ULTRIX Worksystem Software

Release Notes

Order Number: AA-HF07E-TN

ULTRIX Worksystem Software

Release Notes

Order Number: AA-HF07E-TN
June 1990

Product Version:	UWS Version 4.0
Operating System and Version:	ULTRIX Version 4.0

**digital equipment corporation
maynard, massachusetts**

Contents

4 DECwindows Toolkit Programming

6 UWS Documentation

About This Manual

Audience

The *ULTRIX Worksystem Software (UWS), Version 4.0 Release Notes* is for the following audiences:

- New users
- Users migrating from UWS V2.0 to UWS V4.0
- Users migrating from UWS V2.1 to UWS V4.0
- Users migrating from UWS V2.2 to UWS V4.0.

Please note that all UWS V4.0 software has been compiled under ULTIRX V4.0 except the following:

- `/usr/bin/X11/xterm`
- `/usr/bin/X11/x10tox11`
- `/usr/bin/Xgb`
- `/usr/bin/dxfc3d`
- `/usr/lib/rgb`
- `/usr/bin/dxpsview3d`
- Some PostScript-related object files (.o files) and libraries of object files (.a files) from Adobe Systems Inc.

Organization

UWS Version 4.0 Release Notes include the following:

Chapter 1	UWS X Servers Describes the changes since the UWS Version 2.1 release that have been made to the servers including fixed problems and the addition of Display PostScript extensions.

Chapter 2	Display PostScript Provides information on the Display PostScript (DPS) software implementation of the PostScript language via an extension to the DECwindows server.
Chapter 3	Fonts Provides notes on screen font functionality.
Chapter 4	DECwindows Toolkit Programming Describes those areas that apply to DECwindows Toolkit Programming.
Chapter 5	User Environment Describes those applications that apply to the user environment.
Chapter 6	Documentation Provides notes that pertain to the ULTRIX Worksystem Software documentation that is shipped with your ULTRIX Worksystem Software, Version 4.0.

Appendices

Appendix A	Addressed Problems Describes Software Performance Reports (SPRs) that have been addressed for this release of ULTRIX Worksystem Software.

Conventions

The following conventions are used in this manual:

%	The default user prompt is your system name followed by a right angle bracket. In this manual, a percent sign (%) is used to represent this prompt.
#	A number sign is the default superuser prompt.

`>>>`
`CPU`*nn*`>>`
	The console subsystem prompt is two right angle brackets on RISC systems, or three right angle brackets on VAX systems. On a system with more than one central processing unit (CPU), the prompt displays two numbers: the number of the CPU, and the number of the processor

	slot containing the board for that CPU.
`user input`	This bold typeface is used in interactive examples to indicate typed user input.
`system output`	This typeface is used in interactive examples to indicate system output and also in code examples and other screen displays. In text, this typeface is used to indicate the exact name of a command, option, partition, pathname, directory, or file.
UPPERCASE lowercase	The ULTRIX system differentiates between lowercase and uppercase characters. Literal strings that appear in text, examples, syntax descriptions, and function definitions must be typed exactly as shown.
macro	In text, bold type is used to introduce new terms.
filename	In examples, syntax descriptions, and function definitions, italics are used to indicate variable values; and in text, to give references to other documents.
[ ]	In syntax descriptions and function definitions, brackets indicate items that are optional.
{ | }	In syntax descriptions and function definitions, braces enclose lists from which one item must be chosen. Vertical bars are used to separate items.
. . .	In syntax descriptions and function definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
. . .	A vertical ellipsis indicates that a portion of an example that would normally be present is not shown.
`cat`(1)	Cross-references to the *ULTRIX and UWS Reference Pages* include the appropriate section number in parentheses. For example, a reference to `cat`(1) indicates that you can find the material on the `cat` command in Section 1 of the reference pages.
$\boxed{\text{RETURN}}$	This symbol is used in examples to indicate that you must press the named key on the keyboard.
$\boxed{\text{CTRL}/x}$	This symbol is used in examples to indicate that you must hold down the CTRL key while pressing the key *x*

	that follows the slash. When you use this key combination, the system sometimes echoes the resulting character, using a circumflex (^) to represent the CTRL key (for example, ^C for CTRL/C). Sometimes the sequence is not echoed.
ESC X	This symbol is used in examples to indicate that you must press the first named key and then press the second named key. In text, this combination is indicated as ESC-X.
MB1,MB2,MB3	Unless the mouse buttons have been redefined, MB1 indicates the left mouse button, MB2 indicates the middle mouse button, and MB3 indicates the right mouse button.

UWS X Servers 1

1.1 Incorrect Display of the Scanline

The ULTRIX QVSS driver returns a data structure that contains the displayable portion of the scanline. For VR260 monitors this is 1024 bits. For the VR100 monitor this is 960 bits. The bit manipulation routines incorrectly assume that this number is the length of the scanline. This causes the display to be scrambled on a VR100. The scanline on the QVSS is always 1024 although some monitors may not be able to display the full line.

1.2 UWS X Servers

Changes to the servers include bug fixes. All the servers, except the `Xgb` server, have the Display PostScript System extension.

Listed below are the UWS X servers contained in the UWS V4.0 release.

- `Xqdsg`: for 8- and 4-plane VAX color workstations
- `Xqvsm`: for monochrome VAX workstations
- `Xcfb`: for color DECstation 2100s, 3100s, and the DECstation 5000 Model 200
- `Xmfb`: for monochrome DECstation 2100s and 3100s
- `Xgb`: for VAXstation 3520/3540

1.2.1 Default Keyboard Keymap

The default keyboard keymap is US LK201-LA. To load a different keyboard keymap, you must first log in as superuser and then create a symbolic link `keymap_default` in the directory `/usr/lib/X11`, which points to the keyboard keymap you want to load. The following example shows you how to set the default keyboard keymap to the Swedish LK201:

```
# cd /usr/lib/X11
# ln -s keymaps/swedish_lk201lm.decw_keymap keymap_default
```

Note that you must restart the X server after changing the default keyboard keymap. To restart the X server, type the following at the superuser prompt:

```
# /etc/shutdown now
# ^D
```

To set the default keyboard keymap to US LK201-LA, the file `/usr/lib/X11/keymap_default` must be removed.

1.2.1.1 Keymap_default File for Multiple Diskless Clients – If you have multiple diskless clients (for example, three VS2000s) with different keyboards (German, French, English, Spanish) you cannot map the `keymap_default` entry of `/usr/lib/X11` to be private for every client.

1.2.2 Console Messages on Color VAX Displays

When the X server is running and a console window is not provided, system messages that are sent to the console on 8-plane systems are displayed as blank black lines beginning at the left edge of the screen. In addition, the `XPrompter` dialog box or the other portions of the display might be corrupted. Press the `Clear` button in the `Xprompter` dialog box to remove the corruption.

1.2.3 Invalid Font Path

If an invalid font path is specified on the ULTRIX server startup command line, the server will crash when it connects with the first client.

1.2.4 Hostnames in X Server Access Control List

When the X server starts up, it places localhost and the name of the host where the X server is running into the server's access control list. In addition, if there is a `/etc/X0.hosts` file or `/etc/X1.hosts` file, its contents are added to the access control list.

If you have `sm.host_list` resource in your `.Xdefaults` file, the access control list replaces the server's list when `dxsession` starts up. If no such resource exists, `dxsession` will not change the server's list.

When you display the Session Manager Customize Security dialog box, the server's current access control list is listed. If this list is changed and the current settings are saved, a `sm.host_list` resource is placed into the `.Xdefaults` file. The next time you start `dxsession`, the server's list will be replaced by this resource.

1.2.5 X Server Messages File

X server messages are logged in the `/usr/adm/X#msgs` file. The file contains the date and time the X server restarts, X error messages, and miscellaneous information about X server crashes at server restart. The number sign (#) indicates the number of the display where messages are being logged. For workstations with single displays, the messages file name is `X0msgs`. For workstations with two displays (VAXstation II/GPX only) there are two messages files, `X0msgs` and `X1msgs`, the first and second display respectively.

1.2.5.1 X Messages – Some of the X messages are fatal errors.

Messages reported to the `X#msgs` file include:

- "Use" messages, which indicate that the line in the `/etc/ttys` file for starting the server has a typographical error. Check the `/etc/ttys` file for spelling or syntax errors.
- "Resource missing" or "Resource installed improperly" messages, which indicate that fonts are not installed properly, or that the `-fp` switch in the `/etc/ttys` line is not correct. Check to see that the fonts you are using are installed properly. Check to see that the `-fc` option for setting the default cursor font is correct. Check to see that the `-fn` option for setting the default text font is correct.

For example:

```
main: Could not open default font 'XXX'

main: Could not open default cursor font 'XXX'
```

- "Could not open RGB_DB 'XXX'" messages, which indicate that the rgb data base is not installed correctly. These messages refer to files that apply to color workstations only. For VAXstation 3520/3540s, the data base files are in `/usr/lib/rgb.*`. For all other color workstations, the data base files are in `/usr/lib/X11/rgb.*`

```
Could not open RGB_DB 'XXX'
```

- "Screen failed initialization" messages, which usually indicate that the file `/dev/mouse` is not protected correctly or there is already a server running.

```
'screen failed initialization'
```

- Other error messages indicate that the X server is out of memory.

1.2.6 How To Restart The X Server

The following procedure briefly describes a way to restart the X server. This procedure should be followed if your X server should hang:

1. Log in remotely to your workstation.
2. Change your user ID to root using the `su(1)` command.
3. Enter the following command:

   ```
   # ps -ax
   ```

4. Locate the X server process. The output of the `ps(1)` command should display an X server process similar to the following:

   ```
        PID TT STAT  TIME COMMAND
         95 ?  S   117:09 - :0 (Xqdsg)
   ```

5. Kill the X server PID, by entering the following command:

   ```
   # kill -9 95
   ```

   By killing the X server process, the server restarts assuming that the server startup line is the file `/etc/ttys`.

1.2.7 VAX Color Workstations

The `Xqdsg` server:

- Allows you to specify a plane mask in the gc.
- Supports five types of visuals for the root window. The default can be changed by using a command line option in `/etc/ttys`:

  ```
        -class <classname>   type of Visual for root window,
          one of StaticGray, StaticColor, PseudoColor,
          GrayScale, or even TrueColor
  ```

- Provides performance improvements for all filled rectangles (stippled, tiled and solid fill) and dashed lines.

1.2.8 VAXstation 3520/3540

The `Xgb` server uses a different font set, font compiler, color database and PostScript Previewer than the other servers in this release. To avoid confusion, these components have been renamed for use with the `Xgb` server. These names are all resolved at installation time, but when running the font compiler or the Previewer, you need to know the new names, which are listed in the following table:

1.2.9 Component Names for the Xgb Server

Component	UWS V2.0/UWS V2.1	UWS V2.2/UWS V4.0
Fonts	`/usr/lib/dwf`	`/usr/lib/dwf`
Font compiler	`/usr/bin/dxfc`	`/usr/bin/dxfc3d`
Color database	`/usr/lib/X11/rgb.*`	`/usr/lib/rgb.*`
PostScript previewer	`/usr/bin/dxpsview`	`/usr/bin/dxpsview3d`

1.3 X Clients Built Under UWS V2.0 and V2.1

For compatibility, set the `DISPLAY` variable to `unix:0`. Unlike UWS V2.0 and UWS V2.1, there is no shared memory transport.

1.4 MIT Clients

The MIT clients, in `/usr/bin/X11`, are from the MIT X11R3 distribution. Any source code from MIT will link with the installed Xlib with patches through May, 1989. These clients are in the ULXUNMIT400 subset (VAX) or UDXUNMIT400 subset (RISC), which is included in the unsupported software.

1.4.1 Source Code Changes For POSIX Compliance

In order to get MIT X11R3 `/usr/bin/X11/x10tox11` to compile in POSIX compliant operating system ULTRIX 4.0, it was necessary to make the following changes as represented by diffs (R3 = < mods = >):

The need to make these changes resulted from changes in the type declarations of the string functions. These are declared in `/usr/include/strings.h` or `/usr/include/string.h` and should not be declared in a local file.

```
diff resource.h resource.h.mod
368,374d367
< /*
<  * Libc functions.
<  */
< char *sprintf();
< char *strncpy();
< char *strcpy();
< int strlen();
```

In order to get MIT X11R3 xterm to compile in a POSIX compliant operating system, ULTRIX 4.0, it was found necessary to make the following changes as represented by diffs (R3 = < mods = >):

The need to make these changes resulted from changes in the definition of the fileno macro. As a result, the R3 code in these sections encountered fatal compiler errors of "illegal lhs assignment".

```
# diff main.c main.c.mod
848c848
<         fileno(stderr) = i;
- ---
>         stderr -> _file = i;
851c851
<         fileno(stderr) = i;
- ---
>         stderr -> _file = i;

# diff misc.c misc.c.mod
593c593
<                 fileno(stderr) = 2;
- ---
>                 stderr -> _file = 2;
```

1.5 Xlib Library

Changes to the Xlib data structures and programming interfaces to support the Inter-Client Communication Conventions include:

- Removal of the `PAllWMSizeHints` macro.
- Removal of the `XGetWMCommand` and `XSetWMCommand` routines.
- Correction of the order of screen and window arguments in `XIconifyWindow`, `XWithdrawWindow`, and `XReconfigureWMWindow`.
- The `killid` field in the definition of `XStandardColormap` has not been voted on by the wmtalk (window manager talk) list but is viewed as a necessary change by the Director of the X Consortium. A change in this part of the interface is possible if the wmtalk group comes up with an alternative proposal.

1.5.1 LockDisplay and UnLockDisplay Macros

The X Window System programming manuals mention use of `LockDisplay` and `UnlockDisplay` macros for writing multi-threaded X clients that access buffers of the Display structure. You often use these macros when writing Xlib side extensions to the core X11 protocol.

However, UWS Version 2.2 and UWS Version 4.0 do not support these side extensions. Hence, the `LockDisplay` and `UnlockDisplay` macros are not included. If you require these macros, order the UWS Version 4.0 source kit. Note that you may be able to accomplish display structure locking using the public domain macros provided by MIT's Xlib.

Note also that supported core Xlib calls do proper structure locking and are safe for multi-threaded clients to use.

1.5.2 Memory Allocation Routines

`Xlib` uses its own internal versions of `malloc()`, `calloc()`, `free()` and `realloc()`, which are also used by the XUI Toolkit. All clients that link with `Xlib` or the XUI Toolkit should use these versions. The `malloc()`, `free()`, and `realloc()` routines are defined in `XvmsAlloc.o` in `libX11.a`; the `calloc()` routine is defined in `XvmsCalloc.o` in `libX11.a`. If you prefer to use your own internal version of one or more of these routines, then you must take action to avoid multiple declaration errors when linking. You can do this by declaring entry points in your code for all four of these routines and linking normally with:

```
cc prog1.c -lX11
```

Alternatively, note that in `/usr/lib/libc.a`, `calloc()` is defined in `calloc.o`, but `malloc()`,`realloc()`, and `free()` are defined in `malloc.o`. Therefore, if you declare your own private routines for `malloc()`, `realloc()`, and `free()`, but want to pull in `calloc()` from `/usr/lib/libc.a`, you can do so with:

```
ar x /usr/lib/libc.a calloc.o
cc prog2.c calloc.o -lX11
```

1.5.3 ULTRIX System V Emulation Library

To use the ULTRIX System V emulation library you should link `/usr/lib/XlibIntV.o` before `/usr/lib/libX11.a`. For example, you would enter:

```
    cc xsample.c /usr/lib/XlibIntV.o -lX11 -lcV
```

1.5.4 XCopyArea Function

An application may hang and freeze up the system with a `XCopyArea` function call followed by a `XIfEvent` call. To avoid the possibility of deadlock, use a `XSync` function call before issuing the `XIfEvent` function call. `XSync` will flush the output buffer and wait for all requests to be received and processed by the server. In general, if `XIfEvent` is waiting for an expected event generated by a `X` function call, `XSync` should first be used to guarantee that all the requests to the server have been sent out. The application must guarantee that all events are truly written out before the wait is begun.

