;
```

```
  type Color( format : ColorFormat := RGBFormat ) is
    record
      pixel : Xlib.UnsignedLong;
      case format is
        when UndefinedFormat =>
       pad_color    : Pad;
        when CIEXYZFormat =>
       ciexyz_color : CIEXYZ;
        when CIEuvYFormat =>
       cieuvy_color : CIEuvY;
        when CIExyYFormat =>
       ciexyy_color : CIExyY;
        when CIELabFormat =>
       cielab_color : CIELab;
        when CIELuvFormat =>
       cieluv_color : CIELuv;
        when TekHVCFormat =>
       tekhvc_color : TekHVC;
        when RGBFormat =>
       rgb_color    : RGB;
        when RGBiFormat =>
       rgbi_color   : RGBi;
     when others =>
       pad_clr      : Pad;
      end case;
    end record;

  type Color_Ptr is access Color;
  type ColorList is array ( Xlib.Int range <> ) of Color;
  type ColorList_Ptr is access ColorList;

------------------------------------------------
-- package Xwc data types
--
  subtype FontSet is XlibR5.FontSet;
  subtype IC      is XlibR5.IC;
  subtype ICCEncodingStyle is XlibR5.ICCEncodingStyle;

  subtype wchar_t is Xlib.UnsignedShort;

  type wchar_t_String is array ( Xlib.Int range <> ) of wchar_t;
  type wchar_t_String_Ptr is access wchar_t_String;
  type wchar_t_StringList is array ( Xlib.Int range <> ) of wchar_t_String_Ptr;
  type wchar_t_StringList_Ptr is access wchar_t_StringList;

  type TextItem is
    record
      chars    : Xlib.Pointer;
      nchars   : Xlib.Int;
      dlta     : Xlib.Int;
      font_set : FontSet;
    end record;

  type TextItem_Ptr is access TextItem;
  type TextItemList is array ( Xlib.Int range <> ) of TextItem;
  type TextItemList_Ptr is access TextItemList;
```

Relevant portions of package Xt

```
  subtype Char                  is Xlib.Char;
  subtype Char_Ptr              is Xlib.Char_Ptr;
  subtype CharList              is Xlib.CharList;
  subtype CharList_Ptr          is Xlib.CharList_Ptr;
  subtype Double        is Xlib.Double;
  subtype Double_Ptr        is Xlib.Double_Ptr;
  subtype DoubleList        is Xlib.DoubleList;
  subtype DoubleList_Ptr        is Xlib.DoubleList_Ptr;
  subtype Real                  is Xlib.Real;
  subtype Real_Ptr              is Xlib.Real_Ptr;
  subtype RealList              is Xlib.RealList;
  subtype RealList_Ptr          is Xlib.RealList_Ptr;
  subtype Int            is Xlib.Int;
  subtype IntList               is Xlib.IntList;
  subtype Int_Ptr               is Xlib.Int_Ptr;
  subtype IntList_Ptr           is Xlib.IntList_Ptr;
  subtype Long            is Xlib.Long;
  subtype Long_Ptr           is Xlib.Long_Ptr;
  subtype LongList        is Xlib.LongList;
  subtype LongList_Ptr        is Xlib.LongList_Ptr;
  subtype Pointer               is Xlib.Pointer;
  subtype Pointer_Ptr           is Xlib.Pointer_Ptr;
  subtype PointerList           is Xlib.PointerList;
  subtype PointerList_Ptr       is Xlib.PointerList_Ptr;
  subtype Short         is Xlib.Short;
  subtype Short_Ptr         is Xlib.Short_Ptr;
  subtype ShortList         is Xlib.ShortList;
  subtype ShortList_Ptr         is Xlib.ShortList_Ptr;
  subtype StringList            is Xlib.StringList;
  subtype StringList_Ptr        is Xlib.StringList_Ptr;
  subtype String_Ptr            is Xlib.String_Ptr;
  subtype NullString            is Xlib.NullString;
  subtype UnsignedChar          is Xlib.UnsignedChar;
  subtype UnsignedChar_Ptr      is Xlib.UnsignedChar_Ptr;
  subtype UnsignedCharList      is Xlib.UnsignedCharList;
  subtype UnsignedCharList_Ptr  is Xlib.UnsignedCharList_Ptr;
  subtype UnsignedInt           is Xlib.UnsignedInt;
  subtype UnsignedIntList       is Xlib.UnsignedIntList;
  subtype UnsignedLong          is Xlib.UnsignedLong;
  subtype UnsignedLong_Ptr      is Xlib.UnsignedLong_Ptr;
  subtype UnsignedLongList      is Xlib.UnsignedLongList;
  subtype UnsignedLongList_Ptr  is Xlib.UnsignedLongList_Ptr;
  subtype UnsignedShort         is Xlib.UnsignedShort;
  subtype UnsignedShort_Ptr     is Xlib.UnsignedShort_Ptr;
  subtype UnsignedShortList     is Xlib.UnsignedShortList;
  subtype UnsignedShortList_Ptr is Xlib.UnsignedShortList_Ptr;

  function "="( left, right : Xlib.Address ) return Boolean renames Xlib."=";

  C_String_Size : constant := Xlib.C_String_Size;
  Pointer_Size  : constant := Xlib.Pointer_Size;
  Address_Size  : constant := Xlib.Address_Size;

  subtype XtBoolean is Tiny_Integer;

  subtype CacheRef       is Pointer;
```

```
subtype FontStruct_Ptr is Xlib.FontStruct_Ptr;
subtype Image          is Xlib.Image;
subtype Event_Ptr  is Xlib.Event_Ptr;


XTrue   : constant := Xlib.XTrue;
XFalse   : constant := Xlib.XFalse;

XtFalse  : constant XtBoolean := 0;
XtTrue   : constant XtBoolean := 1;

XNULL    : constant Pointer := Xlib.XNULL;
XtNULL   : constant Pointer := XNULL;

-------------------------------------------
-- <X11/Intrinsic.h>
--
subtype Widget           is Pointer;

type Accelerators        is new Pointer;
type AppContext          is new Pointer;
type ArgVal              is new Long;
type CacheType           is new Int;
subtype Cardinal         is Int;
type Dimension           is new UnsignedShort;
type Enum                is new UnsignedChar;
type Modifiers           is new UnsignedInt;
type Pixel               is new UnsignedLong;
type Position   is new Short; -- System Dependent.
type VersionType         is new UnsignedLong;
type WidgetClass         is new Pointer;
type XrmOptionDescRec    is new Pointer;


XT_VERSION  : constant := 11;
XT_REVISION : constant :=  4;
Version : constant := XT_VERSION * 1000 + XT_REVISION;

Widget_Size : constant := 32;

type ModifiersList       is array ( Int range <> ) of Modifiers;

type EventMask       is new UnsignedLong;
type GCMask              is new UnsignedLong;
subtype GeometryMask     is UnsignedLong;
type InputId             is new UnsignedLong;
type ActionHookId        is new UnsignedLong;
type InputMask           is new UnsignedLong;
type IntervalId          is new UnsignedLong;
type RequestId           is new Pointer;
type ValueMask           is new UnsignedLong;
type WorkProcId          is new UnsignedLong;

type TypeConverter       is new Pointer;
TypeConverter_Size : constant := Address_Size;

type Destructor          is new Pointer;
Destructor_Size : constant := Address_Size;
```

```
  type Translations        is new Pointer;
  Translations_Size : constant := Address_Size;

  type BoundActions        is new Pointer;
  BoundActions_Size : constant := Address_Size;

  type Pixel_Ptr is access Pixel;
  type PixelList is array ( Int range <> ) of Pixel;
  type PixelList_Ptr is access PixelList;

  subtype AcceptFocusProc            is Pointer;
  subtype ActionHookProc             is Pointer;
  subtype ActionProc                 is Pointer;
  subtype AlmostProc                 is Pointer;
  subtype ArgsFunc                   is Pointer;
  subtype ArgsProc                   is Pointer;
  subtype CallbackProc               is Pointer;
  subtype CancelConvertSelectionProc is Pointer;
  subtype CaseProc                   is Pointer;
  subtype ConvertSelectionIncrProc   is Pointer;
  subtype ConvertSelectionProc       is Pointer;
  subtype Converter                  is Pointer;
  subtype ConverterProc              is Pointer;
  subtype CreatePopupChildProc       is Pointer;
  subtype ErrorHandler               is Pointer;
  subtype ErrorMsgHandler            is Pointer;
  subtype EventHandler               is Pointer;
  subtype ExposeProc                 is Pointer;
  subtype FilePredicate              is Pointer;
  subtype GeometryHandler            is Pointer;
  subtype InitProc                   is Pointer;
  subtype InputCallbackProc          is Pointer;
  subtype KeyProc                    is Pointer;
  subtype LoseSelectionIncrProc      is Pointer;
  subtype LoseSelectionProc          is Pointer;
  subtype Proc                       is Pointer;
  subtype RealizeProc                is Pointer;
  subtype SelectionCallbackProc      is Pointer;
  subtype SelectionDoneIncrProc      is Pointer;
  subtype SelectionDoneProc          is Pointer;
  subtype SetValuesFunc              is Pointer;
  subtype StringProc                is Pointer;
  subtype TimerCallbackProc          is Pointer;
  subtype WidgetClassProc            is Pointer;
  subtype WidgetProc                 is Pointer;
  subtype WorkProc                  is Pointer;

  ActionProc_Size   : constant := Address_Size;
  CallbackProc_Size : constant := Address_Size;

  InputNoneMask    : constant := 16#0000#;
  InputReadMask    : constant := 16#0001#;
  InputWriteMask   : constant := 16#0002#;
  InputExceptMask  : constant := 16#0004#;

  IMXEvent   : constant := 1;
  IMTimer    : constant := 2;
  IMAlternateInput : constant := 4;
  IMAll            : constant := 7;
```

```
function ToPointer is new Unchecked_Conversion( Widget, Pointer );

CacheNone      : constant CacheType := 16#001#;
CacheAll       : constant CacheType := 16#002#;
CacheByDisplay : constant CacheType := 16#003#;
CacheRefCount  : constant CacheType := 16#100#;