1.5.5 XDrawArc(s) Function

Double dash mode does not work with the `XDrawArc`(s) call, which includes circles and ellipses. This is found in some MIT R3 applications.

1.5.6 Data Structures and Constants

In UWS Version 2.2, additions were made to the `XSizeHints` and `XStandardColormap` data structures and a new structure was added for manipulating properties containing text.

1.5.6.1 X Size Hints – The following routines are used for getting and setting size hints:

`XGetWMSizeHints` returns the size hints stored in the indicated property on the specified window. If the property is of type WM_SIZE_HINTS and format 32 and is long enough to contain a size hints structure, the various fields of the `hints_return` structure are set and a non-zero status is returned. Otherwise, a status of 0 is returned. To get a window's normal size hints, the routine `XGetWMNormalHints` may be used instead.

```
Status XGetWMSizeHints (dpy, w, hints_return, property)
    Display *dpy;
    Window w;
    XSizeHints *hints_return;
    Atom property;
```

`XGetWMSizeHints` can generate BadWindow, BadAtom, or BadValue errors.

`XGetWMSizeHints` supersedes `XGetSizeHints`.

`XSetWMSizeHints` replaces (or sets if the property does not exist) the size hints for indicated property on the specified window. The property is stored with a type of WM_SIZE_HINTS and a format of 32. To set a window's normal size hints, the routine `XSetWMNormalHints` may be used instead.

```
void XSetWMSizeHints (dpy, w, hints, property)
    Display *dpy;
    Window w;
    XSizeHints *hints;
    Atom property;
```

`XSetWMSizeHints` can generate BadWindow, BadAtom, or BadValue errors.

`XSetWMSizeHints` supersedes `XSetSizeHints`.

`XGetWMNormalHints` returns the size hints stored in the WM_NORMAL_HINTS property on the specified window. If the property is of type WM_SIZE_HINTS and format 32 and is long enough to contain a size hints structure, the various fields of the hints_return structure are set and a non-zero status is returned. Otherwise, a status of 0 is returned.

```
Status XGetWMNormalHints (dpy, w, hints_return)
    Display *dpy;
    Window w;
    XSizeHints *hints_return;
```

`XGetWMNormalHints` can generate BadWindow, BadAtom, or BadValue errors.

`XGetWMNormalHints` supersedes `XGetNormalHints`.

`XSetWMNormalHints` replaces (or sets if the property does not exist) the size hints for the WM_NORMAL_HINTS property on the specified window. The property is stored with a type of WM_SIZE_HINTS and a format of 32.

```
void XSetWMNormalHints (dpy, w, hints)
    Display *dpy;
    Window w;
    XSizeHints *hints;
```

`XSetWMNormalHints` can generate BadWindow, BadAtom, or BadValue errors.

1.5.6.2 XStandardColormap – Two new elements were added in UWS Release Version 2.2 to properties of type RGB_COLOR_MAP: the id of the visual from which the colormap was created, and an arbitrary resource id that indicates whether or not the cells held by this standard colormap should be released by freeing the colormap id or by doing a KillClient on the indicated resource (necessary for allocating out of an existing map). These fields can be added to the end of the existing structure (defined in `Xutil.h`) without disrupting the existing interfaces:

```
typedef struct {
        Colormap colormap;
        unsigned long red_max;
        unsigned long red_mult;
        unsigned long green_max;
        unsigned long green_mult;
        unsigned long blue_max;
        unsigned long blue_mult;
        unsigned long base_pixel;
        VisualID visualid;       /* NEW */
     XID killid;          /* NEW */
} XStandardColormap;
```

In addition, a new symbolic value has been added to `Xutil.h` indicating resources have been released by freeing the colormap:

```
            #define ReleaseByFreeingColormap ((XID) 1L)
```

1.5.6.3 XTextProperty – Many of the properties of type STRING were changed in UWS Version 2.2 to allow a variety of types and formats. Since the data stored in these properties are no longer simple null-terminated strings, a new data structure describes the encoding, type, length, and value of the text as well as its value. The following structure was added to `Xutil.h`:

```
typedef struct {
    unsigned char *value;      /* property data */
    Atom encoding;             /* type of property */
    int format;                /* 8, 16, or 32 */
    unsigned long nitems;      /* number of items in valu
} XTextProperty;
```

1.5.6.4 WithdrawnState Constant – Even though interfaces to the WM_STATE property are not being defined, the constant `WithdrawnState` was added to `Xutil.h` with a value of zero.

1.5.7 New Xlib Programming Interfaces

In UWS Version 2.2, the following interfaces were added to Xlib.

1.5.7.1 Allocating Structures for Property Data – The following routines allocate memory for the various data structures reflecting information stored in properties. By allocating these structures at run time, instead of declaring them at compile time, clients avoid memory overwriting problems should additional fields ever be added to these structures.

```
XSizeHints *XAllocSizeHints ()
XStandardColormap *XAllocStandardColormap ()
XWMHints *XAllocWMHints ()
XClassHint *XAllocClassHint ()
XIconSize *XAllocIconSize ()
```

The `XAllocSizeHints`, `XAllocStandardColormap`, `XAllocWMHints`, `XAllocClassHint`, and `XAllocIconSize` routines allocate and return pointers to `XSizeHints`, `XStandardColormap`, `XWMHints`, `XClassHint`, and `XIconSize` structures, respectively. Pointer fields will be set to NULL and all other fields will be set to zero. If sufficient memory is not available, NULL will be returned.

1.5.7.2 Manipulating Top-Level Windows – The following routines change the size or visibility of top-level windows (created as children of the root window).

The `XIconifyWindow` function sends a WM_CHANGE_STATE ClientMessage event with a format of 32 and a first data element of IconicState (as described in the Inter-Client Communications Manual, Section 4.1.4, Changing Window State) to the root window of the specified screen. Window managers may elect to receive this message and, if the window is in its normal state, may treat it as a request to change the window's state from normal to iconic. If the atom WM_CHANGE_STATE cannot be interned, no message is sent, and a status of 0 is returned. A non-zero status is returned if the client message is sent successfully; otherwise, a status of 0 is returned.

```
Status XIconifyWindow (dpy, w, screen)
    Display *dpy;
    Window w;
    int screen;
```

`XIconifyWindow` can generate a BadWindow error.

The `XWithdrawWindow` function unmaps the specified window and sends a synthetic UnmapNotify event to the root window of the specified screen. Window managers may elect to receive this message and treat it as a request to change the window's state to withdrawn. When a window is in the withdrawn state, neither its normal nor its iconic representations is visible. A non-zero status is returned if the UnmapNotify event is sent successfully; otherwise, a 0 is returned.

```
Status XWithdrawWindow (dpy, w, screen)
    Display *dpy;
    Window w;
    int screen;
```

`XWithdrawWindow` can generate a BadWindow error.

The `XReconfigureWMWindow` function does a `ConfigureWindow` on the specified top-level window. If the stacking mode is changed and the request fails with a BadMatch error, the error event is trapped and a synthetic `ConfigureRequestEvent` containing the same configuration parameters is sent to the root of the specified window. Window managers may elect to receive this event and treat it as a request to reconfigure the indicated window.

```
Status XReconfigureWMWindow (dpy, w, screen, mask, changes)
    Display *dpy;
    Window w;
    int screen;
    unsigned int mask;
    XWindowChanges *changes;
```

`XReconfigureWMWindow` can generate BadValue and BadWindow errors.

1.5.7.3 String Lists – The following routines convert between lists of pointers to character strings and text properties.

XStringsToTextProperty sets the specified XTextProperty to be of type STRING (format 8) with a value representing the concatenation of the specified list of null-separated character strings. An extra byte containing NUL (but which is not included in the nitems count) is allocated for the value field of text_prop_return. Storage for this field may be released using XFree. If insufficient memory is available for the new new value string, none of the fields in text_prop_return are set and 0 is returned. Otherwise, a non-zero status is returned.

```
    Status XStringListToTextProperty (list, count, text_prop_retur
        char **list;
        int count;
        XTextProperty *text_prop_return;
```

XTextPropertyToStringList returns a list of strings representing the null-separated elements of text_prop. The data in text_prop must be of type STRING and format 8; multiple elements (such as the strings in a disjoint text selection) are separated by a NUL (encoding 0). The property is not null-terminated. Storage for the list and its contents may be released using XFreeStringList. If insufficient memory is available for the list and its elements, neither of the return values is set and a status of 0 is returned. Otherwise, a non-zero status is returned.

```
Status XTextPropertyToStringList (text_prop, list_return, count_re
    XTextProperty *text_prop;
    char ***list_return;
    int *count_return;

void XFreeStringList (list)
    char **list;
```

XFreeStringList releases memory allocated by XTextPropertyToStringList.

1.5.7.4 Manipulating Text Properties – The following routines are used to obtain and specify information on properties that are intended to hold text. Note that Xlib does not enforce any encoding of textual information.

XGetTextProperty reads the specified property from the window and stores the data in the value field of text_prop_return, the type of the data in the encoding field, the format of the data in the format field, and the number of items of data in the nitems field. The particular interpretation of the property's encoding and data as text is left to the calling application.

```
Status XGetTextProperty (display, window, text_prop_return, property)
    Display *display;
    Window window;
    XTextProperty *text_prop_return;
    Atom property;
```

If the property does not exist on the window, the value field of `text_prop_return` is set to NULL, the encoding field is set to None, the format field is set to 0, and the nitems field is set to 0.

The function returns a non-zero status if it was able to set the fields of text_prop_return, otherwise it returns 0.

`XGetTextProperty` can generate BadWindow, BadAtom, or BadValue errors.

The `XGetWMName`,`XGetWMIconName`, and `XGetWMClientMachine` functions are convenience routines that perform a XGetTextProperty on the properties WM_NAME, WM_ICON_NAME, WM_CLIENT_MACHINE, respectively.

```
    Status XGetWMName (dpy, w, text_prop_return)
    Status XGetWMIconName (dpy, w, text_prop_return)
    Status XGetWMClientMachine (dpy, w, text_prop_return)
        Display *dpy;
        Window w;
        XTextProperty *text_prop_return;
```

`XGetWMName` supersedes `XFetchName`. `XGetWMIconName` supersedes `XGetIconName`.

The `XSetTextProperty` routine replaces (or sets if the property does not exist) the specified property for the indicated window with the data given by the value field of text_prop, the type given by the encoding field, the format given by the format field, and the number of items given by the nitems field.

```
    void XSetTextProperty (dpy, w, text_prop, property)
        Display *dpy;
        Window w;
        XTextProperty *text_prop;
        Atom property;
```

`XSetTextProperty` can generate BadWindow, BadAtom, BadValue, or BadAlloc errors.

`XSetWMName`, `XSetWMIconName`, and `XSetWMClientMachine` are convenience routines that perform an `XSetTextProperty` on the properties WM_NAME, WM_ICON_NAME, and WM_CLIENT_MACHINE, respectively.

```
    void XSetWMName (dpy, w, text_prop)
    void XSetWMIconName (dpy, w, text_prop)
    void XSetWMClientMachine (dpy, w, text_prop)
        Display *dpy;
        Window w;
        XTextProperty *text_prop;
```

XSetWMName supersedes XStoreName. XSetWMIconName supersedes XSetIconName.

XGetCommand reads the WM_COMMAND property from the specified window and returns a string list. If the WM_COMMAND property exists, is of type STRING and format 8, and if there is enough memory to contain the string list, the argvp and argcp fields are filled in with a string list that may be freed with XFreeStringList and a non-zero status is returned. Otherwise, a 0 status is returned.

```
    Status XGetCommand (dpy, w, argvp, argcp)
        Display *dpy;
        Window w;
        char ***argvp;
        int *argcp;
```

1.5.7.5 Size Hints – The following routines obtain and specify size hints:

XGetWMSizeHints returns the size hints stored in the indicated property on the specified window. If the property is of type WM_SIZE_HINTS and format 32 and is long enough to contain a size hints structure, the various fields of the hints_return structure are set and a non-zero status is returned. Otherwise, a status of 0 is returned. To get a window's normal size hints, the routine XGetWMNormalHints may be used instead.

```
    Status XGetWMSizeHints (dpy, w, hints_return, property)
        Display *dpy;
        Window w;
        XSizeHints *hints_return;
        Atom property;
```

XGetWMSizeHints can generate BadWindow, BadAtom, or BadValue errors.

XGetWMSizeHints supersedes XGetSizeHints.

XSetWMSizeHints replaces (or sets if the property does not exist) the size hints for indicated property on the specified window. The property is stored with a type of WM_SIZE_HINTS and a format of 32. To set a window's normal size hints, the routine XSetWMNormalHints may be used instead.

```
    void XSetWMSizeHints (dpy, w, hints, property)
        Display *dpy;
        Window w;
        XSizeHints *hints;
        Atom property;
```

XSetWMSizeHints can generate BadWindow, BadAtom, or BadValue errors.

XSetWMSizeHints supersedes XSetSizeHints.

XGetWMNormalHints returns the size hints stored in the WM_NORMAL_HINTS property on the specified window. If the property is of type WM_SIZE_HINTS and format 32 and is long enough to contain a size hints structure, the various fields of the hints_return structure are set and a non-zero status is returned. Otherwise, a status of 0 is returned.

```
    Status XGetWMNormalHints (dpy, w, hints_return)
        Display *dpy;
        Window w;
        XSizeHints *hints_return;
```

XGetWMNormalHints can generate BadWindow, BadAtom, or BadValue errors.

XGetWMNormalHints supersedes XGetNormalHints.

XSetWMNormalHints replaces (or sets if the property does not exist) the size hints for the WM_NORMAL_HINTS property on the specified window. The property is stored with a type of WM_SIZE_HINTS and a format of 32.

```
    void XSetWMNormalHints (dpy, w, hints)
        Display *dpy;
        Window w;
        XSizeHints *hints;
```

XSetWMNormalHints can generate BadWindow, BadAtom, or BadValue errors.

XSetWMNormalHints supersedes XSetNormalHints.

1.5.7.6 Window Manager Protocols List – The following routines obtain and specify the list of window manager protocols in which the client is willing to participate:

XGetWMProtocols returns the list of atoms stored in the WM_PROTOCOLS property on the specified window. These atoms describe window manager protocols in which the owner of this window is willing to participate. If the property exists, is of type ATOM, is of format 32, and the atom WM_PROTOCOLS can be interned, protocols_return is set to a list that the caller may release with XFree of atoms, count_return is set to the number of elements in list, and a non-zero status is returned. Otherwise, a status of 0 is returned and neither of the return values is set.

```
    Status XGetWMProtocols (dpy, w, protocols_return, count_return)
        Display *dpy;
        Window w;
        Atom **protocols_return;
        int *count_return;
```

XGetWMProtocols can generate BadWindow, BadAtom, or BadValue errors.

The XSetWMProtocols routine replaces (or sets if the property does not exist) the WM_PROTOCOLS property on the specified window to contain the list of atoms given by protocols. The property is stored with a type of ATOM and a format of 32. If the routine is unable to intern the atom WM_PROTOCOLS, a status of 0 will be returned. Otherwise, a non-zero status is returned.

```
Status XSetWMProtocols (dpy, w, protocols, count)
    Display *dpy;
    Window w;
    Atom *protocols;
    int count;
```

XSetWMProtocols can generate BadWindow, BadAtom, BadValue, or BadAlloc errors.

1.5.7.7 Window Manager Colormap Windows List – The following routines are used to obtain and specify the list of windows whose colormaps should be installed by the window manager:

XGetWMColormapWindow returns the list of window identifiers stored in the WM_COLORMAP_WINDOWS property on the specified window. These windows indicate the colormaps that the window manager may need to install for this window. If the property exists, is of type WINDOW, is of format 32, and the atom WM_COLORMAP_WINDOWS can be interned, windows_return is set to a list that the caller may release with XFree of window identifiers, count_return is set to the number of elements in list, and a non-zero status is returned. Otherwise, a status of 0 is returned and neither of the return values is set.

```
Status XGetWMColormapWindow (dpy, w, windows_return, count_ret
    Display *dpy;
    Window w;
    Window **windows_return;
    int *count_return;
```

XGetWMColormapWindows can generate BadWindow, BadAtom, or BadValue errors.

The XSetWMColormapWindows routine replaces (or sets if the property does not exist) the WM_COLORMAP_WINDOWS property on the specified window to contain the list of windows given by colormap_windows. The property is stored with a type of WINDOW and a format of 32. If the routine is unable to intern the atom WM_COLORMAP_WINDOWS, a status of 0 will be returned. Otherwise, a non-zero status is returned.

```
Status XSetWMColormapWindows (dpy, w, colormap_windows, count
    Display *dpy;
    Window w;
    Windows *colormap_windows;
    int count;
```

XSetWMProtocols can generate BadAlloc, BadAtom, and BadWindow errors.

1.5.7.8 Standard Colormaps – The following functions are used to obtain and specify standard colormaps:

The XGetRGBColormaps function returns the RGB colormap definitions stored in the indicated property on the specified window. If the property exists, is of type RGB_COLOR_MAP, is of format 32, and is long enough to contain a colormap definition (if the visualid is not present, the default visual for the screen on which the window is located is assumed; if the killid is not present, None, indicating that the resources cannot be released, is assumed), space for the returned colormaps is allocated and is filled in, and a non-zero status is returned. Otherwise, none of the fields are set and a status of 0 is returned. It is the caller's responsibility to honor the *Inter-Client Communctions Conventions Manual* (ICCCM) restriction that only RGB_DEFAULT_MAP contain more than one definition.

```
Status XGetRGBColormaps (dpy, w, stdcmap_return, count_return, property)
     Display *dpy;
     Window w;
     XStandardColormap **stdcmap_return;
     int *count_return;
     Atom property;
```

XGetRGBColormaps can generate BadAtom and BadWindow errors.

XGetRGBColormaps supersedes XGetStandardColormap.

The XSetRGBColormaps routine replaces (or sets if the property does not exist) the RGB colormap definition in the indicated property on the specified window. The property is stored with a type of RGB_COLOR_MAP and a format of 32. It is the caller's responsibility to honor the ICCCM's restriction that only RGB_DEFAULT_MAP contain more than one definition.

```
    void XSetRGBColormaps (dpy, w, stdcmap, count, property)
        Display *dpy;
        Window w;
        XStandardColormap *stdcmap;
        int count;
        Atom property;
```

XSetRGBColormap can generate BadAlloc, BadAtom, and BadWindow errors.

XSetRGBC supersedes XSetStandardColormap.

1.5.7.9 Convenience Routines – In UWS Release Version 2.2, the following routines were added to provide convenient interfaces to new property formats:

XSetWMProperties provides a convenient interface for setting the essential properties on the specified window for communicating with other clients (particularly window and session managers).

```
void XSetWMProperties (dpy, w, window_name, icon_name, argv, argc,
                 normal_hints, wm_hints)
     Display *dpy;                   /* user's display connection */
     Window w;                       /* window to decorate */
     XTextProperty *window_name;     /* name of application */
     XTextProperty *icon_name;       /* name string for icon */
     char **argv;                    /* command line */
     int argc;                       /* size of command line */
     XSize_hints *normal_hints;      /* size hints in normal state */
     XWM_hints *wm_hints;            /* miscellaneous window manager hi
```

If window_name is non-null, the WM_NAME property is set using XSetWMName. If icon_name is non-null, WM_ICON_NAME is set using XSetWMIconName. If argv is non-null, WM_COMMAND is set using XSetCommand (note that an argc of 0 is allowed to indicate a zero-length command). The hostname of this machine is stored using XSetWMClientMachine. If normal_hints is non-null, WM_NORMAL_HINTS is set using XSetWMNormalHints. If wm_hints is non-null, WM_HINTS is set using XSetWMHints.

XSetWMProperties can return any of errors generated by the routines mentioned above.

XSetWMProperties supersedes XSetStandardProperties.

XWMGeometry combines geometry information (given in the format used by XParseGeometry) specified by you and by the calling program with size hints (usually the ones to be stored in WM_NORMAL_HINTS) and returns the location, size, and gravity (NorthWestGravity, NorthEastGravity, SouthEastGravity or SouthWestGravity) describing the window. If the base size is not set in the hints structure, then the minimum size will be used if set. Otherwise, a base size of 0 is assumed. If no minimum size is set in the hints structure, the base size will be used. A mask of the form returned by XParseGeometry, describe what values come from your specification and whether or not the position coordinates are relative to the right and bottom edges, is returned.

```
int XWMGeometry (dpy, screen, user_geom, def_geom, bwidth, hints,
            x_return, y_return, width_return, height_return,
             gravity_return)
   Display *dpy;                /* user's display connection */
   int screen;                  /* screen on which to do computation */
   char *user_geom;             /* user provided geometry spec */
char *def_geom;                      /* default geometry spec for window */
unsigned int bwidth;                 /* border width */
XSizeHints *hints;                   /* usually WM_NORMAL_HINTS */
int *x_return, *y_return;            /* location; set on successful ret */
int *width_return, *height_return;   /* size; set on successful return */
int *gravity_return;                 /* gravity for these coordinates */
```

Invalid geometry specifications may cause a width or height of 0 to be returned. The caller may pass the address of the hints `win_gravity` field as `gravity_return` to update the hints directly.

`XWMGeometry` supersedes `XGeometry`.

1.5.8 Obsolete Functions in Xlib

The following functions in Xlib became obsolete starting in the UWS Version 2.2 release:

`XGeometry`	superseded by `WMGeometry`
`XGetNormalHints`	superseded by `GetWMNormalHints`
`XGetSizeHints`	superseded by `GetWMSizeHints`
`XGetStandardColormap`	superseded by `GetRGBColormap`
`XGetZoomHints`	no longer supported by ICCCM
`XSetNormalHints`	superseded by `SetWMNormalHints`
`XSetSizeHints`	superseded by `SetWMSizeHints`
`XSetStandardColormap`	superseded by `SetRGBColormap`
`XSetStandardProperties`	superseded by `SetWMProperties`
`XSetZoomHints`	no longer supported by ICCCM

1.5.9 Obsolete Constants in Xutil.h

The following constants became obsolete in `Xutil.h` starting in the UWS Version 2.2 release:

`DontCareState`	no longer supported by ICCCM
`InactiveState`	no longer supported by ICCCM
`ZoomState`	no longer supported by ICCCM

1.5.10 Convenience Functions

The following functions were marked in the UWS Version 2.2 release as convenience functions for window and icon names encoded as STRING properties:

`XFetchName`	generalized by `XGetWMName`
`XGetIconName`	generalized by `XGetWMIconName`
`XSetIconName`	generalized by `XSetWMIconName`
`XStoreName`	generalized by `XSetWMName`

1.5.11 Changes to Xlib Interfaces

In UWS Version 2.2 release, the following changes were made to the Xlib interfaces that corrects omissions in previous releases.

1.5.11.1 Getting Screen Number from Screen Pointer – The screen number of each screen needs to be added to the opaque Screen structure and a macro and function needs to be added to access the index. This allows applications, that use Screen pointers instead of indices to use routines that take screen numbers, which Xlib tends to use instead of screen pointers.

```
int ScreenNumberOfScreen(scr)
int XScreenNumberOfScreen(scr)
    Screen *scr;
```

The `ScreenNumberOfScreen` macro and `XScreenNumberOfScreen` function return the screen index corresponding to the indicated screen pointer.

1.5.11.2 Pixmap Formats – To provide an interface to the Pixmap Format information returned in the connection setup block, the following structure analogous to `XGCValues` needs to be added to `Xlib.h`:

```
typedef struct {
    int depth;
    int bits_per_pixel;
    int scanline_pad;
} XPixmapFormatValues;
```

In addition, the following function for retrieving the information must also be added:

```
XPixmapFormatValues *XListPixmapFormats (dpy, count)
    Display *dpy;
    int *count;
```

The `XListPixmapFormats` function returns an array of `XPixmapFormatValues` structures describing the types of Z format images supported by the display. The argument count is set to the number of pixmap formats supported by the display. The storage for the returned structures may be released by calling `XFree`. If insufficient memory is available, NULL will be returned.

1.5.11.3 Returning Old Error Handlers

1.5.11.3 Returning Old Error Handlers – To make nesting of error handlers possible, the declarations of the routines `XSetErrorHandler` and `XSetIOErrorHandler` return the previous handler, as follows:

```
int (*XSetErrorHandler (handler))()
int (*XSetIOErrorHandler (handler))()
int (*handler)();
```

1.5.11.4 Getting XGCValues from a GC – To make it easier to reuse GCs, a new routine was added in theUWS Version 2.2 release for retrieving the contents of a GC that the `XGCValues` structure can set.

```
Status XGetGCValues (dpy, gc, valuemask, values_return)
    Display *dpy;
    GC gc;
    unsigned long valuemask;
    XGCValues *values_return;
```

The `XGetGCValues` function returns the fields indicated by valuemask of the specified graphics context. The clip mask and dash list (represented by the GCClipMask and GCDashList bits in valuemask) may not be requested. If valuemask contains a valid set of GC mask bits and no errors occur, the requested fields in `values_return` are set and a non-zero status is returned. Otherwise, a status of 0 is returned. Valid GC mask bits are:

Valid GC Mask Bits

GCArcMode	GCBackground
GCCapStyle	GCClipXOrigin
GCClipYOrigin	GCDashOffset
GCFillRule	GCFillStyle
GCFont	GCForeground
GCFunction	GCGraphicsExposures
GCJoinStyle	GCLineStyle
GCLineWidth	GCPlaneMask
GCStipple	GCSubwindowMode
GCTile	GCTileStipXOrigin
GCTileStipYOrigin	

1.6 Intrinsics

While Xlib is ICCCM compliant, the Intrinsics are not. Clients that subclass the Shell widget must recompile in order to work with Xlib.

Display PostScript System 2

The Display PostScript® System, from Adobe Systems, Incorporated, extends the entire PostScript® language to bitmap displays. UWS implements the Display PostScript System as an extension to the UWS server. Using this extension, DECwindows client applications can send both PostScript imaging requests and X requests to the same drawable, using a single network connection.

2.1 Example Programs Using the Display PostScript System

The directory `/usr/examples/dps` contains example programs that use the Display PostScript System. These programs are provided for reference only and are unsupported. Digital assumes no responsibility for the use or reliability of these example programs. No Software Performance Reports (SPRs) will be acknowledged regarding these example programs or their reference pages. For more information about these example programs, see the *Guide to Developing Applications for the Display PostScript System.*

2.2 Additional Documentation

In addition to the documentation described in the Guide to Developing Applications for the Display PostScript System, developers can refer to Adobe's Document Structuring Conventions specification, available free of charge from Adobe Systems, Incorporated (Adobe).

To request Adobe's Document Structuring Conventions specification by electronic mail, send a mail message to the Adobe file server at either of the following network addresses:

Internet:	`ps-file-server@adobe.com`
UUCP:	`...!decwrl!adobe!ps-file-server`

2.3 Fixed Problems Between 2.2 and 4.0

- A problem in output buffering routines has been fixed. Certain sequences of Display PostScript requests caused the client program to crash with a bad pointer reference. The content of the requests did not matter, but the length did. Adding `XFlush()` calls changed the behavior.
- `DPSDestroyContext` would never destroy the context. The context would be rendered unusable, but its resources were not reclaimed.
- `setXrgbactual` operator would sometimes exhibit roundoff errors such that the displayed color was slightly different from the requested color.
- An error in the spacing of text resulted in text that was displayed using prebuilt fonts always being spaced too closely together.
- Text ligatures, characters of two or more letters joined together, are now properly displayed using prebuilts. Previously, they were rendered from outlines.
- Several problems relating to Display PostScript and backing store have been repaired. Notably, output into an obscured window with `backingStore` set to `Always` would not be saved.
- The `colorimage` operator was not working correctly You would get random incorrect results with images that have one or two bits per sample, on an eight-plane display.. Four and eight bits per sample worked correctly.
- There was a problem in the client library that caused incorrect updating of user names with shared or chained contexts.
- Prebuilt fonts were never being used on 100 dpi monitors.
- The following PostScript code segment failed in the `strokepath` operator:

```
200 300 moveto
500 600 lineto
10 setlinewidth

strokepath
```

2.4 setrgbXactual Operator Name Change

The `setrgbXactual` operator has been renamed to setXrgbactual, for consistency in naming with the other `X` specific operators. `setrgbXactual` remains as an alias.

2.5 Known Problems

2.5.1 uappend

There is a problem in fill with `uappend`. The following example defines two overlapping circles. You can fill each and look at them.

```
/ucircle1 newpath -200 200 50 0 360 arc true upath cvlit def
/ucircle2 newpath -150 200 50 0 360 arc true upath cvlit def
```

The following example does what you expect, outlines the two circles:

```
newpath ucircle1 uappend ucircle2 uappend stroke
```

The following example does not do what you expect. It only fills circle1.

```
newpath ucircle1 uappend ucircle2 uappend eofill
newpath ucircle1 uappend ucircle2 uappend
```

2.5.2 Contexts Created Using The Default Colormaps

Display PostScript programs that use `XDPSCreateSimpleContext()` to create display PostScript contexts with the default colormaps will get slightly inferior color rendition. Colors will be chosen from a palette of 64 colors (eight greys) instead of 125 colors. Applications needing to use more than 64 colors can do so by using `XDPSCreateContext()` context that allows using as many colormap cells as necessary.

2.5.3 Changing the Default XStandardColormap

If you who wish to alter the `XStandardColormap` default used by simple display PostScript clients, or you want to free completely the colormap cells used by the `XStandardColormap` you may do so by following the methods described in the ICCCM.

Note

> Before you attempt to change the `XStandarColormap` default, you must be certain that no clients are using `XStandardColormap`.

2.5.4 Automatic PostScript GC

Automatic PostScript garbage collection (gc) is turned on by default. It is a global setting to the server, but it is turned on every time `start` executes, which is each time a client creates a context.

Any client may turn automatic garbage collection off, but doing so turns it off for all clients. It gets turned back on the next time a client creates a context.