-------------------------------------------

ExposeNoCompress       : constant Enum := Enum( Xt.XtFalse );
ExposeCompressSeries   : constant Enum := Enum( Xt.XtTrue );
ExposeCompressMultiple : constant Enum := Enum( 2 );
ExposeCOmpressMaximal  : constant Enum := Enum( 3 );

-------------------------------------------

ExposeGraphicsExpose       : constant := 16#10#;
ExposeGraphicsExposeMerged : constant := 16#20#;
ExposeNoExpose             : constant := 16#40#;

-------------------------------------------

type AddressMode is new Int;

Address          : constant AddressMode := 0;
BaseOffset       : constant AddressMode := 1;
Immediate        : constant AddressMode := 2;
ResourceString   : constant AddressMode := 3;
ResourceQuark    : constant AddressMode := 4;
WidgetBaseOffset : constant AddressMode := 5;
ProcedureArg     : constant AddressMode := 6;

-------------------------------------------

type Arg is
  record
    name  : Pointer;
    value : ArgVal;
  end record;

type Arg_Ptr is access Arg;
type ArgList is array ( Int range <> ) of Arg;
type ArgList_Ptr is access ArgList;

-------------------------------------------

type Widget_Ptr is access Widget;
type WidgetList is array ( Int range <> ) of Widget;
type WidgetList_Ptr is access WidgetList;

-------------------------------------------

type GrabKind is (
  GrabNone,
  GrabNonexclusive,
  GrabExclusive );
```

```
  ------------------------------------------

  type CallbackStatus is (
    CallbackNoList,
    CallbackHasNone,
    CallbackHasSome );

  ------------------------------------------

  type ListPosition is (
    ListHead,
    ListTail );

  ------------------------------------------

  type GeometryResult is (
    GeometryYes,
    GeometryNo,
    GeometryAlmost,
    GeometryDone );

  ------------------------------------------

  type ActionRec is
    record
      str   : Pointer;
      proc  : ActionProc;
    end record;

  type ActionList is array ( Int range <> ) of ActionRec;
  type ActionRec_Ptr is access ActionRec;
  type ActionList_Ptr is access ActionList;

  ------------------------------------------

  type CallbackRec is
    record
      callback  : CallbackProc;
      closure   : Pointer;
    end record;

  type CallbackList is array ( Int range <> ) of CallbackRec;
  type CallbackRec_Ptr is access CallbackRec;
  type CallbackList_Ptr is access CallbackList;

  ------------------------------------------

  type SubstitutionRec is
    record
      match  : Char;
      subs   : Pointer;
    end record;

  type SubstitutionList is array ( Int range <> ) of SubstitutionRec;
  type SubstitutionRec_Ptr is access SubstitutionRec;
  type SubstitutionList_Ptr is access SubstitutionList;

  ------------------------------------------
```

```
type Resource is
  record
    resource_name   : Pointer;
    resource_class  : Pointer;
    resource_type   : Pointer;
    resource_size   : Cardinal;
    resource_offset : Cardinal;
    default_type    : Pointer;
    default_addr    : Pointer;
  end record;

type Resource_Ptr is access Resource;
type ResourceList is array ( Int range <> ) of Resource;
type ResourceList_Ptr is access ResourceList;

-------------------------------------------

type TMRec is
  record
    translats     : Translations;
    proc_table    : BoundActions;
    current_state : Pointer;
    lastEventTime : UnsignedLong;
  end record;

type TMRecList is array ( Int range <> ) of TMRec;
type TMRec_Ptr is access TMRec;
type TMRecList_Ptr is access TMRecList;

-------------------------------------------

type ConvertArgRec is
  record
    address_mode : AddressMode;
    address_id   : Pointer;
    size         : Cardinal;
  end record;

type ConvertArgRec_Ptr is access ConvertArgRec;
type ConvertArgList is array ( Int range <> ) of ConvertArgRec;
type ConvertArgList_Ptr is access ConvertArgList;

-------------------------------------------

type WidgetGeometry is
  record
    request_mode  : GeometryMask;
    x             : Position;
    y             : Position;
    width         : Dimension;
    height        : Dimension;
    border_width  : Dimension;
    pad1          : Dimension;
    sibling       : Widget;
    stack_mode    : Int;
  end record;

type WidgetGeometry_Ptr is access WidgetGeometry;
type WidgetGeometryList is array ( Int range <> ) of WidgetGeometry;
```

```
type WidgetGeomtryList_Ptr is access WidgetGeometryList;

CWX           : constant GeometryMask := 16#001#;
CWY           : constant GeometryMask := 16#002#;
CWWidth       : constant GeometryMask := 16#004#;
CWHeight      : constant GeometryMask := 16#008#;
CWBorderWidth : constant GeometryMask := 16#010#;
CWSibling     : constant GeometryMask := 16#020#;
CWStackMode   : constant GeometryMask := 16#040#;
CWQueryOnly   : constant GeometryMask := 16#080#;

--=========================================

procedure C_Inherit;
Inherit : constant Xt.Pointer := C_Inherit'Address;

--=========================================

function coreWidgetClass return WidgetClass;
function compositeWidgetClass return WidgetClass;
function constraintWidgetClass return WidgetClass;
function shellWidgetClass return WidgetClass;
function ObjectClass return WidgetClass;
function overrideshellWidgetClass return WidgetClass;
function rectObjClass return WidgetClass;
function transientShellWidgetClass return WidgetClass;
function toplevelShellWidgetClass return WidgetClass;
function wmShellWidgetClass return WidgetClass;
function applicationShellWidgetClass return WidgetClass;
```

Relevant portions of package Xm

```
subtype Char                  is Xt.Char;
subtype Char_Ptr              is Xt.Char_Ptr;
subtype CharList              is Xt.CharList;
subtype CharList_Ptr          is Xt.CharList_Ptr;
subtype Double        is Xt.Double;
subtype Double_Ptr        is Xt.Double_Ptr;
subtype DoubleList        is Xt.DoubleList;
subtype DoubleList_Ptr        is Xt.DoubleList_Ptr;
subtype Real                  is Xt.Real;
subtype Real_Ptr              is Xt.Real_Ptr;
subtype RealList              is Xt.RealList;
subtype RealList_Ptr          is Xt.RealList_Ptr;
subtype Int            is Xt.Int;
subtype IntList               is Xt.IntList;
subtype Int_Ptr               is Xt.Int_Ptr;
subtype IntList_Ptr           is Xt.IntList_Ptr;
subtype Long           is Xt.Long;
subtype Long_Ptr          is Xt.Long_Ptr;
subtype LongList        is Xt.LongList;
subtype LongList_Ptr        is Xt.LongList_Ptr;
subtype Pointer               is Xt.Pointer;
subtype Pointer_Ptr           is Xt.Pointer_Ptr;
subtype PointerList           is Xt.PointerList;
subtype PointerList_Ptr       is Xt.PointerList_Ptr;
subtype Short         is Xt.Short;
subtype Short_Ptr         is Xt.Short_Ptr;
```

```
subtype ShortList         is Xt.ShortList;
subtype ShortList_Ptr         is Xt.ShortList_Ptr;
subtype StringList            is Xt.StringList;
subtype StringList_Ptr        is Xt.StringList_Ptr;
subtype String_Ptr            is Xt.String_Ptr;
subtype NullString            is Xt.NullString;
subtype UnsignedChar          is Xt.UnsignedChar;
subtype UnsignedChar_Ptr      is Xt.UnsignedChar_Ptr;
subtype UnsignedCharList      is Xt.UnsignedCharList;
subtype UnsignedCharList_Ptr  is Xt.UnsignedCharList_Ptr;
subtype UnsignedInt           is Xt.UnsignedInt;
subtype UnsignedIntList       is Xt.UnsignedIntList;
subtype UnsignedLong          is Xt.UnsignedLong;
subtype UnsignedLong_Ptr      is Xt.UnsignedLong_Ptr;
subtype UnsignedLongList      is Xt.UnsignedLongList;
subtype UnsignedLongList_Ptr  is Xt.UnsignedLongList_Ptr;
subtype UnsignedShort         is Xt.UnsignedShort;
subtype UnsignedShort_Ptr     is Xt.UnsignedShort_Ptr;
subtype UnsignedShortList     is Xt.UnsignedShortList;
subtype UnsignedShortList_Ptr is Xt.UnsignedShortList_Ptr;

type ArmAndActivate    is new Pointer;
type BooleanProc       is new Pointer;
type CacheCopyProc     is new Pointer;
type CacheDeleteProc   is new Pointer;
type ColorProc         is new Pointer;
type DirProc           is new Pointer;
type ExportProc        is new Pointer;
type FileProc          is new Pointer;
type FocusMovedProc    is new Pointer;
type GeoCreateProc     is new Pointer;
type ImportOperator    is new Pointer;
type ImportProc        is new Pointer;
type InputCreateProc   is new Pointer;
type InputDisProc      is new Pointer;
type IntFunc           is new Pointer;
type MenuProc          is new Pointer;
type MenuTrav          is new Pointer;
type OutputCreateProc  is new Pointer;
type ParentProcessProc is new Pointer;
type PopFunc           is new Pointer;
type QualProc          is new Pointer;
type TraversalProc     is new Pointer;
type VisualChangeProc  is new Pointer;
type VoidProc          is new Pointer;

type Offset is new UnsignedLong;
type Offset_Ptr is access Offset;
type OffsetList is array ( Int range <> ) of Offset;

XmNULL : constant Pointer := Xt.XtNULL;

subtype XmString      is Pointer;
subtype XmStringTable is Pointer;
type XmStringList is array ( Int range <> ) of XmString;
type XmStringList_Ptr is access XmStringList;

subtype Gadget        is Xt.Widget;
```

```
  type KeySymTable is array ( Int range <> ) of Xlib.KeySym;

  subtype ButtonType    is UnsignedChar;
  type ButtonTypeTable is array ( Int range <> ) of ButtonType;

  type TextPosition is new Long;
  type TextSource   is new Pointer;

  type HighlightMode is (
    HIGHLIGHT_NORMAL,
    HIGHLIGHT_SELECTED,
    HIGHLIGHT_SECONDARY_SELECTED );

  subtype TextFormat is Xlib.Atom;