There is no way to determine the current state of automatic garbage collection.

Fonts 3

ULTRIX Worksystem Software Version 4.0 (UWS V4.0) includes changes since UWS Version 2.1 in the use of fonts for screen displays. This chapter includes explanatory information on how fonts are installed, named, and organized in directories. It also provides notes on using fonts for applications programmers.

Fonts are installed by UWS (or by individual applications) and are read by the X servers on behalf of applications. A X Consortium standard defines the format in which font sources are distributed, bitmap distribution format (BDF). However, X servers and applications use fonts in a compiled format, which is not standardized.

Most of the UWS V4.0 servers use fonts in the X11 portable compiled font (PCF) format. This represents a change from the UWS V2.1 release, whose compiled fonts were in DECwindows format (DWF) or server natural format (SNF). (The Xgb server for VS3520 and 3540 systems, however, continue to use DWF format.) UWS V4.0 also makes changes to the directories in which fonts are located and expands the directory structure for user-supplied fonts and application-supplied fonts.

UWS V4.0 also adds fonts for use with the X implementation of Display PostScript (XDPS). These fonts have their own format, which is a PostScript-compatible ASCII format. If you have additional PostScript-compatible fonts, you can install them for use with Display PostScript.

If you are a system manager or applications programmer, you should familiarize yourself with how the font format change and new directories might affect your environment.

The sections of this chapter are as follows:

- Fonts and font utilities on the kit
- Font names and aliases
- Default font directories
- Application-specific and custom fonts
- Display PostScript fonts
- Information for applications developers
- Layered applications with fonts

3.1 Fonts and Font Utilities

The UWS V4.0 software includes font sets for 75 dots-per-inch (dpi) and 100 dpi displays.

To use fonts other than those supplied in this kit, you must compile their `.bdf` font source files. Use the font compiler `/usr/bin/dxfc` to create `.pcf` files. (For the Xgb server, use `/usr/bin/dxfc3d` to create `.dwf` files.) Next, use `/usr/bin/dxmkfontdir` to create a list of the fonts in the directory for use by the X server. For more information, refer to `dxfc(1X)` and `dxmkfontdir(1X)` in the *UWS Reference Pages*.

UWS V4.0 also contains 123 MIT X11 Release 3 fonts for 75 dpi displays and fonts for compatibility with X10 applications. The MIT and X10 compatibility fonts are unsupported, and are packaged in the unsupported portion of the software distribution.

3.2 Default Font Directories

To be usable, a font must be installed in a directory on the X server's font search path. The default fonts for UWS V4.0 are the 75 dpi fonts, but you can install and use 100 dpi fonts.

In a change from UWS V2.1, the server's font search path does not include both 75 dpi and 100 dpi font directories. To use the 100 dpi fonts, you must alter the font path (see the subsection "100 dpi Fonts").

This section describes the font directories for the supported 75 dpi and 100 dpi fonts. It also lists the Digital-supplied font directories for UWS V4.0, including those for unsupported fonts and user-supplied fonts.

3.2.1 75 dpi Fonts

The default fonts are the 75 dpi fonts. If you install just the mandatory subsets, you install compiled 75 dpi fonts. The server directory search path for these fonts is as follows:

```
/usr/lib/X11/fonts/decwin/75dpi/
/usr/lib/X11/fonts/MIT/
/usr/lib/X11/fonts/compX10/
/usr/lib/X11/fonts/local/75dpi/
/usr/lib/X11/fonts/apps/75dpi/
/usr/lib/X11/fonts/private/75dpi/
```

When a font is requested (by the Xlib routine `XQueryFont` for example), the DECwindows X server looks for the font first in `/usr/lib/X11/fonts/decwin/75dpi/`, then in `/usr/lib/X11/fonts/MIT/`, and so on.

The 75 dpi fonts for the Xgb server are in DWF format; they are installed in the following directories:

```
/usr/lib/dwf/75dpi
/usr/lib/dwf/compX10
/usr/lib/dwf/mitX11
```

3.2.2 100 dpi Fonts

The installation procedure also allows you to install 100 dpi fonts in an optional subset. The server search path for 100dpi fonts is as follows:

```
/usr/lib/X11/fonts/decwin/100dpi/
/usr/lib/X11/fonts/MIT/
/usr/lib/X11/fonts/compX10/
/usr/lib/X11/fonts/local/100dpi/
/usr/lib/X11/fonts/apps/100dpi/
/usr/lib/X11/fonts/private/100dpi/
```

The 100 dpi fonts for the Xgb server are installed in the following directories:

```
/usr/lib/dwf/100dpi
/usr/lib/dwf/compX10
/usr/lib/dwf/mitX11
```

To use the 100 dpi fonts, you must put them on the X server's search path by editing the workstation's `/etc/ttys` file. The server start-up line looks like the following:

```
:0 "/usr/bin/login -P /usr/bin/Xprompter -C /usr/bin/dxsession" none on
secure window="/usr/bin/Xcfb"
```

Modify it to include the `-fd` and `-dpi 100` switches as follows:

```
:0 "/usr/bin/login -P /usr/bin/Xprompter -C /usr/bin/dxsession" none on
secure window="/usr/bin/Xcfb -fd 100 -dpi 100"
```

In a change from previous releases of UWS, a UWS V4.0 font search path uses only the option specified in the `/etc/ttys` file.

After editing the `/etc/ttys` file, restart the X server to change the default font search path.

3.2.3 Font Directory Contents

The following list describes the contents of the font directories.

`/usr/lib/X11/fonts/decwin/75dpi/` and
`/usr/lib/X11/fonts/decwin/100dpi/`

UWS V4.0 uses these font directories for its supported fonts. Reserve these directories for use by the UWS software (consider them as being for *read only* use). These directories might be moved or deleted in future releases of UWS. These directories are shared in the diskless environment.

`/usr/lib/X11/fonts/MIT/`

This font directory holds fonts distributed by the MIT X Consortium. The MIT fonts are unsupported and future releases of the UWS product might move or delete the directory and its contents. Consider the directory as being for *read only* use. It is shared in the diskless environment. Applications should not install fonts in this directory.

`/usr/lib/X11/fonts/compX10/`

This directory contains unsupported fonts needed for compatibility with X10. The directory and its contents might be moved or deleted with future releases of UWS. Consider this directory for *read only* use. It is shared in the diskless environment. Applications should not install fonts in this directory.

`/usr/lib/X11/fonts/local/75dpi/` and
`/usr/lib/X11/fonts/local/100dpi/`

These font directories are a local version of `.../fonts/decwin`. You can use these directories to install site-specific fonts, such as the corporate logo. The directories are not supported, and the installation process does no maintenance on them. However, the directories are on the default server font path, so fonts placed here will be automatically available. These directories are not intended for use by applications. They are shared in the diskless environment.

`/usr/lib/X11/fonts/apps/75dpi/` and
`/usr/lib/X11/fonts/apps/100dpi/`

Shared applications can install fonts in these directories. Like the app-default file or class names, there is no registry; applications must use unique names to avoid collisions. The directories are unsupported, and no maintenance is performed during installation. However, they are in the default server font path, so fonts placed here will be automatically available. The directories are shared in the diskless environment.

`/usr/lib/X11/fonts/private/75dpi/` and
`/usr/lib/X11/fonts/private/100dpi/`

Applications that install fonts that are licensed on a per-workstation basis can install fonts in these directories. Like app-default files or class names, there is no registry; applications must use unique names to avoid collisions. These directories are unsupported, and no maintenance is performed during installation. These directories are in the default server font path. These directories are not shared in the diskless environment.

`/usr/lib/DPS/outline/`

This directory contains subdirectories for Display PostScript (DPS) fonts. The DPS font directories differ from the previously-described server font directories. The fonts are not `.pcf` files and they are not on the server search path; they are used by DPS only. The `decwin` subdirectory holds the default Display PostScript fonts installed by UWS V4.0. The `local` subdirectory is where you can install site-specific fonts for DPS; this directory is shared in the diskless environment. The `apps` subdirectory is for applications that install fonts for sharing in the diskless environment. The `private` directory is where applications can install fonts that are not shared.

3.2.4 Installation Subsets and Server Font Directories

The following table lists the directories into which installation subsets install supported server fonts.

3.2.5 Supported Fonts

Subset	Font Location(s)
UDTAFM400	`/usr/lib/font/metrics`
ULTAFM400	`/usr/lib/font/metrics`
UDWFONT400	`/usr/lib/X11/fonts/decwin/75dpi`
UWSFONT400	`/usr/lib/X11/fonts/decwin/75dpi`
UDWFONT15400	`/usr/lib/X11/fonts/decwin/100dpi`
UWSFONT15400	`/usr/lib/X11/fonts/decwin/100dpi`
UWS3DFONT400	`/usr/lib/dwf/75dpi, /usr/lib/dwf/100dpi`

The following table lists the directories into which installation subsets install unsupported server fonts.

3.2.6 Unsupported Fonts

Subset	Font Location(s)
UDXUNCOMP400	`/usr/lib/X11/fonts/compX10`
ULXUNCOMP400	`/usr/lib/X11/fonts/compX10`
UDXUNMIT400	`/usr/lib/X11/fonts/MIT`
ULXUNMIT400	`/usr/lib/X11/fonts/MIT`
UWS3DFONT400	`/usr/lib/dwf/mitX11, /usr/lib/dwf/comp`

3.3 Application-Specific and Custom Fonts

The X servers can read fonts in PCF format, compressed PCF format, and BDF format (except for the Xgb server, which reads only DWF format fonts, as in the previous UWS release). It is best to compile fonts into PCF format. Compressed PCF files cause slight performance slow-down; the uncompiled BDF files substantially degrade the server's performance.

The servers do not read:

- Fonts produced by the MIT R3 compiler (whose file suffix is `.snf`).
- Fonts produced by the DS3100 UWS V2.0 and 2.1 (RISC) font compiler (whose file suffix is `.snf`).
- Fonts produced by the VAX UWS V2.0 and UWS V2.1 font compiler (whose file suffix is `.dwf`).

To compile and install BDF fonts:

1. Compile the fonts using the `dxfc` command. (Use the `dxfc3d` command if you have a Xgb server). Put the output in a directory that is on the default font path, such as `/usr/lib/X11/fonts/local/75dpi/`.
2. Use the `dxmkfontdir` utility to create a list of the fonts in the directory. The list goes into a file named `fonts.dir`.

Users can now access the fonts when they log in. To compress fonts into `.pcf.Z` files, use the `compress` utility.

You should place fonts in the default font directories to ensure that your fonts are always available in the X server's font search path. If you choose to install your fonts elsewhere, alter the search path to make the fonts available by editing the `/etc/ttys` file for each workstation.

In the `/etc/ttys file`, add the new directory to the server startup line. Separate multiple font directories with commas. Use the +fp command line switch and the following format to prepend elements to the font path:

```
+fp path[,path...]
```

Use the fp+ switch to append elements to the font path:

```
fp+ path[,path...]
```

For example:

```
:0 "/usr/bin/login -P /usr/bin/Xprompter -C /usr/bin/dxsession" none on
secure window="/usr/bin/Xcfb fp+ /udir/susan/toyfonts/"
```

Note that if the `/etc/ttys` file is ever deleted, as it could be during a system software update, the X server will be unable to find these fonts until you again edit the file. For this reason, using the Digital-supplied font directories is the preferred practice (refer to "Font Directory Contents" in a preceding section for descriptions of the font directories).

3.4 Display PostScript Fonts

By installing the UWS V4.0 software, you obtain all of the fonts you need to use Display PostScript. These fonts are in the directory `/usr/lib/DPS/outline/decwin`.

If you want to install additional PostScript-compatible ASCII fonts to use with Display PostScript, place them in the directories listed in the section "Default Font Directories." Restart the X server after installing the fonts.

3.5 Information for Applications Developers

If you distribute fonts with your application, they must be recompiled for UWS V4.0. Applications should ship fonts in BDF format and compile them as part of the installation. This is necessary because the format of compiled fonts is variable. For example, the VMS and ULTRIX operating systems have different formats for compiled fonts and, as noted previously, this version of UWS uses a format that differs from that of the previous version of UWS. Compiling fonts as part of the installation protects your application from future format changes. For information on compiling and installing fonts, refer to "Application-Specific and Custom Fonts."

3.6 Font Names and Aliases

Font names consist of a series of parameter values separated by dashes, describing the typographic characteristics of the font. However, by using aliases, you can use a set of less cumbersome names.

3.6.1 Font Names

UWS V4.0 uses the naming convention specified by the standard *X Logical Font Description Conventions, X Windows System, Version 11*. UWS font names are standard logical font descriptions for the X protocol (XLFDs). A sample font name is as follows:

```
-ADOBE-ITC Avant Garde Gothic-Book-R-Normal--14-100-100-100-P-80-ISO88
```

In left-to-right order, these are the fields in the font name and the values in the sample name:

Field	**Sample Value**
Foundry	Adobe
Typeface family style	ITC Avant Garde Gothic
Weight	Book Weight
Slant (Roman (R), Italic (I), Oblique (O), Reverse Italic (RI),or Reverse Oblique (RO))	Roman
Pixel Size	14
Point Size, in decipoints	100
Resolution, horizontal and vertical, in pixels/dots per inch	100 dots per inch

A comparable font for a 75 dpi screen has the following name:

```
-ADOBE-ITC Avant Garde Gothic-Book-R-Normal--10-100-75-75-P-59-ISO8859
```

This font needs 10 pixels to appear as 10 points. This font differs from the previous sample font only in pixel size, resolution, and character width.

You can use wildcards in font names to specify the values of certain font characteristics but allow the server to provide the appropriate values for those that differ on different workstation screens. A question mark (?) wildcard substitutes for a single character, and an asterisk (*) wildcard substitutes for one or more fields. The following font name specifies a 10-point ITC Avant Garde Gothic font of book weight, roman style, and normal spacing for display on either 75 or 100 dpi systems:

```
-ADOBE-ITC Avant Garde Gothic-Book-R-Normal--*-100-*-*-P-*
```

When you use the asterisk, be sure that the substitutions resolve clearly. For example, in the following font name, the leftmost asterisk substitutes for two fields before the 100:

```
-ADOBE-ITC Avant Garde Gothic-Book-R-Normal--*-100-*-P-*
```

The name resolves to two fonts:

```
-ADOBE-ITC Avant Garde Gothic-Book-R-Normal--11-80-100-100-P-80-ISO8859-1
-ADOBE-ITC Avant Garde Gothic-Book-R-Normal--14-100-100-100-P-80-ISO8859-1
```

The first is an 8-point font; the second is a 10-point font. The server chooses one of the fonts. UWS V4.0 servers choose the first font (in ASCII-sorted order), but applications should avoid dependence on this algorithm, as it could change in a future release. For more on specifying fonts from applications, refer to "Information for Applications Developers."

3.6.2 Font Name Aliases

UWS V4.0 font files have lowercase names that indicate the contents of the file, and they have a `.pcf` suffix (except for fonts for the Xgb server, which have a `.dwf` suffix). Applications and defaults files do not reference fonts by font file name; they use font names. Each directory for UWS V4.0 fonts has an alias file, called `fonts.alias`. By default, this file uses a special notation (FILE_NAMES_ALIASES) to define the name of each `.pcf` file in the directory as an alias for the font it contains. For example, the font whose name is

```
DEC-ADOBE-Helvetica-Bold-R-Normal--12-120-75-75-P-70-DEC-ISOLATIN1
```

is stored in the file

```
/usr/lib/X11/fonts/decwin/75dpi/helvetica12.pcf
```

The font can be referred to in Xlib routines or defaults files as either

```
DEC-ADOBE-Helvetica-Bold-R-Normal--12-120-75-75-P-70-DEC-ISOLATIN1
```

or

```
helvetica12
```

You can set up additional aliases in the `fonts.alias` file. For more information, refer to the `dxmkfontdir(1X)` reference page.