  XNULL : constant Pointer := Xt.XNULL;

  ------------

  DESTROY     : constant := 0;
  UNMAP       : constant := 1;
  DO_NOTHING  : constant := 2;

  ------------

  type MwmHints is
    record
      flags       : Long;
      functions   : Long;
      decorations : Long;
      input_mode  : Int;
    end record;

  type MwmInfo is
    record
      flags     : Long;
      wm_window : Xlib.Window;
    end record;

  MWM_HINTS_FUNCTION    : constant := 16#01#;
  MWM_HINTS_DECORATIONS : constant := 16#02#;
  MWM_HINTS_INPUT_MODE  : constant := 16#04#;

  MWM_FUNC_ALL        : constant := 16#01#;
  MWM_FUNC_RESIZE     : constant := 16#02#;
  MWM_FUNC_MOVE       : constant := 16#04#;
  MWM_FUNC_MINIMIZE   : constant := 16#08#;
  MWM_FUNC_MAXIMIZE   : constant := 16#10#;
  MWM_FUNC_CLOSE      : constant := 16#20#;

  MWM_DECOR_ALL       : constant := 16#01#;
  MWM_DECOR_BORDER    : constant := 16#02#;
  MWM_DECOR_RESIZEH   : constant := 16#04#;
  MWM_DECOR_TITLE     : constant := 16#08#;
  MWM_DECOR_MENU      : constant := 16#10#;
  MWM_DECOR_MINIMIZE  : constant := 16#20#;
  MWM_DECOR_MAXIMIZE  : constant := 16#40#;

  MWM_INPUT_MODELESS                 : constant := 0;
```

```
MWM_INPUT_PRIMARY_APPLICATION_MODAL : constant := 1;
MWM_INPUT_SYSTEM_MODAL              : constant := 2;
MWM_INPUT_FULL_APPLICATION_MODAL    : constant := 3;

MWM_INPUT_APPLICATION_MODAL    : constant := MWM_INPUT_PRIMARY_APPLICATION_MODAL;

MWM_INFO_STARTUP_STANDARD : constant := 1;
MWM_INFO_STARTUP_CUSTON   : constant := 2;

------------

LABEL_FONTLIST  : constant := 1;
BUTTON_FONTLIST : constant := 2;
TEXT_FONTLIST   : constant := 3;

------------

LOOK_AT_SCREEN        : constant := 16#001#;
LOOK_AT_CMAP          : constant := 16#002#;
LOOK_AT_BACKGROUND    : constant := 16#004#;
LOOK_AT_FOREGROUND    : constant := 16#008#;
LOOK_AT_TOP_SHADOW    : constant := 16#010#;
LOOK_AT_BOTTOM_SHADOW : constant := 16#020#;
LOOK_AT_SELECT        : constant := 16#040#;


BACKGROUND     : constant := 16#01#;
FOREGROUND     : constant := 16#02#;
TOP_SHADOW     : constant := 16#04#;
BOTTOM_SHADLOW : constant := 16#08#;
XmSELECT       : constant := 16#10#;

------------

PRIMITIVE : constant := 0;
MANAGER   : constant := 0;

GET_ACTUAL_SIZE   : constant := 1;
GET_PREFERED_SIZE : constant := 2;
GEO_PRE_SET       : constant := 3;
GEO_POST_SET      : constant := 4;

------------

GEO_EXPAND : constant := 0;
GEO_CENTER : constant := 2;
GEO_PACK   : constant := 2;

------------

GEO_PROPORTIONAL : constant := 0;
GEO_AVERAGING    : constant := 1;
GEO_WRAP         : constant := 2;

------------

GEO_ROW_MAJOR    : constant := 0;
GEO_COLUMN_MAJOR : constant := 1;
```

```
  ------------

  MENU_POPDOWN       : constant := 0;
  MENU_PROCESS_TREE  : constant := 1;
  MENU_TRAVERAL      : constant := 2;
  MENU_SHELL_POPDOWN : constant := 3;
  MENU_CALLBACK      : constant := 4;
  MENU_BUTTON        : constant := 5;
  MENU_CASCADING     : constant := 6;
  MENU_SUBMENU       : constant := 7;
  MENU_ARM           : constant := 8;
  MENU_DISARM        : constant := 9;
  MENU_BAR_CLEANUP   : constant := 10;

  ------------

  IGNORE_EVENTTYPE : constant := -1;

  ------------

  DEFAULT_INDICATOR_DIM : constant := 9;

  ------------

  NO_EVENT             : constant := 16#000#;
  ENTER_EVENT          : constant := 16#001#;
  LEAVE_EVENT          : constant := 16#002#;
  FOCUS_IN_EVENT       : constant := 16#004#;
  FOCUS_OUT_EVENT      : constant := 16#008#;
  MOTION_EVENT         : constant := 16#010#;
  ARM_EVENT            : constant := 16#020#;
  ACTIVATE_EVENT       : constant := 16#040#;
  HELP_EVENT           : constant := 16#080#;
  KEY_EVENT            : constant := 16#100#;
  MULTI_ARM_EVENT      : constant := 16#200#;
  MULTI_ACTIVATE_EVENT : constant := 16#400#;
  ALL_EVENT            : constant := 16#4ff#;

  ------------

  PUSHBUTTON        : constant := 1;
  TOGGLEBUTTON      : constant := 2;
  CHECKBUTTON       : constant := 2;
  RADIOBUTTON       : constant := 3;
  CASCADEBUTTON     : constant := 4;
  SEPARATOR         : constant := 5;
  DOUBLE_SEPARATOR  : constant := 6;
  TITLE             : constant := 7;

  -----------------------------------------

  type KeyboardData is
    record
      eventType  : UnsignedInt;
      keysym     : Xlib.KeySym;
      key        : Xlib.KeyCode;
      modifiers  : UnsignedInt;
      component  : Xt.Widget;
      needGrab   : Xt.XtBoolean;
```

```
    isMnemonic : Xt.XtBoolean;
  end record;

type KeyboardData_Ptr is access KeyboardData;
type KeyboardDataList is array ( Int range <> ) of KeyboardData;
type KeyboardDataList_Ptr is access KeyboardDataList;

-------------------------------------------

type ClipboardPendingRec is
  record
    DataId    : Int;
    PrivateId : Int;
  end record;

type ClipboardPendingRec_Ptr is access ClipboardPendingRec;
type ClipboardPendingList is array ( Int range <> ) of ClipboardPendingRec;
type ClipboardPendingList_Ptr is access ClipboardPendingList;

-------------------------------------------

type AnyCallbackStruct is
   record
     reason   : Int;
     event    : Xlib.Event_Ptr;
   end record;

type AnyCallbackStruct_Ptr is access AnyCallbackStruct;

-------------------------------------------

type ArrowButtonCallbackStruct is
   record
     reason      : Int;
     event       : Xlib.Event_Ptr;
     click_count : Int;
   end record;

type ArrowButtonCallbackStruct_Ptr is access ArrowButtonCallbackStruct;

-------------------------------------------

type CommandCallbackStruct is
   record
     reason   : Int;
     event    : Xlib.Event_Ptr;
     value    : XmString;
     length   : Int;
   end record;

type CommandCallbackStruct_Ptr is access CommandCallbackStruct;

-------------------------------------------

type DrawingAreaCallbackStruct is
   record
     reason   : Int;
     event    : Xlib.Event_Ptr;
     window   : Xlib.Window;
```

```
   end record;

type DrawingAreaCallbackStruct_Ptr is access DrawingAreaCallbackStruct;

-------------------------------------------

type DrawnButtonCallbackStruct is
   record
     reason      : Int;
     event       : Xlib.Event_Ptr;
     window      : Xlib.Window;
     click_count : Int;
   end record;

type DrawnButtonCallbackStruct_Ptr is access DrawnButtonCallbackStruct;

-------------------------------------------

type FileSelectionBoxCallbackStruct is
   record
     reason         : Int;
     event          : Xlib.Event_Ptr;
     value          : XmString;
     length         : Int;
     mask           : XmString;
     mask_length    : Int;
     dir            : XmString;
     dir_length     : Int;
     pattern        : XmString;
     pattern_length : Int;
   end record;

type FileSelectionBoxCallbackStruct_Ptr is access FileSelectionBoxCallbackStruct;

-------------------------------------------

type FocusMovedCallbackStruct is
   record
     reason         : Int;
     event          : Xlib.Event_Ptr;
     cont           : Xt.XtBoolean;
     old_widget     : Xt.Widget;
     new_widget     : Xt.Widget;
     focus_policy   : Xt.UnsignedChar;
   end record;

type FocusMovedCallbackStruct_Ptr is access FocusMovedCallbackStruct;

-------------------------------------------

type FocusData is new Pointer;  -- Can't find this structure in the .h files.

type ListCallbackStruct is
   record
     reason                 : Int;
     event                  : Xlib.Event_Ptr;
     item                   : XmString;
     item_length            : Int;
     item_position          : Int;
```

```
     selected_items          : Pointer;
     selected_item_count     : Int;
     selected_item_positions : Pointer;
     selection_type          : Int;
   end record;

type ListCallbackStruct_Ptr is access ListCallbackStruct;

-------------------------------------------

type PushButtonCallbackStruct is
   record
     reason      : Int;
     event       : Xlib.Event_Ptr;
     click_count : Int;
   end record;

type PushButtonCallbackStruct_Ptr is access PushButtonCallbackStruct;

-------------------------------------------

type RowColumnCallbackStruct is
   record
     reason         : Int;
     event          : Xlib.Event_Ptr;
     widget         : Xt.Widget;
     data           : Xt.Pointer;
     callbackstruct : Xt.Pointer;
   end record;

type RowColumnCallbackStruct_Ptr is access RowColumnCallbackStruct;

-------------------------------------------

type ScaleCallbackStruct is
   record
     reason  : Int;
     event   : Xlib.Event_Ptr;
     value   : Int;
   end record;

type ScaleCallbackStruct_Ptr is access ScaleCallbackStruct;