3.6.3 Specifying Fonts

An application should use wildcards for display-specific fields in a font name. When the wildcards in the name are resolved, the resulting font then matches both the application's needs and the characteristics of the workstation display.

Resolution and pixel size are characteristics that vary on different displays, and these are good properties for which to use wildcards. For example, an application can specify the FAMILY_NAME, WEIGHT_NAME, and POINT_SIZE properties for a font, omitting RESOLUTION_X, RESOLUTION_Y, and PIXEL_SIZE. This produces the correct physical font size for the workstation's display resolution at startup.

Applications reading font specifications from a defaults file should use `XListFonts` or `XListFontsWithInfo` to query the server for a list of matching fonts. The application itself could then resolve the wildcards before requesting an open font. In this way, the application controls the font selection method.

If the application instead simply passes the font name with wildcards to the server in the argument to an `OpenFont` request, the server resolves the wildcards. The server has a simple selection method that might not produce the font most beneficial to the application. The algorithm for choosing a font from an ambiguous font name pattern is server-dependent.

3.6.4 Font Properties

The font properties generally include the font name fields and other useful global font information, such as the height of capitals (CAP_HEIGHT), calculated weight and setwidth (WEIGHT and SETWIDTH), and so on. All UWS V4.0 fonts (except Terminal) have at least the following properties:

```
FONT_ASCENT         FONT_DESCENT
DEFAULT_CHAR        X_HEIGHT
WEIGHT              POINT_SIZE
FACE_NAME           COPYRIGHT
FAMILY_NAME         FONT_NAME_REGISTRY
FOUNDRY             WEIGHT_NAME
SLANT               SETWIDTH_NAME
ADD_STYLE_NAME      PIXEL_SIZE
RESOLUTION_X        RESOLUTION_Y
AVERAGE_WIDTH       CHARSET_REGISTRY
CHARSET_ENCODING    CHARSET_COLLECTIONS
CAP_HEIGHT          NOTICE
```

Fonts derived from PostScript printer fonts also have the property `_DEC_DEVICE_NAMES`. This property sets up the correspondence between printer fonts requested by a X Display PostScript (XDPS) client (such as a Courier font requested in a PostScript file) and the screen fonts available to the server. During a font lookup, XDPS chooses a font whose property is "PS=Courier" when a client file requests Courier font. Some examples are as follows:

```
_DEC_DEVICE_NAMES "PS=AvantGarde-Demi"
_DEC_DEVICE_NAMES "PS=Century-Schoolbook-Bold-Italic"
```

Any application (and not just Display PostScript applications) can use this property to ensure that a screen font corresponds to a specified PostScript printer font. To do so, the application searches for a screen font with the _DEC_DEVICE_NAMES property specifying that printer font.

3.7 Layered Applications with Bundled Fonts

Any layered application that bundles screen fonts and runs under UWS V2.1 might be affected by changes in the UWS V4.0 font formats and file extensions. If the application bundled the BDF form of the fonts, you can upgrade them, using the `dxfc` command (or the `dxfc3d` command for a Xgb server). However, if the application bundled just the compiled form of the fonts, you cannot automatically upgrade the fonts to use with UWS V4.0.

3.7.1 DECwrite ULTRIX for VAX V1.0

The DECwrite Equation Editor does not work on UWS V4.0. To use the Equation Editor on UWS V4.0, contact your Customer Support Center (CSC) to request revised versions of the Equation Editor screen fonts.

When contacting the CSC, specify that you use DECwrite V1.0 and that you are a UWS V4.0 customer. You will also be asked to specify your installation media type and provide your mailing address.

3.7.2 DECwrite ULTRIX for RISC V1.0

Source files for DECwrite Equation Editor screen fonts are included in the DECwrite kit, but they must be reinstalled to be compiled into the new format. Please contact your Customer Support Center (CSC) for instructions.

3.8 Viewing/Mailing DDIF Files with Missing External References

`DDIF` files created by applications such as `DECwrite`, may contain references to external files. If you try to view a `DDIF` file with `dxvdoc`, `dxmail`, or `libdvr.a` that contains references to external files that cannot be located on your system, the view fails.

`dxvdoc` returns a message such as "Unexpected error converting aggregate" or "CDA could not open file". `dxmail` displays a blank viewer window; no error message. If you have an application using `libdvr.a`, the application will receive an error status such as `CDA$_OPENFAIL`.

If you mail a `DECwrite` file that contains references to a system-library style file, (system-library style files are installed as part of a `DECwrite` installation in `/usr/lib/cda/*.doc_style`), to a user on a system without DECwrite installed, the mail message will not be viewed successfully on the receiving end. The user will see a blank viewer window, and no message in `dxmail`.

System-library style files are not packed up in the mail message because they are assumed to be a system resource that is present on each system. Normally externally referenced files are packed up in the mail, unless the references are stored as "no-copy" such as DECwrite references to system-library style files.

To get around the problem, make sure your files with external references have all of their externally referenced files present on the machine on which you wish to view them.

Use the cdoc utility to determine if your file has any references to external files. The following command converts a DDIF file to analysis format, and then searches the analysis output myfile.doc for any external reference labels:

```
# cdoc -s ddif -d analysis myfile.doc | grep ERF_LABEL
```

If you find that there are references to files not present on your system, copy over the files to be able to successfully view the entire document. If you are missing DECwrite system-library style files, copy them from a system with DECwrite installed from /usr/lib/cda/*.*.doc_style.

DECwindows Toolkit Programming 4

This chapter describes new and changed features pertaining to DECwindows programming toolkit since UWS V2.1.

4.1 New Widgets and Gadgets

This section describes the widgets and gadgets that have been added to the Toolkit since UWS V2.1.

- Pulldown Menu Entry Gadgets
- Color Mix Widget
- Compound String Text Widget

4.1.1 Pulldown Menu Entry Gadgets

With UWS Version 4.0, the DECwindows Toolkit now provides pull-down menu entry gadgets and widgets. As with gadget variants of other widgets, they should be used unless the gadget does not provide enough flexibility.

The PULLDOWN MENU ENTRY GADGET CREATE routine has been added to the Toolkit. This routine creates a pull-down menu entry gadget. See the manual *Guide to the XUI Toolkit: C Language Binding* for a description of this routine.

4.1.2 Color Mix Widget

In UWS Version 2.2 release, the color mixing widget was added. This widget enables end users to define colors. It is a pop-up dialog box with two main components: a color display region and a color mixing tool. The default color display region contains two color tiles: one presents the original color, and the other shows the new color as it is modified by the user. The default mixing tool follows the RGB color model.

Both the color display region and mixing tool can be replaced with custom components by the application. The color values returned to the application are standard X Window System, Version 11, RGB values (0 through 65,535).

Note that the color mixing widget is simply a mechanism for applications to query users for a color. It does not allocate resources (color cells) for the application.

The color mix widget includes the following routines:

- COLOR MIX CREATE - creates color mix widget (low-level create only)
- COLOR MIX GET NEW COLOR - gets current RGB values of new color
- COLOR MIX SET NEW COLOR - sets current RGB values of new color

4.1.2.1 Hue Lightness Saturation (HLS) – In the UWS Version 4.0 release, the Hue Lightness Saturation (HLS) has been added. The XUI color mix widget now supports the HLS color model as well as RGB. This functionality is built into the color mixer. No code changes are required and the same X11 RGB callback information is returned regardless of which color model is used. The impact is only to end users. A new option menu presents the color model choices - HLS and RGB - giving users the ability to switch between models as often as they wish.

There are 11 new resources (10 of which are labels):

- `DwtNcolorModel` color model currently being used. Choices are:

  `DwtColorModelHLS` `DwtColorModelRGB` Default is `DwtColorModelHLS`.

  Note that is assumed that applications probably will only set this resource before the widget is managed (if at all), and allow users to switch color models themselves using the option menu.
- `DwtNhueLabel` label of hue scale widget. Default is "Hue:".
- `DwtNlightLabel` label of lightness scale widget. Default is "Lightness:".
- `DwtNsatLabel` label of saturation scale widget. Default is "Saturation:".
- `DwtNblackLabel` label for zero end of lightness scale widget. Default is "Black".
- `DwtNwhiteLabel` label for 100% end of lightness scale widget. Default is "White".
- `DwtNgrayLabel` label for zero end of saturation scale widget. Default is "Gray".

- `DwtNfullLabel` label for 100% end of saturation scale widget. Default is "Full".
- `DwtNoptionLabel` label for color model option menu. Default is "Color Model: ".
- `DwtNhlsLabel` label for color model option menu HLS option. Default is "HLS".
- `DwtNrgbLabel` label for color model option menu RGB option. Default is "RGB".

4.1.3 Colormix Red, Green and Blue Labels

There is a problem in the toolkit color mixing widget such that the Red, Green, and Blue labels cannot be modified after widget creation. You must specify label changes during widget creation.

4.1.4 Attached Dialog Box Widget

Attached Dialog Box widget does not retain offsets after widget creation. The values returned by `XtGetValues` are zero, instead of what they were when the widgets were created.

4.1.5 Compound String Text Widget

The compound string text widget supports the same routines and attributes as `SText`, except that it uses compound strings instead of C null-terminated strings.

The compound string text widget includes the following routines:

- CS TEXT - create cs text widget (high-level call)
- CS TEXT CREATE - create cs text widget (low-level call)
- CS TEXT CLEAR SELECTION - clears global selection
- CS TEXT GET EDITABLE - obtains current permission state
- CS TEXT GET MAX LENGTH - obtains maximum allowable text length
- CS TEXT GET SELECTION - retrieves global selection
- CS TEXT GET STRING - retrieves current text
- CS TEXT REPLACE - replaces a portion of the current text
- CS TEXT SET EDITABLE - sets permission state
- CS TEXT SET MAX LENGTH - sets maximum allowable text length

- CS TEXT SET SELECTION - makes specified text current global selection
- CS TEXT SET STRING - sets text

4.2 New Routines

This section describes additions to the following routines that have been added since UWS V2.1:

- Low Level Routines
- Compound String Routines
- Cut and Paste Routines
- Convenience Routines

4.2.1 New Resources for Low-Level Toolkit Routines

The XUI Toolkit provided new resources for low-level routines in the UWS V2.2 release. The following table lists some of those new widget attributes and corresponding resource names and provides descriptions of each attribute and their associated default values.

FILE SELECTION WIDGET:

Attribute:	`file_to_extern_proc`
Resource Name:	`DwtNfileToExternProc`
Description:	Converts native internal file names to custom external file names displayed to the user.
Default Value:	NULL
Attribute:	`file_to_intern_proc`
Resource Name:	`DwtNfileToInternProc`
Description:	Converts custom external file names, displayed to the user, to native internal file names.
Default Value:	NULL
Attribute:	`mask_to_extern_proc`
Resource Name:	`DwtNmaskToExternProc`
Description:	Converts native internal directory masks to custom directory masks displayed to the user
Default Value:	NULL

FILE SELECTION WIDGET:

Attribute:	`mask_to_intern_proc`
Resource Name:	`DwtNmaskToInternProc`
Description:	Converts custom external directory masks, displayed to the user, to internal directory masks.
Default Value:	NULL

HELP WIDGET:

Attribute:	`gototopic_label`
Resource Name:	`DwtNgototopicLabel`
Description:	The label for the Go To Topic menu entry.
Default Value:	"Go To Topic"

Attribute:	`gobacktopic_label`
Resource Name:	`DwtNgobacktopicLabel`
Description:	The label for the Go Back push button in the help widget window.
Default Value:	"Go Back"

Attribute:	`visittopic_label`
Resource Name:	`DwtNvisittopicLabel`
Description:	Label for the Visit Topic menu item in the View pull-down menu.
Default Value:	"Visit Topic"

Attribute:	`close_label`
Resource Name:	`DwtNcloseLabel`
Description:	Label for the Exit push button in the help widget window.
Default Value:	"Exit"

Attribute:	`helphelp`
Resource Name:	`DwtNhelphelpLabel`
Description:	The label for the Overview menu item in the Using Help pull-down menu.
Default Value:	"Overview"

Attribute:	`helpontitle_label`
Resource Name:	`DwtNhelpontitleLabel`
Description:	Label for the help widget title bar used in conjunction with the applications name.
Default Value:	"Help On"

HELP WIDGET:

Attribute:	`help_acknowledge_label`
Resource Name:	`DwtNhelpAcknowledgeLabel`
Description:	Label for the Acknowledge push button in the error message box.
Default Value:	"Acknowledge"

Attribute:	`help_on_help_title`
Resource Name:	`DwtNhelpOnHelpTitle`
Description:	Label for the title bar in the Help-on-Help widget.
Default Value:	"Using Help"

Attribute:	`cache_help_library`
Resource Name:	`DwtNcacheHelpLibrary`
Description:	A boolean attribute that specifies whether or not the text of the help library is stored in the help widget's cache memory.
Default Value:	FALSE

Attribute:	`map_callback`
Resource Name:	`DwtNmapCallback`
Description:	The callback routine or routines called when the help widget is mapped.
Default Value:	NULL

MENU WIDGET:

Attribute:	`change_vis_ atts`
Resource Name:	`DwtNchangeVisAtts`
Description:	A boolean specifying whether a menu can modify the visual attributes of its children. When this attribute is FALSE, a menu widget cannot make changes to its children.
Default Value:	TRUE

Attribute:	`menu_exten_last_row`
Resource Name:	`DwtNmenuExtendLastRow`
Description:	The width of the menu (for vertical menus) or the height of the menu (for horizontal menus).
Default Value:	TRUE

POP-UP DIALOG WIDGET:

Attribute:	`auto_unrealize`
Resource Name:	`DwtNautoUnrealize`
Description:	A boolean that specifies whether a dialog box unrealizes itself when it is unmanaged.
Default Value:	FALSE

MESSAGE BOX WIDGET:

Attribute:	`second_label`
Resource Name:	`DwtNsecondLabel`
Description:	The text for a secondary label
Default Value:	NULL

Attribute:	`label_alignment`
Description:	The alignment of the primary label

Default Value:	`AlignmentCenter`
	`AlignmentBegin`
	`AlignmentEnd`

Attribute:	`second_label_alignment`
Resource Name:	`DwtNsecondLabelAlignment`
Description:	The alignment of the secondary label
Default Value:	`DwtAlignmentBeginning`

Attribute:	`icon_pixmap`
Resource Name:	`DwtNiconPixmap`
Description:	The pixmap used for the icon.
Default Value:	"!" for caution box icon "*" for message box icon wait cursor (watch) for work box icon

PUSH BUTTON WIDGET:

Attribute:	`insensitive_pixmap`
Resource Name:	`DwtNinsensitivePixmap`
Description:	The pixmap used when the push button is set to insensitive. Applies only if push button label is specified as pixmap.
Default Value:	NULL

SCROLL BAR CREATE WIDGET:

Attribute:	`show_arrows`
Resource Name:	`DwtNshowArrows`
Description:	A boolean indicating whether or not scrol bar has stepping arrows.
Default Value:	TRUE

STEXT WIDGET:

Attribute:	`user_data`
Resource Name:	`DwtNuserData`
Description:	Any private data to be associated with the textwidget. The XUI Toolkit does not interpret this data.
Default Value:	NULL

TOGGLE BUTTON WIDGET:

Attribute:	`insensitive_pixmap_on`
Resource Name:	`DwtNinsensitivePixmap`
Description:	Displayed when the button state is TRUE and widget is insensitive
Default Value:	NULL
Attribute:	`insensitive_pixmap_off`
Resource Name:	`DwtNinsensitivePixmapOff`
Description:	Displayed when button state is FALSE and widget is insensitive
Default Value:	NULL

4.2.2 New Compound String Routines

The following new compound string routines have been added since the UWS V2.1 release:

- STRING INIT CONTEXT -- initializes the context required by GET NEXT SEGMENT
- STRING FREE CONTEXT -- frees a compound string context structure

For more information see the section "PERFORMANCE OF INIT GET SEGMENT" section in these release notes.