-------------------------------------------

type ScrollBarCallbackStruct is
   record
     reason : Int;
     event  : Xlib.Event_Ptr;
     value  : Int;
     pixel  : Int;
   end record;

type ScrollBarCallbackStruct_Ptr is access ScrollBarCallbackStruct;

-------------------------------------------

type SelectionBoxCallbackStruct is
   record
```

```
      reason   : Int;
      event    : Xlib.Event_Ptr;
      value    : XmString;
      length   : Int;
    end record;

type SelectionBoxCallbackStruct_Ptr is access SelectionBoxCallbackStruct;

-----------------------------------------

type TextBlockRec is
  record
    ptr    : Xt.Pointer;
    length : Int;
    format : TextFormat;
  end record;

type TextBlockRec_Ptr is access TextBlockRec;

subtype TextBlock is TextBlockRec_Ptr;

-----------------------------------------

type TextVerifyCallbackStruct is
   record
     reason     : Int;
     event      : Xlib.Event_Ptr;
     doit  : Xt.XtBoolean;
     currInsert : TextPosition;
     newInsert  : TextPosition;
     startPos   : TextPosition;
     endPos     : TextPosition;
     text       : TextBlock;
   end record;

type TextVerifyCallbackStruct_Ptr is access TextVerifyCallbackStruct;

-----------------------------------------

type ToggleButtonCallbackStruct is
   record
     reason : Int;
     event  : Xlib.Event_Ptr;
     set    : Int;
   end record;

type ToggleButtonCallbackStruct_Ptr is access ToggleButtonCallbackStruct;

--=========================================
--   Definitions from XmP.h
-----------------------------------------

type StringTable         is new Pointer;
subtype StringContext    is Pointer;
type StringCharSet       is new String;
type StringComponentType is new UnsignedChar;
type StringDirection     is new UnsignedChar;
subtype FontList         is Pointer;
subtype FontContext      is Pointer;
```

```
-------------------------------------------

type ColorData is
  record
    screen         : Xlib.Screen_Ptr;
    color_map      : Xlib.Colormap;
    allocated      : UnsignedChar;
    background     : Xlib.Color;
    foreground     : Xlib.Color;
    top_shadow     : Xlib.Color;
    bottom_shadow  : Xlib.Color;
    select_color   : Xlib.Color;
  end record;

type ColorData_Ptr is access ColorData;
type ColorDataList is array ( Int range <> ) of ColorData;
type ColorDataList_Ptr is access ColorDataList;

-------------------------------------------

type SyntheticResource is
  record
    resource_name   : Pointer;
    resource_size   : Xt.Cardinal;
    resource_offset : Xt.Cardinal;
    export_proc     : ExportProc;
    import_proc     : ImportProc;
  end record;

type SyntheticResource_Ptr is access SyntheticResource;
type SyntheticResourceList is array ( Int range <> ) of SyntheticResource;
type SyntheticResourceList_Ptr is access SyntheticResourceList;

-------------------------------------------

type InputActionRec is
  record
    event      : Xlib.Event_Ptr;
    action     : Int;
    params     : Pointer;
    num_params : Xt.Cardinal;
  end record;

type InputActionRec_Ptr is access InputActionRec;
type InputActionList is array ( Int range <> ) of InputActionRec;
type InputActionList_Ptr is access InputActionList;

-------------------------------------------

type ParentProcessDataRec is
  record
    process_type : Int;
    input_action : InputActionRec;
  end record;

INPUT_ACTION  : constant := 1;

XmRETURN  : constant := 1;
```

```
  XmCANCEL  : constant := 2;

  -------------------------------------------

  type SimpleMenuRec is
    record
      count            : Int;
      post_from_button : Int;
      callback         : Xt.CallbackProc;
      label_string     : StringTable;
      accelerator      : Pointer;
      accelerator_text : StringTable;
      mnemonic         : Pointer; -- KeySymTable;
      mnemonic_charset : Pointer; -- StringCharSetTable;
      button_type      : Pointer; -- ButtonTypeTable;
      button_set       : Int;
      option_label     : XmString;
      option_mnemonic  : Xlib.KeySym;
    end record;


  -------------------------------------------

  type BuildVirtualKeyStruct is
    record
      mods   : Xt.Modifiers;
      key    : Pointer;
      action : Pointer;
    end record;

  type BuildVirtualKeyStruct_Ptr is access BuildVirtualKeyStruct;
  type BuildVirtualKeyList is array ( Int range <> ) of BuildVirtualKeyStruct;
  type BuildVirtualKeyList_Ptr is access BuildVirtualKeyList;

  -------------------------------------------

  type KidGeometryRec is
    record
      kid   : Xt.Widget;
      box   : Xt.WidgetGeometry;
    end record;

  type KidGeometryRec_Ptr is access KidGeometryRec;
  subtype KidGeometry is KidGeometryRec_Ptr;
  type KidGeometryList is array ( Int range <> ) of KidGeometryRec;
  type KidGeometryList_Ptr is access KidGeometryList;

  -------------------------------------------

  type GeoRowLayoutRec is
    record
      end_rows       : Xt.XtBoolean;
      fix_up         : VoidProc;
      even_width     : Xt.Dimension;
      even_height    : Xt.Dimension;
      min_height     : Xt.Dimension;
      stretch_height : Xt.XtBoolean;
      uniform_border : Xt.XtBoolean;
      border         : Xt.Dimension;
```

```
    fill_mode      : UnsignedChar;
    fit_mode       : UnsignedChar;
    sticky_end     : Xt.XtBoolean;
    space_above    : Xt.Dimension;
    space_end      : Xt.Dimension;
    space_between  : Xt.Dimension;
    max_box_height : Xt.Dimension;
    boxes_width    : Xt.Dimension;
    fill_width     : Xt.Dimension;
    box_count      : Xt.Dimension;
  end record;

type GeoRowLayoutRec_Ptr is access GeoRowLayoutRec;
subtype GeoRowLayout is GeoRowLayoutRec_Ptr;   -- Motif defines ths as a pointer
type GeoRowLayoutList is array ( Int range <> ) of GeoRowLayoutRec;
type GeoRowLayoutList_Ptr is access GeoRowLayoutList;

-----------------------------------------

type GeoMatrixRec is
  record
    composite      : Xt.Widget;
    instigator     : Xt.Widget;
    instig_request : Xt.WidgetGeometry;
    parent_request : Xt.WidgetGeometry;
    in_layout      : Xt.WidgetGeometry_Ptr;
    boxes          : KidGeometry;
    layouts        : GeoRowLayout;
    margin_w       : Xt.Dimension;
    margin_h       : Xt.Dimension;
    stretch_boxes  : Xt.XtBoolean;
    uniform_border : Xt.XtBoolean;
    border         : Xt.Dimension;
    max_width      : Xt.Dimension;
    boxes_height   : Xt.Dimension;
    fill_height    : Xt.Dimension;
    width          : Xt.Dimension;
    height         : Xt.Dimension;
    set_except     : BooleanProc;
    almost_except  : BooleanProc;
    no_geo_request : BooleanProc;
    extension      : Xt.Pointer;
  end record;

type GeoMatrixRec_Ptr is access GeoMatrixRec;
subtype GeoMatrix is GeoMatrixRec_Ptr;
type GeoMatrixList is array ( Int range <> ) of GeoMatrixRec;
type GeoMatrixList_Ptr is access GeoMatrixList;

-----------------------------------------

type PartResource is
  record
    resource_name   : Pointer;
    resource_class  : Pointer;
    resource_type   : Pointer;
    resource_size   : Xt.Cardinal;
    resource_offset : Xt.Cardinal;
    default_type    : Pointer;
```

```
    default_addr    : Pointer;
  end record;

-------------------------------------------

InheritGeoMatrixCreate : constant Pointer := Xt.Inherit;
InhiertFocusMovedProc  : constant Pointer := Xt.Inherit;

-------------------------------------------

type ArrowPixmap is
  record
    height              : Xt.Dimension;
    width               : Xt.Dimension;
    depth               : UnsignedInt;
    top_shadow_color    : Xt.Pixel;
    bottom_shadow_color : Xt.Pixel;
    foreground_color    : Xt.Pixel;
    display             : Xlib.Display;
    pixmap              : Xlib.Pixmap;
  end record;


-------------------------------------------

ARROW_BUTTON_BIT           : constant := 32;
ARROW_BUTTON_GADGET_BIT    : constant := 33;
BULLETIN_BOARD_BIT         : constant := 34;
CASCADE_BUTTON_BIT         : constant :=  1;
CASCADE_BUTTON_GADGET_BIT  : constant :=  2;
COMMAND_BOX_BIT            : constant :=  3;
DIALOG_SHELL_BIT           : constant :=  4;
DRAWING_AREA_BIT           : constant := 35;
DRAWN_BUTTON_BIT           : constant := 14;
FILE_SELECTION_BOX_BIT     : constant := 36;
FORM_BIT                   : constant :=  6;
FRAME_BIT                  : constant := 37;
GADGET_BIT                 : constant :=  8;
LABEL_BIT                  : constant :=  9;
LABEL_GADGET_BIT           : constant := 10;
LIST_BIT                   : constant :=  5;
MAIN_WINDOW_BIT            : constant := 11;
MANAGER_BIT                : constant := 12;
MENU_SHELL_BIT             : constant := 13;
MESSAGE_BOX_BIT            : constant := 38;
PRIMITIVE_BIT              : constant := 15;
PUSH_BUTTON_BIT            : constant := 16;
PUSH_BUTTON_GADGET_BIT     : constant := 17;
ROW_COLUMN_BIT             : constant := 18;
SASH_BIT                   : constant := 39;
SCALE_BIT                  : constant := 40;
SCROLL_BAR_BIT             : constant := 19;
SCROLLED_WINDOW_BIT        : constant := 20;
SELECTION_BOX_BIT          : constant := 21;
SEPERATOR_BIT              : constant := 22;
SEPERATOR_GADGET_BIT       : constant := 23;
TEXT_BIT                   : constant := 24;
TEXT_FIELD_BIT             : constant :=  7;
TOGGLE_BUTTON_BIT          : constant := 25;
```

```
TOGGLE_BUTTON_GADGET_BIT   : constant := 26;
VPANED_BIT                 : constant := 41;
VENDOR_SHELL_BIT           : constant := 42;

-------------------------------------------

--
--  Widget class indices used with XmPartOffset and XmField Macros.
--  Because they are used with macros, these are probably not really
--  needed, but I added them just in case some one tried to use them
--

ObjectIndex              : constant := 0;
RectObjIndex             : constant := ObjectIndex + 1;
WindowObjIndex           : constant := RectObjIndex + 1;
CoreIndex                : constant := 0;
CompositeIndex           : constant := WindowObjIndex + 2;
ConstraintIndex          : constant := CompositeIndex + 1;
GadgetIndex              : constant := RectObjIndex + 1;
PrimitiveIndex           : constant := WIndowObjIndex + 2;
ManagerIndex             : constant := ConstraintIndex + 1;

ArrowBIndex              : constant := PrimitiveIndex + 1;
ArrowButtonIndex         : constant := ArrowBIndex;
LabelIndex               : constant := PrimitiveIndex + 1;
ListIndex                : constant := PrimitiveIndex + 1;
ScrollBarIndex           : constant := PrimitiveIndex + 1;
SeparatorIndex           : constant := PrimitiveIndex + 1;
TextIndex                : constant := PrimitiveIndex + 1;

CascadeBIndex            : constant := LabelIndex + 1;
CascadeButtonIndex       : constant := CascadeBIndex;
DrawnBIndex              : constant := LabelIndex + 1;
DrawnButtonIndex         : constant := DrawnBIndex;
PushBIndex               : constant := LabelIndex + 1;
PushButtonIndex          : constant := PushBIndex;
ToggleBIndex             : constant := LabelIndex + 1;
ToggleButtonIndex        : constant := ToggleBIndex;

ArrowBGIndex             : constant := GadgetIndex + 1;
ArrowButtonGadgetIndex   : constant := ArrowBGIndex;
LabelGIndex              : constant := GadgetIndex + 1;
LabelGadgetIndex         : constant := LabelGIndex;
SeparatorGIndex          : constant := GadgetIndex + 1;
SeparatorGadgetIndex     : constant := SeparatorGIndex;

CascadeBGIndex           : constant := LabelGIndex + 1;
CascadeButtonGadgetIndex : constant := CascadeBGIndex;
PushBGIndex              : constant := LabelGIndex + 1;
PushButtonGadgetIndex    : constant := PushBGIndex;
ToggleBGIndex            : constant := LabelGIndex + 1;
ToggleButtonGadgetIndex  : constant := ToggleBGIndex;

BulletinBIndex           : constant := ManagerIndex + 1;
BulletingBoardIndex      : constant := BulletinBIndex;
DrawingAIndex            : constant := ManagerIndex + 1;
DrawingAreaIndex         : constant := DrawingAIndex;
FrameIndex               : constant := ManagerIndex + 1;
PanedWIndex              : constant := ManagerIndex + 1;
```

```
  PanedWindowIndex         : constant := PanedWIndex;
  RowColumnIndex           : constant := ManagerIndex + 1;
  ScaleIndex               : constant := ManagerIndex + 1;
  ScrolledWIndex           : constant := ManagerIndex + 1;
  ScrolledWindowIndex      : constant := ScrolledWIndex;

  FormIndex                : constant := BulletinBIndex + 1;
  MessageBIndex            : constant := BulletinBIndex + 1;
  MessageBoxIndex          : constant := MessageBIndex;
  SelectionBIndex          : constant := BulletinBIndex + 1;
  SelectionBoxIndex        : constant := SelectionBIndex;

  MainWIndex               : constant := ScrolledWIndex + 1;
  MainWindowIndex          : constant := MainWIndex;

  CommandIndex             : constant := SelectionBIndex + 1;
  FileSBIndex              : constant := SelectionBIndex + 1;
  FileSelectionBoxIndex    : constant := FileSBIndex;

  ShellIndex               : constant := CompositeIndex + 1;
  OverrideShellIndex       : constant := ShellIndex + 1;
  WMShellIndex             : constant := ShellIndex + 1;
  VendorShellIndex         : constant := WMShellIndex + 1;
  TransientShellIndex      : constant := VendorShellIndex + 1;
  TopLevelShellIndex       : constant := VendorShellIndex + 1;
  ApplicationShellIndex    : constant := TopLevelShellIndex + 1;

  DialogSIndex             : constant := TransientShellIndex + 1;
  DialogShellIndex         : constant := DialogSIndex;
  MenuShellIndex           : constant := OverrideShellIndex + 1;

  LONGBITS     : constant := Long'Size;
  HALFLONGBITS : constant := LONGBITS / 2;

  -------------------------------------------

  type TextScanType is (
    SELECT_POSITION,
    SELECT_WHITESPACE,
    SELECT_WORD,
    SELECT_LINE,
    SELECT_ALL,
    SELECT_PARAGRAPH );

  -------------------------------------------
  --  Motif Widget Classes
  -------------------------------------------

  function ArrowButtonGadgetClass return Xt.WidgetClass;
  function ArrowButtonWidgetClass return Xt.WidgetClass;
  function BulletinBoardWidgetClass return Xt.WidgetClass;
  function CascadeButtonGadgetClass return Xt.WidgetClass;
  function CascadeButtonWidgetClass return Xt.WidgetClass;
  function CommandWidgetClass return Xt.WidgetClass;
  function DialogShellWidgetClass return Xt.WidgetClass;
  function DrawingAreaWidgetClass return Xt.WidgetClass;
  function DrawnButtonWidgetClass return Xt.WidgetClass;
  function FileSelectionBoxWidgetClass return Xt.WidgetClass;
  function FormWidgetClass return Xt.WidgetClass;
```

```
  function FrameWidgetClass return Xt.WidgetClass;
  function GadgetClass return Xt.WidgetClass;
  function LabelGadgetClass return Xt.WidgetClass;
  function LabelWidgetClass return Xt.WidgetClass;
  function ListWidgetClass return Xt.WidgetClass;
  function MainWindowWidgetClass return Xt.WidgetClass;
  function ManagerWidgetClass return Xt.WidgetClass;
  function MenuShellWidgetClass return Xt.WidgetClass;
  function MessageBoxWidgetClass return Xt.WidgetClass;
  function PanedWindowWidgetClass return Xt.WidgetClass;
  function PrimitiveWidgetClass return Xt.WidgetClass;
  function PushButtonGadgetClass return Xt.WidgetClass;
  function PushButtonWidgetClass return Xt.WidgetClass;
  function RowColumnWidgetClass return Xt.WidgetClass;
  function SashWidgetClass return Xt.WidgetClass;
  function ScaleWidgetClass return Xt.WidgetClass;
  function ScrollBarWidgetClass return Xt.WidgetClass;
  function ScrolledWindowWidgetClass return Xt.WidgetClass;
  function SelectionBoxWidgetClass return Xt.WidgetClass;
  function SeparatorGadgetClass return Xt.WidgetClass;
  function SeparatorWidgetClass return Xt.WidgetClass;
  function TextFieldWidgetClass return Xt.WidgetClass;
  function TextWidgetClass return Xt.WidgetClass;
  function ToggleButtonGadgetClass return Xt.WidgetClass;
  function ToggleButtonWidgetClass return Xt.WidgetClass;
  function VendorShellWidgetClass return Xt.WidgetClass;
  function compositeObjectClass return Xt.WidgetClass;
  function compositeWidgetClass return Xt.WidgetClass renames Xt.compositeWidgetClass;
  function constraintWidgetClass return Xt.WidgetClass renames Xt.constraintWidgetClass;
  function coreWidgetClass return Xt.WidgetClass renames Xt.coreWidgetClass;
  function extObjectClass return Xt.WidgetClass;
  function objectClass return Xt.WidgetClass renames Xt.objectClass;
  function rectObjClass return Xt.WidgetClass renames Xt.rectObjClass;

  ------------------------------------------
  -- W I D G E T   C A L L S

  VERSION    : constant := 1;
  REVISION   : constant := 1;
  Version1   : constant := ( VERSION * 1000 + REVISION );


-- define used to denote an unspecified piap

  UNSPECIFIED_PIXMAP: constant := 2;

--***************
--*
--* String structure defines. These must be here (at the start of the file)
--* becaused they are used later on.
--*
--***************

  STRING_DIRECTION_L_TO_R : constant StringDirection := 0;
  STRING_DIRECTION_R_TO_L : constant StringDirection := 1;

  STRING_COMPONENT_UNKNOWN: constant := 0;
  STRING_COMPONENT_CHARSET: constant := 1;
  STRING_COMPONENT_TEXT: constant := 2;
```

```
  STRING_COMPONENT_DIRECTION: constant := 3;
  STRING_COMPONENT_SEPARATOR: constant := 4;   -- 5-125 reserved

  STRING_COMPONENT_END: constant := 126; -- no more comp in string

  STRING_COMPOUND_STRING : constant := 127; -- tag for whole TCS

  STRING_COMPONENT_USER_BEGIN: constant := 128; -- 128-255 are user tags
  STRING_COMPONENT_USER_END: constant := 255;
  STRING_DEFAULT_CHARSET: constant StringCharSet( 1 .. 2 ) := ( others => ascii.nul );

  FALLBACK_CHARSET              : constant StringCharSet( 1 .. 10 ) := “ISO8859-1” &
ascii.nul;

  --
  --  size policy values
  --

  CHANGE_ALL  : constant := 0;
  CHANGE_NONE  : constant := 1;
  CHANGE_WIDTH  : constant := 2;
  CHANGE_HEIGHT : constant := 3;

  --
  --  unit type values
  --

  PIXELS     : constant := 0;
  MILLIMETERS_100TH : constant := 1;
  INCHES_100TH     : constant := 2;
  POINTS_100TH     : constant := 3;
  FONT_UNITS_100TH  : constant := 4;