4.2.3 Cut and Paste Routines

The following Cut and Paste routines since UWS Version 2.1 have been added:

- Start Copy to Clipboard -- identical to Begin Copy to Clipboard, except that the time stamp of the event is included
- Start Copy from Clipboard -- indicates that the application is ready to start copying data from the clipboard
- End Copy from Clipboard -- indicates that the application has completed copying data from the clipboard
- Clipboard Register Format -- registers the length of the data for formats not specified by the *Inter-Client Communications Conventions Manual* (ICCCM).

4.2.4 New Convenience Routines

The following convenience routines have been added since UWS Version 2.1:

- ACTIVATE WIDGET CONVENIENCE ROUTINE -- Provides a mechanism for applications to activate a UWS Toolkit pushbutton. This is useful in cases where buttons present actions also available in menus. For example, in DECwindows mail you can reply to a message either by activating the REPLY button or by selecting the Reply menu item. If the menu item is selected, ACTIVATE WIDGET can be used to flash the REPLY button presenting a more consistent UI.
- GET USER DATA CONVENIENCE ROUTINE -- Provides a short-cut for retrieving the widget user data field. Returns the user data associated with the widget.

UWS Version 2.2 release of the XUI Toolkit corrected the binding for number children `DwtNumChildren`.

4.3 Bug Fixes and Other Changes

This section describes problem fixes and miscellaneous changes to the XUI Toolkit since the UWS V2.1 release.

4.3.1 Changes to Existing Convenience Routines

The UWS Version of the XUI Toolkit corrected the binding for number children `DwtNumChildren`.

4.3.2 DEC Windows Resource Manager (DRM)

Both the Intrinsics and the DECWindows resource manager (DRM) may now be initialized as many times as required by the application. This is an extension of the MIT R3 Intrinsics and should not be used by applications that want to remain R3 compatible.

4.3.3 Internal Format of Compound Strings

The internal format of compound strings changed in the UWS V2.2 release. Compound strings are now stored in CDA format. This change is transparent to applications that treat compound strings as opaque entities.

4.3.4 Performance of INIT GET Segment

In the UWS V2.2 release, the change to compound strings significantly decreased the performance of the `DwtInitGetSegment` when used to fetch multiple segments from a compound string. Because of this, the routines STRING INIT CONTEXT and STRING FREE CONTEXT were added and should be used for better performance.

4.3.5 dwtappli.h

Two erroneous declarations are contained in `dwtappli.h` They are:

- `DwtGetSlider`
- `DwtSetSlider`

4.3.6 Font Units

In UWS V2.0, the XUI Toolkit used the QUAD_WIDTH and RESOLUTION properties of a font in determining the font-unit value for a dialog box. In UWS V2.1, V2,2, and V4.0, the XUI Toolkit uses the AVERAGE_WIDTH and RESOLUTION_Y properties. The font-unit value for the default DECwindows font is exactly the same as in UWS V2.0, but it could be different for any other font.

4.3.7 Cutting and Pasting Between, UWS V2.0 and V4.0 Toolkit Applications

UWS V2.0 Toolkit applications and V4.0 Toolkit applications cannot cut and paste text between each other.

If you run one application linked against the UWS V2.0 Toolkit and another application linked against the UWS V4.0 Toolkit, you cannot cut from one and paste to the other. No error message will display. All V2.0 applications share a common clipboard, and all V4.0 applications share a completely separate clipboard. Steps have been taken to ensure that no similar clipboard incompatibility exists among future versions.

4.3.8 Destroy Callback

Unlike other toolkit callbacks, the destroy callback returns only two arguments: widget id and tag. The reason argument is NULL. Applications therefore should avoid setting destroy callbacks to call general callback routines (handling numerous actions such as activate, arm, disarm, and so forth) that depend on a reason argument.

4.3.9 Listbox Dynamic Sizing

The proper way to change the listbox width is through `SetValues.` Listbox does not support dynamic dimension changes. Therefore, placing listboxes inside attached dialog boxes, with attachments to both the left and right side of the attached dialog box, may lead to the items selectable area not spanning the full width of the listbox under certain circumstances. This is because attached dialog boxes dynamically resize their children when the attached dialog box changes size.

Also note that the proper way to change the listbox height is through the `ItemsCount` attribute. Modifying the height attribute will not reconfigure the number of visible items. For example, doubling the listbox height but not modifying the `ItemsCount` attribute results in a listbox only half full of items, with the remaining area left blank.

4.3.10 Help Widget Listbox

In certain circumstances, the help widget's listbox selectable area does not span the entire width of the widget. However, all items may still be selected by simply clicking the mouse button on the item text.

4.3.11 Fileselection Callback

In UWS V2.0, the fileselection widget did not correctly set the dirmask field in the `DwtFileSelectionCallbackStruct`. To determine the directory mask upon widget callback applications were forced to use the `GetValues` mechanism. This problem has been fixed. Applications that use `GetValues` in their callback routines should use the callback field instead to save time.

4.3.12 DECwindows Toolkit and the MIT R3 Intrinsics.

The version number of the Intrinsics supplied as part of the DECwindows kit does not match the MIT R3 Intrinsics. The MIT R3 Intrinsics version number is 11003, while the DECwindows Intrinsics are 7001. Applications or widgets that switch between the two Intrinsics should be aware of this.

The DECwindows routine `XtNameToWidget` does not conform to the MIT R3 Intrinsics. The specification states that the first component of the names parameter is matched against the children of the passed reference widget; the implemtation matches the first component of the names parameter against the reference widget, not the children. Thus to use the DECwindows version the name of the reference widget should be added to the beginning of the name list.

4.3.13 Selection Pushbuttons

Setting the OK and CANCEL pushbutton labels to NULL or to empty strings does not remove the pushbuttons as stated in the documentation, but instead it results in blank labels.

4.3.14 Using Accelerators on Pushbutton and Togglebutton Gadgets.

Only the first gadget child of a widget parent may have a "#" operator, such as #override, in its button accelerator specification. All gadget button accelerators of a widget parent will have the same # operator as the first gadget child.

4.3.15 Generating widget/gadget exposes

Generating widget/gadget exposes by calling `SetValues` without visual changes is not supported.

In previous versions of XUI, some widgets incorrectly redisplayed after `SetValues` whenever the arglist contained a visual field, even if that field did not change. For example, an application could initiate a pushbutton redraw by passing an unchanged borderwidth in `SetValues` if the widget was a child of a dialog box using font units.

Applications can redraw widgets either by changing a visual field or by calling `XClearArea` on the widget window.

4.3.16 Toggle Button Set State Routine

A problem in the toolkit `DwtToggleButtonSetState` affects toggle buttons with on/off pixmaps. If the widget has not been realized, `SetState` correctly updates the toggle button value but NOT the on/off pixmap. When realized, they display the wrong pixmap.

A simple workaround is to use the `SetValues` mechanism instead setting `DwtNvalue` to TRUE. This correctly updates the pixmap as well as the toggle button value regardless of whether the widget is realized.

4.3.17 Toggle Button Gadgets

Preventing toggle button gadgets from redrawing themselves after applications had changed their value via `SetValues`, (although their values had changed) has been fixed. This problem only affected visible toggle button gadgets modified through `SetValues`.

A problem introduced in the R3 intrinsics prevented applications that read in user's X defaults files from opening more than one display has been fixed.

4.3.18 Dialog Box Race Condition

XUI Toolkit dialog boxes perform a `XGrabKey` on the TAB key so that they can synchronously transfer focus to the next child within the Dialog Box. If a Dialog Box receives a TAB key while the Toolkit is filtering events (for example, while another modal dialog box is up), the original Dialog Box does not see the TAB event and never calls `XAllowEvents` to unfreeze the keyboard. If this happens, you must exit the application and restart it to unfreeze the keyboard.

4.3.19 Right To Left Compound Strings

Right to left compound strings are displayed left to right in dialog box title resource.

4.3.20 DwtGetNextSegment Function

There is an error in the ULTRIX documentation concerning the parameters to the compound string utility routine `DwtGetNextSegment`. The `direction_r_to_l` parameter is shown in the ULTRIX version of the

XUI Toolkit Reference Manual as the address of a Boolean. It should be the address of an integer. Passing the address of a Boolean to this routine can lead to unexpected results, including alignment errors on some platforms.

4.3.21 DwtResolvePartOffsets Function

This function is obsolete. The -DNOT_VMS_V1 flag has been removed. Any programs that depended on this function must be aware that the flag is no longer valid. All interfaces (both VAX and RISC) should act as if the flag was set at all times.

4.3.22 Delete sub-menu

If an application destroys the original sub-menu and then immediately updates the field with the new menu using `SetValues`, the sub-menu field is updated with the new widget id. However, because destroy is a two-phase process, the menu does not know its original sub-menu has been destroyed and when the second phase executes (later in the Mainloop), the parent menu is informed that its sub-menu has been destroyed and promptly sets that field to NULL; even though it now points to the new widget. This can result in an "X - NOT A VALID Window" error.

The workaround is to get back to `Mainloop` and wait a sufficient period of time to allow the second phase of destroy to complete (and menu to clear its sub-menu field) before updating using `SetValues`

4.3.23 Size of Core

You should not count on the size of the core part record as a fixed size, but obtain the size at widget initialization time. This is so that you do not have to recompile programs whenever a new version of UWS is released. Indices and part offsets are used to access fields in your own widgets. For further information, see the *Guide to the XUI Toolkit: C Language Binding*.

4.3.24 DwtWidget.h File

The following problem in UWS V2.2 has been fixed in UWS V4.0:

When compiling a program that includes `DwtWidget.h`, or `DECwDwtWidgetProg.h` that is linked to `DwtWidget.h`, fatal syntax errors were reported due to the typedef `Object` not being declared.

4.3.25 Option Menus

There is a restriction on the type of widget that may parent an option menu. Option menus are documented as being able to have a parent that is a shell widget. However, only a menu shell widget may be the parent of an option menu widget. If you attempt to parent an option menu widget with any other type of shell widget an empty, transparent, window will be displayed with no menu button available.

To create an application that only uses one option menu make the option menu be a child of a popup menu. The following code fragment provides an example of this.

```
include <X11/DwtAppli.h>
     .
     .
     .
Widget toplevel, popup, options;
Widget buttons[3];
int num_args;
Arg args[1];

toplevel = XtInitialize("example", "Example",
                (XrmOptionDescRec *) NULL,
                0, &argc, argv);

popup = DwtMenuPopupCreate(toplevel, "Popup1", (ArgList) NULL, 0);

buttons[0] = DwtPushButtonGadgetCreate(popup, "a", (ArgList) NULL, 0);
buttons[1] = DwtPushButtonGadgetCreate(popup, "b", (ArgList) NULL, 0);
buttons[2] = DwtPushButtonGadgetCreate(popup, "c", (ArgList) NULL, 0);
XtManageChildren(buttons, 3);

num_args = 0;
XtSetArg(args[num_args], DwtNsubMenuId, popup);
num_args++;
options = DwtOptionMenuCreate(popup, "Options", args, num_args);
XtManageChild(options);

XtRealizeWidget(toplevel);
     .
     .
     .
```

4.3.26 Popup Dialog Boxes

There is a problem in the intrinsics that allows popup dialog boxes with no icon button, `DwtNnoIconify` set TRUE, to be initially created iconified `DwtNiconic` TRUE. The "iconified" popup does not have an icon box and cannot be popped up. Additionally, operations such as `SetInputFocus` on the popup will lead to an access violation.

4.3.27 New and Omitted Widget Arguments

Several new and omitted widget arguments have been added since UWS Version 2.2:

attached_dialog_box:	direction_r_to_l
caution_box:	direction_r_to_l
color_mix:	hue_label, light_label, sat_label, black_la white_label, gray_label, full_label, optio
command_window:	direction_r_to_l
dialog_box:	direction_r_to_l
menu_bar:	menu_extend_last_row, direction_r_to_l
popup_menu:	menu_extend_last_row, direction_r_to_l
pulldown_menu:	menu_extend_last_row, direction_r_to_l
radio_box:	menu_extend_last_row, direction_r_to_l
work_area_menu:	menu_extend_last_row, direction_r_to_l
list_box:	spacing, direction_r_to_l
file_selection:	auto_unmanage, auto_unrealize, direction_r_t
selection:	auto_unmanage, auto_unrealize, direction_r_t
help_box:	direction_r_to_l
main_window:	direction_r_to_l
messag_box:	direction_r_to_l
option_menu:	direction_r_to_l
popup_attached_db:	direction_r_to_l
popup_dialog_box:	direction_r_to_l
scale:	direction_r_to_l
scroll_bar:	direction_r_to_l
scroll_window:	direction_r_to_l
seperator:	direction_r_to_l
window:	direction_r_to_l
work_in_progress_box:	direction_r_to_l

4.3.28 Multiple Segment Compound Strings

Multiple segment compound strings in UIL can now be created with the specified or default character set.

4.3.29 Constraint attributes

Constraint attributes are now not allowed on non constraint widgets.

4.3.30 Large Value Tables

Large Value Tables can now fit into DRM Context Buffers.

4.3.31 Setting DEC_KANJI or DEC_HANZI as the default character

Setting DEC_KANJI or DEC_HANZI as the default character set would incorrectly generate the error "support for this character set may be removed in a future release." This has been fixed.

4.4 User Interface (UIL)

The following is a list of the new UIL features since the UWS Version 2.1 Release:

- Continuous image support (color and gray scale)
- Multiple callback procedures per reason
- Direct support of constraint arguments
- Integer tables (for use as tag values in callbacks)
- Support for CS Text widget
- Support for XBITMAPFILE function (reads X bitmap files from disk)
- Support for new compound string routines.

4.5 User Interface Language Compiler

The following arguments are not directly available in the UIL compiler for UWS Version 4.0.

Argument	Data Type	Default Value	Valid for These Objects
auto_unmanage	boolean	True	selection file_selection
auto_unrealize	boolean	False	selection file_selection
direction_r_to_l	integer	DwtDirectionRightDown	color_mix
grab_key_syms	translation_table	Default translation table syntax	color_mix
menu enu_extend_last_row	boolean	True	menu_bar popup_menu pulldown menu radio_box work_area_menu
no_resize	boolean	True	color_mix
take_focus	boolean	True (modal); false (modeless)	color_mix

To set one of these arguments in a UIL module, use the ARGUMENT function to define the argument. See the *Guide to the XUI User Interface Language Compiler* for information on the ARGUMENT function.