  --
  --  Menu defines
  --

  NO_ORIENTATION     : constant := 0;
  VERTICAL     : constant := 1;
  HORIZONTAL         : constant := 2;

  WORK_AREA     : constant := 0;
  MENU_BAR     : constant := 1;
  MENU_PULLDOWN      : constant := 2;
  MENU_POPUP         : constant := 3;
  MENU_OPTION        : constant := 4;

  NO_PACKING         : constant := 0;
  PACK_TIGHT     : constant := 1;
  PACK_COLUMN     : constant := 2;
  PACK_NONE     : constant := 3;


--************************************************************************
-- *  Label defines
-- ************************************************************************/

  ALIGNMENT_BEGINNING: constant := 0;
  ALIGNMENT_CENTER: constant := 1;
```

```
  ALIGNMENT_END: constant := 2;


--**********************************************************************
-- *  ToggleButton  defines
-- **********************************************************************/

  N_OF_MANY  : constant := 1;
  ONE_OF_MANY  : constant := 2;


--**********************************************************************
-- *  Form resources and defines
-- **********************************************************************/

--*  Form defines

  ATTACH_NONE   : constant := 0;
  ATTACH_FORM   : constant := 1;
  ATTACH_OPPOSITE_FORM     : constant := 2;
  ATTACH_WIDGET      : constant := 3;
  ATTACH_OPPOSITE_WIDGET   : constant := 4;
  ATTACH_POSITION   : constant := 5;
  ATTACH_SELF   : constant := 6;

  RESIZE_NONE : constant := 0;
  RESIZE_GROW : constant := 1;
  RESIZE_ANY : constant := 2; --  for BulletinBoard, DrawingArea  */


  COMMAND_ABOVE_WORKSPACE : constant := 0;
  COMMAND_BELOW_WORKSPACE : constant := 1;

  --************************************************************************
  --*  Callback reasons
  --************************************************************************

  CR_NONE               : constant := 0;
  CR_HELP               : constant := 1;
  CR_VALUE_CHANGED      : constant := 2;
  CR_INCREMENT          : constant := 3;
  CR_DECREMENT          : constant := 4;
  CR_PAGE_INCREMENT     : constant := 5;
  CR_PAGE_DECREMENT     : constant := 6;
  CR_TO_TOP             : constant := 7;
  CR_TO_BOTTOM          : constant := 8;
  CR_DRAG: constant := 9;
  CR_ACTIVATE: constant := 10;
  CR_ARM : constant := 11;
  CR_DISARM: constant := 12;
  CR_MAP       : constant := 16;
  CR_UNMAP  : constant := 17;
  CR_FOCUS: constant := 18;
  CR_LOSING_FOCUS: constant := 19;
  CR_MODIFYING_TEXT_VALUE : constant := 20;
  CR_MOVING_INSERT_CURSOR : constant := 21;
  CR_EXECUTE: constant := 22;
  CR_SINGLE_SELECT: constant := 23;
  CR_MULTIPLE_SELECT: constant := 24;
```

```
  CR_EXTENDED_SELECT: constant := 25;
  CR_BROWSE_SELECT: constant := 26;
  CR_DEFAULT_ACTION: constant := 27;
  CR_CLIPBOARD_DATA_REQUEST : constant := 28;
  CR_CLIPBOARD_DATA_DELETE  : constant := 29;
  CR_CASCADING      : constant := 30;
  CR_OK               : constant := 31;
  CR_CANCEL           : constant := 32;
  CR_APPLY            : constant := 34;
  CR_NO_MATCH         : constant := 35;
  CR_COMMAND_ENTERED  : constant := 36;
  CR_COMMAND_CHANGED  : constant := 37;
  CR_EXPOSE      : constant := 38;
  CR_RESIZE      : constant := 39;
  CR_INPUT      : constant := 40;


--**********************************************************************
-- *  PushButton defines
-- **********************************************************************/

  SHADOW_IN: constant := 7;
  SHADOW_OUT: constant := 8;

--**********************************************************************
-- *  Arrow defines
-- **********************************************************************/

  ARROW_UP: constant := 0;
  ARROW_DOWN: constant := 1;
  ARROW_LEFT: constant := 2;
  ARROW_RIGHT: constant := 3;


--**********************************************************************
-- *  Separator defines
-- **********************************************************************/

  NO_LINE             : constant := 0;
  SINGLE_LINE         : constant := 1;
  DOUBLE_LINE         : constant := 2;
  SINGLE_DASHED_LINE  : constant := 3;
  DOUBLE_DASHED_LINE  : constant := 4;
  SHADOW_ETCHED_IN: constant := 5;
  SHADOW_ETCHED_OUT: constant := 6;

  XmPIXMAP: constant := 1;
  STR   : constant := 2;


--**********************************************************************
-- *  ScrollBar resource names and  s
-- **********************************************************************/

--  Resources for scrollbar

--  Defined values for scrollbar

  MAX_ON_TOP: constant := 0;
```

```
  MAX_ON_BOTTOM : constant := 1;
  MAX_ON_LEFT: constant := 0;
  MAX_ON_RIGHT  : constant := 1;


--***************
-- *
-- * Selection types
-- *
-- ***************/

  SINGLE_SELECT  : constant := 0;
  MULTIPLE_SELECT : constant := 1;
  EXTENDED_SELECT : constant := 2;
  BROWSE_SELECT  : constant := 3;

  STATIC: constant := 0;
  DYNAMIC: constant := 1;

-- ************************************************************************
-- *                                      *
-- * Scrolled Window defines.*
-- *                                      *
-- ************************************************************************/

  VARIABLE        : constant := 0;
  CONST            : constant := 1;
  RESIZE_IF_POSSIBLE: constant := 2;

  AUTOMATIC: constant := 0;
  APPLICATION_DEFINED: constant := 1;

--STATIC0     ** This is already defined by List ***/
  AS_NEEDED: constant := 1;

  SW_TOP: constant := 1;
  SW_BOTTOM: constant := 0;
  SW_LEFT: constant := 2;
  SW_RIGHT: constant := 0;

  TOP_LEFT : constant := 3;  -- (SW_TOP | XW_LEFT)
  BOTTOM_LEFT : constant := 2;  -- (SW_BOTTOM  | SW_LEFT)
  TOP_RIGHT : constant := 1;  -- (SW_TOP | SW_RIGHT)
  BOTTOM_RIGHT : constant := 0;  -- (SW_BOTTOM  | SW_RIGHT)

-- ************************************************************************
-- *                                      *
-- * MainWindow Resources                                                 *
-- *                                      *
-- ************************************************************************/


-- ************************************************************************
-- *                                      *
-- * Text Widget defines   *
-- *                                      *
-- ************************************************************************/

  MULTI_LINE_EDIT: constant := 0;
```

```
  SINGLE_LINE_EDIT: constant := 1;


-- ************************************************************************
-- *                                      *
-- *  DIALOG defines..  BulletinBoard and things common to its subclasses *
-- *          CommandBox    MessageBox    Selection    FileSelection      *
-- *                                      *
-- ************************************************************************/

--child type defines for ...GetChild() */

  DIALOG_NONE          : constant := 0; -- a valid default button type */
  DIALOG_APPLY_BUTTON  : constant := 1;
  DIALOG_CANCEL_BUTTON    : constant := 2;
  DIALOG_DEFAULT_BUTTON   : constant := 3;
  DIALOG_OK_BUTTON        : constant := 4;
  DIALOG_FILTER_LABEL     : constant := 5;
  DIALOG_FILTER_TEXT      : constant := 6;
  DIALOG_HELP_BUTTON      : constant := 7;
  DIALOG_LIST  : constant := 8;
  DIALOG_HISTORY_LIST     : constant := DIALOG_LIST;
  DIALOG_LIST_LABEL  : constant := 9;
  DIALOG_MESSAGE_LABEL    : constant := 10;
  DIALOG_SELECTION_LABEL  : constant := 11;
  DIALOG_PROMPT_LABEL     : constant := DIALOG_SELECTION_LABEL;
  DIALOG_SYMBOL_LABEL     : constant := 12;
  DIALOG_TEXT      : constant := 13;
  DIALOG_VALUE_TEXT       : constant := DIALOG_TEXT;
  DIALOG_COMMAND_TEXT     : constant := DIALOG_TEXT;
  DIALOG_SEPARATOR        : constant := 14;
  DIALOG_FILE_SELECTION   : constant := 15;

-- defines for callbacks */

--  dialog style defines  */
  DIALOG_WORK_AREA: constant := 0;
  DIALOG_MODELESS: constant := 1;
  DIALOG_APPLICATION_MODAL: constant := 2;
  DIALOG_SYSTEM_MODAL: constant := 3;

-- ************************************************************************
-- * SelectionBox, XmFileSelectionBox and XmCommand - misc. stuff       *
-- ************************************************************************/

-- defines for selection dialog type

  DIALOG_PROMPT       : constant := 7;
  DIALOG_SELECTION    : constant := 8;
  DIALOG_COMMAND      : constant := 9;

-- ************************************************************************
-- *  MessageBox           stuff not common to other dialogs            *
-- ************************************************************************/

-- defines for dialog type

  DIALOG_ERROR        : constant := 1;
  DIALOG_INFORMATION  : constant := 2;
```

```
  DIALOG_MESSAGE      : constant := 3;
  DIALOG_QUESTION     : constant := 4;
  DIALOG_WARNING      : constant := 5;
  DIALOG_WORKING      : constant := 6;

-- ************************************************************************
-- * Resource names used by Scale
-- ************************************************************************/

------------------------------------------------------------------------
-- <Xm/CutPaste.h>
--

  ClipboardFail     : constant := 0;
  ClipboardSuccess  : constant := 1;
  ClipboardTruncate : constant := 2;
  ClipboardLocked   : constant := 4;
  ClipboardBadFormat   : constant := 5;
  ClipboardNoData   : constant := 6;

------------------------------------------------------------------------

  MULTICLICK_DISCARD     : constant := 0;
  MULTICLICK_KEEP        : constant := 1;

------------------------------------------------------------------------
-- <Xm/List.h>
--
  INITIAL      : constant := 0;
  ADDITION     : constant := 1;
  MODIFICATION : constant := 2;

------------------------------------------------------------------------

  TAB_GROUP                 : constant := 1;
  STICKY_TAB_GROUP          : constant := 2;
  EXCLUSIVE_TAB_GROUP       : constant := 3;
  DYNAMIC_DEFAULT_TAB_GROUP : constant := 255;

  TRAVERSE_CURRENT          : constant := 0;
  TRAVERSE_NEXT             : constant := 1;
  TRAVERSE_PREV             : constant := 2;
  TRAVERSE_HOME             : constant := 3;
  TRAVERSE_NEXT_TAB_GROUP   : constant := 4;
  TRAVERSE_PREV_TAB_GROUP   : constant := 5;
  TRAVERSE_UP               : constant := 6;
  TRAVERSE_DOWN             : constant := 7;
  TRAVERSE_LEFT             : constant := 8;
  TRAVERSE_RIGHT            : constant := 9;
```

Relevant portions of package XlibR5

```
  subtype ICCEncodingStyle is Int;

  StringStyle : constant := 0;
  CompoundTextStyle : constant := 1;
  TextStyle : constant := 2;
  StdICCTextStyle : constant := 3;

  subtype FontSet is Pointer;
```

```
type FontSetExtents is
  record
    max_ink_extent     : Rectangle;
    max_logical_extent : Rectangle;
  end record;

for FontSetExtents use
  record
    max_ink_extent     at 0 range 0 .. Rectangle_Size - 1;
    max_logical_extent at Rectangle_Size / 8 range 0 .. Rectangle_Size - 1;
  end record;

type FontSetExtentsList is array ( Int range <> ) of FontSetExtents;
type FontSetExtents_Ptr is access FontSetExtents;
type FontSetExtentsList_Ptr is access FontSetExtentsList;

subtype IM is Pointer;
subtype IC is Pointer;

subtype IMStyle is UnsignedLong;
subtype IMCallback is Pointer;
```

Relevant portions of package XmuR5

```
type WidgetNode is
  record
    label             : Xt.Pointer;
    widget_class_ptr  : Xt.Pointer;
    superclass        : Xt.Pointer;
    children          : Xt.Pointer;
    siblings          : Xt.Pointer;
    lowered_label     : Xt.Pointer;
    lowered_classname : Xt.Pointer;
    have_resources    : Xlib.Bool;
    resources         : Xt.Pointer;
    constraints       : Xt.Pointer;
    constraintwn      : Xt.Pointer;
    nconstraints      : Xt.Cardinal;
    data              : Xt.Pointer;
  end record;
```

Relevant portions of package Xcms

```
subtype Float is Xlib.Double;
subtype ColorFormat is Xlib.UnsignedLong;
subtype CCC is Xlib.Pointer;

type BooleanList is array ( Xlib.Int range <> ) of Boolean;

UndefinedFormat : constant ColorFormat := 0;
CIEXYZFormat    : constant ColorFormat := 1;
CIEuvYFormat    : constant ColorFormat := 2;
CIExyYFormat    : constant ColorFormat := 3;
CIELabFormat    : constant ColorFormat := 4;
CIELuvFormat    : constant ColorFormat := 5;
TekHVCFormat    : constant ColorFormat := 6;
RGBFormat       : constant ColorFormat := Xlib.Int'First;
RGBiFormat      : constant ColorFormat := Xlib.Int'First + 1;
```

```
--
--   Xcms Status Return values
--

Failure : constant  := 0;
Success : constant  := 1;
SuccessWithCompress : constant  := 2;

type RGB is
  record
    red   : Xlib.UnsignedShort;
    green : Xlib.UnsignedShort;
    blue  : Xlib.UnsignedShort;
  end record;

type RGBi is
  record
    red   : Float;
    green : Float;
    blue  : Float;
  end record;

type CIEXYZ is
  record
    X  : Float;
    Y  : Float;
    Z  : Float;
  end record;

type CIEuvY is
  record
    u_prime : Float;
    v_prime : Float;
    Y       : Float;
  end record;

type CIExyY is
  record
    X  : Float;
    Y  : Float;
    YY : Float;
  end record;

type CIELab is
  record
    L_star  : Float;
    a_star  : Float;
    b_star  : Float;
  end record;

type CIELuv is
  record
    L_star  : Float;
    u_star  : Float;
    v_star  : Float;
  end record;

type TekHVC is
```

```
    record
      H  : Float;
      V  : Float;
      C  : Float;
    end record;

  type Pad is
    record
      pad0  : Float;
      pad1  : Float;
      pad2  : Float;
      pad3  : Float;
    end record;

  Pad_Size : constant := 4 * Float'Size;

  type Color( format : ColorFormat := RGBFormat ) is
    record
      pixel : Xlib.UnsignedLong;
      case format is
        when UndefinedFormat =>
       pad_color    : Pad;
        when CIEXYZFormat =>
       ciexyz_color : CIEXYZ;
        when CIEuvYFormat =>
       cieuvy_color : CIEuvY;
        when CIExyYFormat =>
       ciexyy_color : CIExyY;
        when CIELabFormat =>
       cielab_color : CIELab;
        when CIELuvFormat =>
       cieluv_color : CIELuv;
        when TekHVCFormat =>
       tekhvc_color : TekHVC;
        when RGBFormat =>
       rgb_color    : RGB;
        when RGBiFormat =>
       rgbi_color   : RGBi;
     when others =>
       pad_clr      : Pad;
      end case;
    end record;

  type Color_Ptr is access Color;
  type ColorList is array ( Xlib.Int range <> ) of Color;
  type ColorList_Ptr is access ColorList;
```

Relevant portions of package Xwc

```
  subtype FontSet is XlibR5.FontSet;
  subtype IC      is XlibR5.IC;
  subtype ICCEncodingStyle is XlibR5.ICCEncodingStyle;

  subtype wchar_t is Xlib.UnsignedShort;

  type wchar_t_String is array ( Xlib.Int range <> ) of wchar_t;
  type wchar_t_String_Ptr is access wchar_t_String;
  type wchar_t_StringList is array ( Xlib.Int range <> ) of wchar_t_String_Ptr;
```

```
type wchar_t_StringList_Ptr is access wchar_t_StringList;

type TextItem is
  record
    chars    : Xlib.Pointer;
    nchars   : Xlib.Int;
    dlta     : Xlib.Int;
    font_set : FontSet;
  end record;

type TextItem_Ptr is access TextItem;
type TextItemList is array ( Xlib.Int range <> ) of TextItem;
type TextItemList_Ptr is access TextItemList;
```

B

Concurrent-Specific Information

Using the AXI Bindings

Once installed, the AXI bindings may be used in Ada applications. This section describes the steps required in order to compile, link, execute, and debug programs that use the AXI bindings.

Compiling

In order to compile applications that reference any part of the AXI bindings (e.g., Xlib, Xt, Motif), a user must first include the appropriate environment(s) on the Environment Search Path for the MAXAda environment where the application is to be compiled.

It is necessary to include either one or both of the following MAXAda environments on the Environment Search Path:

```
xlibxt
motif
```

The first is necessary for all programs that use the Xlib or Xt interfaces. If these are used exclusively, then this is the only additional environment required on a MAXAda environment's Environment Search Path. It can be added using the MAXAda utility **a.path**.

The MAXAda utility **a.path** recognizes the AXI environment names as keywords. Therefore, if only the Xlib and Xt interface will be used, then the following command should be issued:

```
a.path -A xlibxt
```

If Motif is also to be used, then both libraries must be included on the Environment Search Path. However, the **motif** environment includes the **xlibxt** environment on its search path, so adding **motif** to your path will transitively add **xlibxt**. To add both environments to your Environment Search Path, issue the following command:

```
a.path -A motif
```

Aside from including the appropriate environments on the Environment Search Path, nothing special needs to be done in order to compile applications that use the AXI interface. Compile source code that contains AXI calls in the same manner as you would any other Ada source file by invoking the MAXAda build utility, **a.build**. For example:

```
a.build helloworld
```

compiles the MAXAda partition **helloworld** which contains a simple Motif "helloworld" program that utilizes the AXI bindings. The source for this program is provided in this Appendix.

Linking

Linking AXI programs does not require any special steps other than the presence of the appropriate AXI environments on the Environment Search Path of the target MAXAda environment. The AXI interface internally links in the C libraries required (e.g., **`libX11.a`**, **`libXt.a`**, **`libXm.a`**) when applications use any AXI- specific features that would require these C libraries to be linked.

Note that the AXI interface only supports X11R6 versions of the C libraries and that AXI programs will not link unless the R6 versions are installed on your system.

AXI contains Ada shared objects for each supplied AXI environment. Therefore, Ada programs that utilize any AXI environment may choose to link AXI environments either statically or dynamically.

The AXI environments require routines from two PowerMAX OS 4.1 network libraries which are available only as shared objects. For this reason, **`a.build`** automatically links the required shared object (**`libnsl.so`**, **`libsocket.so`**, **`libresolv.so`** and **`libc.so`**) into the AXI program, whether the rest is static or dynamic.

For more information regarding Ada shared objects and shared libraries, refer to the *MAXAda Reference Manual*.

Executing

Executing a program that utilizes the AXI interface requires only that you run the program on an X terminal with an appropriate `DISPLAY` variable setting.

Debugging

Ada programs that use the AXI interface may be debugged using the NightView debugger just like any other Ada program. However, since no source is supplied for the bodies of the packages that make up the AXI interface, debugging through any AXI functions and procedures may be done only at the assembly level. Because most AXI calls access C libraries, these would have to be debugged at the assembly level anyway.

The NightView debugger traps most UNIX signals by default, stopping the program being debugged in the process. Because X programs rely heavily on the use of SIGIO, NightView will not stop (by default) when the application being debugged encounters SIGIO. This default behavior may be modified using the **`handle`** command. All other UNIX signals are trapped and handled by the debugger as documented in the *NightView User's Guide*.

Programming Hints

Setting and Getting Widget Resources

The X Tool kit and Motif widget-creation convenience functions expect you to identify widget resources in a static array. This array is passed as an argument to the convenience function along with an argument specifying the number of resources in the list.

For example, the often-used Xt function `CreateManagedWidget()` accepts as its arguments: the name of the widget to create; the class of the widget; the widget's parent; and, as its last two arguments, an array of resource pairs and an integer specifying the number of resource pairs in the array.

When using the AXI bindings, Motif and Xt resources may easily be constructed by creating argument lists using the `SetArg` procedure that is defined in the `Xt` package. An argument list that is created using this technique may then be passed to the Ada version of the convenience function along with the number of resources defined in the list.

The following Ada code fragments demonstrate how you may specify a horizontal orientation when creating a simple Motif `RowColumn` widget. You first create an argument list of widget resources. Then

you pass this list as an argument to the `CreateManagedWidget` function found in the `Xt` package. In this example, only one resource pair is specified in the list (to specify the orientation of the widget):

```
with Xt;
with Xm;
with Xmdef;

        ...

        arglist      : Xt.arglist(1..1);
        rowcol_w, toplevel   : Xt.Widget;

        ...