4.6 Save-Unders

`Xcfb` and `Xmfb` servers have save-unders turned on by default. `Xqdsg` and `Xqdsm` have save-unders turned off by default.

4.6.1 Turning Save-Unders Off

Save-unders can be turned off by invoking the servers using `-su` option. For example, to turn save-unders off in the Xcfb color server, enter the following:

```
Xcfb -su
```

This is typically done in the file `/etc/ttys`.

4.6.2 Turning Save-Unders On

To turn Save-unders on, invoke your server without using the `-su` option. Check the `/etc/ttys` file.

4.6.3 Memory Considerations

Save-unders do not imply an automatic win. There are cases where an application handles exposures faster than the server. As save-unders require additional server-side memory, it is recommended that you allow applications to handle exposures on low-end systems, such as VS-2000.

User Environment 5

This chapter describes applications that pertain to the user environment.

5.1 UWS Window Manager - dxwm

This section provides notes on naming windows and icons conventions for the `dxwm` window manager.

5.1.1 Naming Windows and Icons

The `dxwm` window manager currently lets you name windows and icons. Client programs can specify their names by using the XA_WM_NAME and XA_WM_ICON_NAME properties defined by DECwindows.

The `dxwm` window manager uses the values of these properties when it decorates the client window or icon. If the client does not specify a value for the XA_WM_ICON_NAME property, `dxwm` uses the one set for the XA_WM_NAME property for the icon as well.

The following example shows you how to define the name of a window by using the Xlib `XChangeProperty` function:

```
main ( ) {
Window win;
int winW, winH;
int winX, winY;
XSetWindowAttributes xswa;
      /*  open the display  */
winW = 600;
winH = 600;
winX = (DisplayWidth(dpy,0)-winW)>>1;
winY = (DisplayHeight(dpy,0)-winW)>>1;
xswa.event_mask = 0
xswa.background_pixel = BlackPixel(dpy,0);
win = XCreateWindow(dpy,RootWindow(dpy,0),winX,winY,winW,winH,0,
      DefaultDepth(dpy,0),InputOutput,DefaultVisual(dpy,0),
      CWEventMask | CWBackPixel, &xswa);
XChangeProperty (dpy,win,XA_WM_NAME,XA_STRING,8,PropModeReplace,
      "My Window",9);
      .
      .
      .
}
```

Notes

1. The XA_WM_NAME property specifies that property is to be changed, while XA_STRING specifies its data type.
2. The argument 8 indicates that the data is in 8-bit format.
3. The `PropModeReplace` argument indicates that the previous associated information is to be discarded. For further information, see the *Guide to the Xlib Library*.
4. The string ''My Window'' is the new name for the XA_WM_NAME property, and the argument 9 indicates that the string has nine characters.
5. To change the icon name, substitute XA_WM_ICON_NAME for XA_WM_NAME.

5.2 Operator Cannot Log into Session Manager

The operator cannot log into the operator account from the `Start Session` login prompt. To invoke operator services, the operator can log in to any account, then use the `su` command to log in as `operator`. You can create a special account for the operator or use the root account. Remember to use the `passwd`(1) command to create a password for the operator account. This is necessary even though the operator does not log directly into the operator account.

5.3 Mouse Support

There is no support for the mouse model number VSXXX-AA on a VAXstation II.

5.4 Calculator - dxcalc

All text in `dxcalc` is now in UIL files and the following features have been since UWS V2.1:

- Context-sensitive help for each key
- Support for standard Edit menu accelerators
- A Customize Menu with standard, savable features.

5.5 Calendar - dxcalendar

The following notes apply to the `dxcalendar` program:

- There has been a change in the database format of the `dxcalendar` program for VAX systems. The `dxcalendar` program informs you of the format change by asking if you want to save the old file. The new format is equivalent to the DECstation 3100 format.
- Repeat entries are lost when converting the data file of `dxcalendar` from UWS Version 2.0 (VAX) to UWS Version 4.0 (VAX or RISC).
- The `dxcalendar` program allows you to define your working hours (using the Customize dialog box). Defining your working hours has two benefits:
    - When `dxcalendar` starts, it aligns the day display such that the start of your working hours is at the top of the display.
    - A dark line (time bar) is displayed on the left and right side of the day display. On the left side, the time bar is expanded to map your timeslots 1:1. On the right hand side, the scrollbar represents the 24 hours and the time bar represents your working day.

5.5.1 Viewing More Than Ten Hours In A Day

At initial startup, `dxcalendar` displays only a few hours at a time in the DayView... dialog box.

To display ten or more hours, use the Day View... dialog box under the Customize menu and change the Increment value to Half Hour. Press the Apply and OK options to confirm your changes. At this point, you may wish to resize the Day View to use most or all of the height of the screen. Then select the Save Settings option under the Customize menu to enable the `dxcalendar` application to start next time with the selected settings.

5.6 Cardfiler - dxcardfiler

All text in `dxcardfiler` is now in UIL files and the following features have been added since UWS V2.1:

- Context-sensitive help for each key
- Support for standard menu accelerators
- A Customize Menu with standard, savable features.

5.7 Clock - dxclock

All text in `dxclock` is now in UIL files and the following features have been added since UWS V2.1:

- Context-sensitive help for each key
- A Customize Menu with standard, savable features
- A display that uses less screen space. The new arrangement allows for use of a smaller window while keeping all the information displayed and readable.
- A 24 hour time display

5.8 Visual Differences Program - dxdiff

Using the `dxdiff` program can be difficult. If used incorrectly, your terminal may freeze and input focus may be lost. Additionally, all commands and controls except those of the window manager are lost. The correct procedure for using the `dxdiff` program is as follows:

1. Select file number one (left)
2. Select file number two (right)
3. From the Differences pulldown menu, select item Do Differences. Note that this may take some time and no clockface becomes visible.

5.9 DECwindows Debugger - dxdb

The `dxdb` program will not allow you to view a module that has the left bracket character (`[`) as a part of its name.

5.10 CDA Viewer - dxvdoc

The menus for the `dxvdoc` program have changed slightly to conform better with other UWS applications. The Set-Up... menu item has moved from the File menu to the Customize menu and has been renamed Options.... It still provides the same functionality, allowing you to change processing options.

The `dxvdoc` program now starts up in the center rather than the upper left of the screen.

5.11 Mail - dxmail

The `dxmail` application does not notify users of problems viewing compound documents. Using the CDA viewer for example, unresolved external references problems result in an empty view window displayed in `dxmail`'s read window. No message is given to the user when a bad file is sent.

In addition, if the `Quit` option is selected from the Continue/Quit/CoreDump&Quit dialog box, all workstation windows will freeze until all `dxmail` windows disappear.

There is a problem in `listbox` such that if resize is Fixed (FALSE) and a wide item is added in the visible region of the `listbox` - the List Box menu does not grow, and the horizontal scroll bar fails to get updated. This behavior only occurs if the added item is visible. A workaround is always to add the item in the non-visible region of `listbox`. For example:

```
(1) DwtListBoxSetPos (top)

(2) DwtListBoxAdditem (do not specify a position, it will be added at
                          the bottom of the list - non-visible area)
```

5.12 NotePad - dxnotepad

The `dxnotepad` application has the following new features:

- Edit
- Search
- Navigate
- Customize

5.12.1 Using The Undo Option

Selecting all the text in `dxnotepad`, indenting +/- 4 spaces and selecting UNDO causes all the text to disappear until you UNDO a second time or REDO once. Because of this behavior, it appears that UNDO loses all your text when it really does not.

5.13 Paint - dxpaint

The following notes pertain to the `dxpaint` program:

- The functions Select_All (from the options menu) and Flood Fill (using the Paint Bucket tool) operate only on the visible portion of the screen.

- The Options menu has changed slightly in appearance. The toggle for writing mode (either transparent or opaque) has been changed to two radio buttons.
- Support for AIL (Applications Interface Library) allowing `dxpaint` to be called by other applications.
- Support for 300 dpi images (images larger than the screen) has been included. Previously, paint could only edit images of size <= size of the screen. Now images of arbitrary size can be edited.
- You can now set output resolution as well as picture size.
- You can use scroll bars to move to any portion of the picture.
- The Fullview option allows you to display the entire picture scaled down, choose (window sized) the area to move to, and crop the picture to an arbitrary size.
- Keyboard accelerators include:
  - Alt/C = Copy
  - Alt/P = Paint
  - Alt/Q = Quit
  - Alt/S = Save
  - Alt/V = Paste
  - Alt/W = Refresh
  - Alt/X = Cut
  - Alt/Z = Undo
- Select All and Flood Fill now act only on the visible portion of the picture and not on the entire picture.
- Transparent/Opaque writing mode now affects spraycan, selection, text in addition to what it already affected.
- Scale Picture options allows you to scale the entire picture.
- Brush and Spraycan use the outline pattern instead of the Fill pattern.

5.13.1 Drawing Rectangles or Squares Using a Small Line Width

Drawing rectangles (or squares) whose width is smaller than the line width, may leave some garbage in the pixmap. If while rubberbanding a rectangle and you decide that you do not want it, go ahead and complete the rectangle by releasing MB1, and then perform an UNDO.

However, if you attempt to rubberband a rectangle whose line is zero width or length, garbage might appear in the pixmap that an UNDO will not clear away.

5.13.2 Specifying a Tilde (~) as Part of a File Specification

The tilde (~) will not be translated as the user's login directory if used as part of the file specification. As a result, if a tilde is used, paint will respond with a message that the file could not be saved (or opened). Always use the full path name when opening or saving files.

A tilde will be parsed as the user's login directory if used as part of the file filter.

5.14 PostScript Previewer - dxpsview

Many PostScript files created by document formatters, such as those used with `ditroff`, adhere to the Adobe page description commenting conventions. You can tell whenever your file is commented by checking to see if the first line is:

```
%!PS-Adobe-1.0
```

Note

> The `dxpsview3d` application for the VAXstation 3520/3540 is the UWS V2.0 `dxpsview3d` application, and should not be confused with the `dxpsview3d` application based on the Display PostScript extension.

The Previewer now lets you view files whose first two characters are not:

```
%!
```

If you open such a file, a warning message asks you to confirm that the file is a PostScript file.

Skipping from page to page is faster on structured files. This works only if the files are structured and commented correctly. If you are unable to preview a file, try toggling Use Comments under the Options menu.

Changes since the UWS V2.1 release:

- You can specify whether `dxpsview` draws images directly to the screen or stores images until an entire page is ready to display using the Watch Progress item from the Options menu.

- You can change the scale by which images are magnified or shrunk. To change the scale, chose the Sheet Selection Item from the Option menu. Please note that if you decide to use this option excessive memory use might result.

5.15 Puzzle - dxpuzzle

All text in dxpuzzle is now in UIL files and the following features have been added since the UWS V2.1 release:

- Context-sensitive help is available for each key
- A Customize Menu with standard, savable features.

5.16 Session Manager - dxsession

For some systems, the Red, Green, and Blue intensity labels that appear in the color selection box that you invoke from the Customize Window dialog box are clipped so that only the tops of the letters are visible. To correct the problem, edit the `/usr/lib/X11/app-defaults/SessionManager` file and delete the following three lines:

```
*Color Attributes.RedScale.height
*Color Attributes.GreenScale.height
*Color Attributes.BlueScale.height
```

5.16.1 Customize Language

The Customize Language dialog box now allows you to specify a language (for example, English US, French) for subsequent applications. The language list also includes Default that indicates a default set of user interface specifications that are installation dependent. These are used if language-specific specifications are unavailable.

If you select a language, the `dxsession` program sets the `xnlLanguage` resource for all applications, and records this setting in the user's `.Xdefaults` file. It also sets the `$LANG` environment variable. If you use the `Default` option, the `xnlLanguage` resource is removed from the root property and the user's .Xdefaults file, and the `$LANG` environment variable is set to the empty string.

Note that the `dxsession`'s use of the `$LANG` variable is inconsistent with the naming conventions used by NLS (Native Language System). This inconsistent use of the `$LANG` variable may cause `XPG3` conforming internationalized applications to behave in an unexpected fashion. For example, applications will not be able to change their locale based on the `dxsession`'s setting of `$LANG`. See the `lang(5int)` reference pages for a description of supported languages.

Note

For a language to take effect, it must be installed and licensed.

5.16.2 Customize Window

You can change the window manager through the Customize Window dialog box that will change the resource value for `sm.windowManagerName` in your `.Xdefaults` file.

If you decide to use the default window manager, the session manager will use the value of the resource `sm.windowManagerName_default`. Typically, this is set only in the app-defaults file for the session manager. If no such value exists, the default window manager value defaults to `/usr/bin/dxwm`.

5.16.3 Setting The Window Screen Background Using The Customize Menu

The Window Screen background pattern option under the Customize menu does not work properly. It does not reflect the pattern until Apply or OK option is selected. It does not update if Default option is selected.

5.17 User Executive - dxue

Since the UWS V2.0 release, the `dxue` application has changed some resources and some of its widget hierarchy. Its class name has changed to `Executive` and its instance name is `argv[0]`, which is the name by which it is invoked. Any resources supplied in your `.Xdefaults` file specific to `dxue` must change to use this new name.

There are some new per-view resources. They are `iconFormat` and `titleFormat`. resources. The value of these are `printf`-like format strings with the following conventions:

```
%v is replaced by the view name
%d is replaced by the current directory
```

To specify a field width, replace *%* with *%n*, where *n* is the number to be used. If you specify a negative number, the field will be left-justified. Otherwise, the field will be right-justified, which is the default. To specify a literal newline, use `\n`. To specify a literal tab, use `\t`.

The `iconMaxDirSegments` and `titleMaxDirSegments` resources provide further control over contents of icons and title bars. They specify how many segments of the current directory will appear in the place of `%d` in a format string. If `iconMaxDirSegments` is 2, then the last two segments of the path will appear in the icon.

Views can now be customized individually by name. The resource name of the background color of the startup view is `*Startup.background`. The resource names of the icon and title bar format resources are also at this level. The names of these resources of the startup view are:

```
*Startup.iconFormat
*Startup.titleFormat
*Startup.iconMaxDirSegments
*Startup.titleMaxDirSegments
```

New views that are not named have the name `Unnamed`. Changing the name of a view does not cause new resources to be applied to it. A renamed view will retain the resources of its original name.

5.18 DECterm Terminal Emulator - dxterm

5.18.1 User Defined Key definitions (UDKs)

UDK definitions supported by `dxterm` allow you to define definitions for shifted function keys (F6 to F20, including Help and Do), using escape sequences. This is described in the *DECterm Text Programming Manual* (part of the EK-DECTERM-DK kit, which also includes the *DECterm Graphics Programming Manual*; this kit can be ordered from DECdirect at 1-800-DIGITAL). UDKs are also described in manuals for video terminals, such as the *VT330/VT340 Programmer Reference Manual.*

Since UDKs use shifted function keys, you can define these keys for any desired purpose without conflicting with the un-shifted function keys that are reserved for operating system and application use. UDKs are designed to be used by user, not by applications. You can choose to lock your UDKs through `dxterm` by selecting the menu item General from the Customize menu, or by using the UDK Definition Device Control String (DCS). Thus, applications cannot count on being able to redefine these keys.

Do not confuse UDKs with operating system features such as the DEFINE/KEY command on the VMS system and termcaps on the ULTRIX system. Those features work with unshifted function keys (or in the case of termcaps, with any keyboard key that sends a known input sequence). UDKs use the shifted function keys that send input sequences that look to the application and operating system as if they were typed by you.