        Xt.SetArg (arglist(1), Xmdef.Norientation, Xt.Argval(Xm.HORIZONTAL));
        rowcol_w := Xt.CreateManagedWidget("Row", Xm.RowColumnWidgetClass,
        toplevel, arglist, 1);
        ...
```

A few things are important to note regarding resource lists:

- The resource list is of the AXI-defined type `Xt.arglist` and is an array type. The `SetArg` procedure may be called repetitively to set as many resource pairs in the argument list array as necessary.
- The Motif resource names are defined in the `Xmdef` package and parallel the names of the resources as they appear in the C language header files with a similar notation. (In this case `Norientation` appears as it would in C without the `Xt` or `Xm` prefix that the C language uses. For Ada, these resource names MUST BE prefixed by `Xmdef.`, UNLESS a `use` clause is given for this package, as all resource names are defined in the `Xmdef` package).
- Constants for resource values are defined within either the `Xt` or `Xm` packages, depending upon whether the value is an X Tool kit or Motif resource value. The names for these constants also mimic the C language definitions in the C header files (`Xm.HORIZONTAL` in this case parallels the C value `XmHORIZONTAL`).
- Resource values may either be simple integer values or must be converted to the AXI-defined type `Xt.Argval` as shown in the preceding example. The `SetArg` procedure is overloaded to accept either type as a resource value.

Callbacks

Callback routines are an integral part of X Toolkit programming. To register a callback routine in an Ada program using Xt or Motif is quite similar to the C language; however, certain programming techniques should be used when programming Ada callbacks in order to minimize errors.

A callback may be registered in Ada by using the `AddCallback` procedure found in the `Xt` package. Its arguments are similar to the C function `XtAddCallback()`. Be aware when programming Ada callbacks that all callback routines must be defined in library-level packages. Callback routines may not be defined in shared instantiations of generic packages. Because the Concurrent C compiler and the MAXAda compiler use similar calling conventions, nothing further needs to be done in order to register an Ada callback routine with the Xt Tool kit. The requirement that Ada callbacks be defined at the library level eliminates confusion introduced by uplevel references.

An example of an Ada callback routine and a code fragment demonstrating how to register the callback

follow:

```
--
-- A simple Ada callback routine in a library-level package body.
--

with text_io;
package body callback is
--
        procedure call_me (w   : in Xt.Widget;
            client_data : in Xt.Pointer;
            call_data   : in Xt.Pointer) is
        begin
            text_io.put_line (“Callback was called”);
        end call_me;
--
end callback;

--
-- This callback can then be registered with the X Toolkit
-- for an AXI application where “some_widget” is defined.
--

with callback;
procedure test is
--
        ...

        Xt.AddCallback(some_widget, Xtdef.NactivateCallback,
            callback.call_me’address, Xt.XtNULL);
        ...

--
end test;
```

Full debugging of Ada callbacks is supported when using the NightView symbolic debugger; however, traversing the call stack backwards from a callback routine into C code found within the Xt library can be debugged only at the assembly level from within the debugger.

Ada Tasking in AXI Applications

The interplay between tasks and AXI depends on the tasking configuration of each application. To understand the configuration choices, a basic understanding of the Ada tasking implementation is required.

Overview of Tasking Implementation

This section provides a brief review of the MAXAda tasking implementation. Consult the *MAXAda Reference Manual* for a more complete explanation of Ada tasking and run-time configuration.

Ada tasks are implemented as states of execution that are dedicated to execute the Ada source instructions associated with an Ada “task” construct, as defined by the Ada 95 Reference Manual (ANSI/ISO/IEC-8652:1995). Tasks execute on a CPU when an operating system lighweight process (LWP) is assigned to serve them. The relationship between one or more LWPs and a task is known as the task’s *weight*. Tasks can be either one of two weights:

- Bound

- Multiplexed

A *bound task* is a task that has a distinct LWP permanently assigned to its execution. That LWP is always available to execute the task whenever the task is in an executable state. That LWP is never used for any other purpose while the task exists.

A *multiplexed task* is a member of a group of tasks that have one or more LWPs assigned for their execution. At any given time, an LWP may or may not be available to execute a single task within its group.

For example, a multiplexed task that executes a system call which requires it to block may effectively block all other multiplexed tasks within its group (if there is only a single LWP assigned to serve a group; this is the default configuration.)

Programmers may set the weight of specific tasks or all tasks via the implementation-defined pragma `TASK_WEIGHT`. Programmers may set the default weight for all tasks via the link-time options **`-bound`** or **`-multiplexed`** as specified to the **`a.partition`** MAXAda utility.

By default, all tasks are multiplexed with a single LWP to serve them.

Multithreaded AXI Access

The X Window Interface libraries are not protected from concurrent access by tasks (or threads or LWPs) within a single application. Therefore, it is best to isolate all AXI activity to a single task within an application. Alternative methods are available, but they require that the programmer accurately predict the actual scheduling of all tasks within the application so as to prevent concurrent access to AXI subprograms.

AXI Blocking Behavior

When a task makes an AXI call that is potentially blocking, the LWP that executes the call may block. This prevents any other task that must be served by that LWP from executing until the call is completed or interrupted.

If a bound task makes such a call, it will not affect any other tasks in the application, since a bound task has a single LWP dedicated for its use. That LWP would not be used for any other task in the application.

Alternatively, if a multiplexed task makes such a call, it is likely that no other *multiplexed* task will execute until the call is completed or interrupted.

The time-slicing nature of multiplexed tasks eventually causes the AXI call to be interrupted, thus allowing other multiplexed tasks to execute. (The call will be continued some time in the future.) However, the length of the time slice may complicate predictability and may adversely affect application performance. (E.g., no application activity occurs for multiplexed tasks until the expiration of the time slice for the task making the AXI call.)

If blocking issues are of concern to the programmer, we recommend using bound tasks when interfacing to AXI subprograms. This prevents potentially blocking AXI calls from affecting the scheduling of other tasks in the application.

Optimizations

The MAXAda compiler supports three levels of optimization. The details of these levels of optimization are outlined in the *MAXAda Reference Manual*. Ada programs that use the AXI interface may be fully optimized.

Other link-time optimizations are available and may be performed to improve the execution speed of a program. However, selective linking (**`a.partition -c`** option) will be disabled due to the use of shared libraries.

AXI imposes no restrictions on using any of the optimization techniques available while compiling and

linking AXI applications using the MAXAda environment.

An Example Motif Program

The following is the source code for a simple Motif program that uses the AXI bindings. It may be compiled and linked using the MAXAda compiler as shown earlier in this Appendix.

```
--
-- Make all appropriate AXI packages visible
--
with Xm;
with Xrm;
with Xt;
with Xlib;


procedure helloworld is
--
        app_context     : Xt.AppContext;
        toplevel, label : Xt.Widget;
        argv    : Xt.stringlist_ptr;
        options : Xrm.optiondesclist (1 .. 0);
        fallback: string (1 .. 0) := “”;
        args    : Xt.arglist (0 .. 1);
--
begin
--
        argv := Xlib.getarguments;

        -- Create and initialize the application context

        app_context := Xt.createapplicationcontext;
        Xt.AppInitialize (app_context, “Hello”,  options, 0, argv,
            fallback, args, 0, toplevel);

        -- Create a simple label widget

        label := Xm.CreateLabel (toplevel, “Hello_World”, args, 0);
        Xt.ManageChild (label);

        -- Enter the event loop

        Xt.RealizeWidget (toplevel);
        Xt.AppMainLoop (app_context);
--
end helloworld;
```

Other sample programs that demonstrate the usage of the AXI bindings are available from Concurrent. Contact your Concurrent analyst to obtain a copy of the sample programs. The sample programs that are available use the AXI bindings to exercise features in Xlib, Xt and Motif.

Programming Pitfalls

Many Xt and Motif references discourage the use of system calls such as `fork()`, `exec()`, and `system()` because error conditions that may arise (when using these system calls) are virtually unrecoverable from an

X application's standpoint. If error recovery is not a critical matter from the X application's view, then the use of such system calls may be appropriate, and often useful.

AXI calls were designed to map as closely as possible to the C language library functions to which they *bind*. Ada typing restrictions sometimes make this task more difficult, and several AXI-defined Ada types have been defined in order to make this task a bit easier. Users should always consult the AXI documentation or refer to the package specifications in order to choose the most appropriate form of any Xlib, Xt, or Motif function that they wish to employ. Often several different Ada interfaces are available (as overloaded Ada functions and procedures) for a single C function. If chosen properly, much work can be saved by avoiding Ada type conversions when programming with the AXI bindings.

Spine for 1.0” Binder

Product Name: 0.5" from top of spine, Helvetica, 36 pt, Bold

Volume Number (if any): Helvetica, 24 pt, Bold

Volume Name (if any): Helvetica, 18 pt, Bold

Manual Title(s): Helvetica, 10 pt, Bold, centered vertically within space above bar, double space between each title

Bar: 1" x 1/8" beginning 1/4" in from either side

Part Number: Helvetica, 6 pt, centered, 1/8" up