5.18.2 Using UDKs

UDKs work only when `dxterm` is operating in VT300 mode, not VT100 mode or VT52 mode. To check this, go into the Customize General dialog box, select General from the Customize menu, and make sure that the Terminal Mode is VT300 Mode, 7 Bit Controls or VT300 Mode, 8 Bit Controls.

In the description that follows, 8-bit characters are given in terms of their hexadecimal ASCII values using the C notation. For example, 0x41, decimal 65, is the ASCII code for the letter A. The 8-bit ASCII character set is given in the *UWS DECwindows Desktop Applications Guide,*"UPSS DEC Supplemental").

5.18.3 Defining UDKs

UDK definitions are not typed on the keyboard. (If they were, they would be intercepted by the operating system and not seen by `dxterm`). They must be output from the computer to `dxterm`. This can be accomplished in a number of ways. One way is to create a file, using any text editor that allows you to enter non-printing characters such as ESCAPE. Then copy the file to the terminal, using TYPE on the VMS system, or cat(1) on the ULTRIX system. Another way is to output text strings directly containing the UDK definitions; on VMS, from a command file using `write sys$output`, and on the ULTRIX system from a shell script using `echo(1)`. An additional way is to output the strings to the terminal using a program. See the examples below.

To define one or more UDKs, use the following escape sequence:

DCS	Pc;Pl	|	Ky1/St1;...Kyn/Stn	ST
Device Control String Introducer (ASCII 0x90)	Clear and Lock Parameters	Final Character (ASCII 0x7c)	Key Definition String	String Terminator (ASCII 0x9c)

In a 7-bit environment, DCS can be sent as ESC P (ASCII 0x1b and ASCII 0x50, with no space in between) and ST can be sent as ESC\ (ASCII 0x1b and ASCII 0x5c).

Pc is the clear parameter:

0	Clear all keys before loading new values (that is, set them all to empty).
1	Clear just the keys that you are loading.

If you do not specify Pc, it defaults to 0, that is, all keys that are not defined in this device control string will be cleared.

Pl is the lock parameter:

0	Lock the keys. If you want to load new values into the keys later, you must unlock the keys in Customize Gener
1	Do not lock the keys. The keys are unlocked and can be redefined with another DECUDK string.

If you do not specify Pl, it defaults to 0, that is, the keys will be locked after this device control string takes effect.

Note

If Pl is 1 and the keys are already locked, nothing happens. This is because once the keys are locked they can only be unlocked through the Customize General display in `dxterm`, that is they cannot be unlocked from a program.

This sequence clears all UDKs without locking them:

```
DCS 0 ; 1 | ST
```

This sequence locks all UDKs without clearing them:

```
DCS 1 ; 0 | ST
```

5.18.4 Key Definition Strings

Ky1/St1;...Kyn/Stn are the key definition strings; you can have zero or more of these, each of which defines a single shifted function key. Each string consists of a string selector number (Kyn) and a string parameter (Stn), separated by a slash (ASCII 0x2f). A semicolon (ASCII 0x3b) separates different strings.

Note

You cannot define the shifted function key F11 through `dxterm`.

The key selector number (Kyn) indicates which key you are defining:

Key	Value	Key	Value
F6	17	Help	28
F7	18	Do	29
F8	19		
F9	20	F17	31
F10	21	F18	32
		F19	33
F11	23	F20	34
F12	24		
F13	25		
F14	26		

Note that these are not ASCII codes but digits, so the code for F18, 32, means the digit 3 (ASCII 0x33) followed by the digit 2 (ASCII 0x32).

The string parameter (Stn) for each string definition is the encoded definition of the key being defined, that is the sequence of ASCII codes that will be sent to the application. String parameters consist of a series of hexadecimal pairs, one pair for each character in the definition. Each hex pair defines an 8-bit character according to its value in the ASCII table; the hex pair can be upper case (for example, `4E` for the letter "N") or lower case (for example, `4e` for the letter "N").

5.18.5 General Guidelines

You should consider the following guidelines when loading UDKs:

- Before loading new definitions, it is a good idea to clear the old key definitions without locking them and then load the new definitions in another DECUDK string. This will prevent the memory used for UDK definitions from becoming fragmented.
- If you redefine a key, the old definition is lost. This may free up some space if the new definition uses fewer bytes than the old one.
- There are two ways to lock UDKs, but only one way to unlock them. To lock UDKs you can use the Lock UDKs toggle button in Customize General or a DECUDK control string. To unlock UDKs, you must use the Lock UDKs toggle button.
- The default value for each key definition is empty. When you clear UDKs, they are empty.
- Using `dxterm`, you cannot save UDK definitions; they are lost when you exit out of the `dxterm` window. Because of this, it is a good idea to load the key definitions that you want in your login file, LOGIN.COM on the VMS system or `.login` on the ULTRIX system.
- An invalid hex pair in a DECUDK string stops a UDK load sequence. When a load sequence stops (due to an error or other cause), `dxterm` saves any keys already loaded and displays the rest of the DECUDK sequence on the screen.

5.18.6 Example of an ULTRIX Shell Script

The following example is an ULTRIX shell script that demonstrates how to define more than one shifted function key in the same DECUDK Device Control String (DCS). Note that DCSs can continue over more than one line, as shown in this program. This program was written to execute in VT300 mode, 7-bit controls.

This example defines the shifted function key F6 to be `ls -l`<CR>, where <CR> indicates a carriage return. It also defines the shifted function key F7 to be `date`<CR>. In the example that follows ^[is the escape character as it appears when inserted using the text editor vi(1). To enter the escape character in `vi`, while in insert mode, first enter CRTL/V, then press the escape key.

Once you execute the shell script the shifted functions keys F6 and F7 will be defined.

The ULTRIX shell script follows:

```
echo '^[P'                # DCS Introducer
echo '1;'                 # Pc = 1, Clear only keys that are defined
echo '1'                  # Pl = 1, Do not lock the shifted function k
echo '|'                  # | = Final Character
echo '17/6c73202d6c0d'    # Ky1/St1 = F6/ls -l<CR>
echo ';18/646174650d'     # Ky2/St2 = F7/date<CR>
echo '^[\'                # ST = String Terminator
```

5.18.7 Example of an ULTRIX C Program

The following example shows how to define these same two keys, function key F6 and function key F7, from a C program instead of a shell script on ULTRIX.

```
#define ESCAPE '\033'

main()
    {

    /*
     * Send the UDK introducer that does not clear or lock UDK
     * definitions.  Remember that the default for the clear and
     * lock parameters is 0, so if these parameters are omitted th
     * UDKs will be both cleared and locked.
     */

    printf( "%cP1;1|", ESCAPE );

    /*
     * Define shift-F6 to be "ls -l" terminated with a Return.
     */

    printf( "17/6c73202d6c0d" );

    /*
     * Define shift-F7 to be "date" terminated with a Return.
     */

    printf( ";18/646174650d" );

   /*
* Terminate the DECUDK command with <ESC>\ and send a newline.
*/

printf( "%c\\\n", ESCAPE );

}
```

In this example the final `printf` was terminated with a line feed, but in fact line feeds and spaces could have been inserted at any point in the DECUDK device control string except for within the introducer sequence (in this case <ESC>P|). Note that 8-bit characters can be defined as well as 7-bit characters, although 7-bit ASCII has been used for convenience in the above examples.

5.18.8 Known Problems

5.18.8.1 Using VTX with ReGIS Commands – While using `VTX` with `ReGIS` commands, `dxterm` will freeze the terminal and lose input focus This can be avoided by entering the `DCL` command `SET TERMINAL/NOREGIS` before running `VTX`.

The **-xrm** option, of `dxterm` that specifies a resource string to be used does not work properly.

5.18.8.2 dxterm Does Not Clear Out /etc/utmp – Do not end your login session by selecting the quit menu item under the session menu item through the Session Manager if there are one or more dxterms running. First logout out of each dxterm, then select the quit menu item. Failure to follow this procedure will leave `ttyXX` entries in `/etc/utmp`. Thus, users remotely logged into your workstation will see erroneous data when using commands such as `w(1)` and `who(1)`.

5.18.8.3 Using ioctl With sigio Will Hang dxterm – `ioctl` combined with attempting to use the `sigio` signal will hang the `dxterm` on a workstation and will eventually hang the system such that nothing works (the network and so forth) resulting in having to reset and reboot the system. The arguments to the `ioctl` and `fnctl` are as follows:

```
if((filed < 0) || (sigvec(SIGIO, &iovec, 0) == -1) ||
                 (fcntl(filed, F_SETFL, FNDELAY|FASYNC) == -1 ) ||
                 (ioctl(filed, FIONBUF, &_nbufcnt) == -1)) {
          return RETERR;
}
```

The program will access the tape and read data. The signal never arrives. The DECterm hangs at this point. If things get severe enough the whole system will hang.

5.18.8.4 Using System V Shell (sh5) as Default – A DECterm window does not go away when you attempt to log out if you are using AT&T's System V shell (sh5) as the `/usr/etc/passwd` default shell and start a DECterm session using `dxterm -ls`.

You can only remove the window from the screen by iconifying it or ending the session.

UWS Documentation 6

The notes in this section pertain to the ULTRIX documentation that was shipped as part of your ULTRIX Worksystem Software, Version 4.0. This section discusses:

- Online Software Product Description (SPD)
- Xlib Manual Additions
- Macros Described in Appendix C of the Xlib Manual
- Discrepancies between DECwindows Toolkit and the Toolkit documentation.

6.1 Online Software Product Description (SPD)

An online copy of the UWS V4.0 Software Product Description is provided as a reference to the software and hardware configurations that UWS V4.0 supports. The online SPD is located in the directory `/usr/etc`. The online SPD file name is `spd_uws`.

Note

This electronic copy of the SPD is to be used only as a guide, and is not warranted to be accurate or complete, nor is it to be used as a substitute for the printed SPD that comes with your software. The printed SPD is the legal document listing supported software components and supported hardware configurations for your distribution.

6.2 Xlib Manual Additions

The following functions were added in the UWS Version 2.2 Release to the *Guide to the Xlib Library.*

To obtain the visual ID from a `Visual`, use `XVisualIDFromVisual`.

```
VisualID XVisualIDFromVisual(visual)
       Visual *visual;
```

visual Specifies the visual type.

The `XVisualIDFromVisual` function returns the visual ID for the specified visual type.

To obtain the legal KeyCodes for a display, use `XDisplayKeycodes`.

```
XDisplayKeycodes(display, min_keycodes_return, max_keycodes_retı
        Display *display;
        int *min_keycodes_return, max_keycodes_return;
```

display Specifies the connection to the X Server.

minu_keycodes_return
Returns the minimum number of KeyCodes.

max_keycodes_return
Returns the maximum number of KeyCodes. The `XDisplayKeycodes` function returns the min-keycodes and max-keycodes supported by the specified display. The minimum number of KeyCodes returned is never less than 8, and the maximum number of Keycodes returned is never greater than 255. Not all KeyCodes in this range are required to have corresponding keys.

To obtain a pointer to the resource manager string of a display, use `XResourceManagerString`

```
char *XResourceManagerString(display)
        Display *display;
```

display Specifies the connection to the X server.

The `XResourceManagerString` returns the RESOURCE_MANAGER property from the server's root window of screen zero, which was returned when the connection was opened using `XOpenDisplay`.

6.2.1 Macros Described in Appendix C

The following macros were added in the UWS Version 2.2 release to the *Guide to the Xlib Library*, Appendix C.

```
XExtCodes *XAddExtension(display)
        Display *display;
```

For local Xlib extensions, XAddExtension allocates the XExtCodes structure, bumps the extension number count, and chains the extension onto the extension list.

To transmit variable length data, use the Data macros. If the data fits into the output buffer, then this macro copies it to the buffer. If it does not fit, however, the Data macro calls _XSend, which transmits first the contents of the buffer and then your data. The Data macros take three arguments: the Display, a pointer to the beginning of the data, and the number of bytes to be sent.

```
Data(display, (char *) data, nbytes);

Data16(display, (short *) data, nbytes);

Data32(display, (long *) data, nbytes);
```

Data, Data16, and Data32 can use their last argument more than once, so that argument should be a variable rather than an expression such as ''nitems*sizeof(item)''. This kind of computation should be done in a separate statement before calling the macros.

Use the appropriate macro when sending byte, short, or long data.

6.2.2 XRead Functions

If there is variable length data after the reply, change the True to False, and use the appropriate _XRead function to read the variable length data.

_XRead reads the specified number of bytes into data.

```
_XRead(display, data, nbytes)
        Display *display;
        char *data;
        long nbytes;
```

_XRead16 reads the specified number of bytes, unpacking them as 16-bit quantities, into the specified array as shorts.

```
_XRead16(display, data, nbytes)
        Display *display;
        short *data;
        long nbytes;
```

_XRead32 reads the specified number of bytes, unpacking them as 32-bit quantities, into the specified array as longs.

```
_XRead32 (display, data, nbytes)
        Display *display;
        long *data;
        long nbytes;
```

`_XRead16Pad` reads the specified number of bytes, unpacking them as 16-bit quantities, into the specified array as shorts. If the number of bytes is not a multiple of four, `_XRead16Pad` reads up to three additional pad bytes.

```
_XRead16Pad (display, data, nbytes)
        Display *display;
        short *data;
        long nbytes;
```

`_XReadPad` reads the specified number of bytes into data. If the number of bytes is not a multiple of four, `_XReadPad` reads up to three additional pad bytes.

```
_XReadPad (display, data, nbytes)
        Display *display;
        char *data;
        long nbytes;
```

6.3 Discrepancies between DECwindows Toolkit and the Toolkit documentation.

6.3.1 XtRegisterClass

The DRM Register Class routine `XtRegisterClass` is documented to have the widget class parameter passed by reference. The code currently requires it to be passed by value.

6.3.2 XtDisplayInitialize

The Display Initialize routine `XtDisplayInitialize` is incorrectly documented as showing the parameter `display_name` to be a character string passed by descriptor. It should be documented as a Display structure pointer passed by value.

6.4 XUI Toolkit Manual

On page 10-7 in the XUI Toolkit, Programming Volume 2 there is an error in the definition of the second argument of `DwtGetNextSegment(...)`. It is incorrectly listed as:

```
char *text_return
```

It should be:

```
char **text_return
```

6.5 UID File Descriptions

There are no descriptions of the files in the directory `/usr/lib/X11/uid/`. The files in this `uid` directory are compiled versions of the User Interface Language (UIL) files that define the form (menus, buttons, command callbacks, fonts, geometry, etc.) for the functionality of the applications.

Additionally, the UWS V4.0 reference pages for various applications do not mention the uid files, even though some other files related to the applications, in particular the app-defaults files, are shown.

6.6 X Server Reference Pages

Two Section 8 reference pages based on the VAX and RISC versions of `X(8X)` have been added to the UWS reference pages inventory since the UWS Version 2.1 release. The reference pages are as follows:

- `Xmfb(8X)` - RISC version with cross-references from `Xcfb`.
- `Xqvsm(8X)` - VAX version with cross-references from `Xqdsg` and `Xgb`.

Addressed Problems A

Appendix A contains some of the Software Performance Reports (SPRs) that have been addressed for this release of ULTRIX Worksystem Software.

A.1 ULTRIX Worksystem Software SPRs

The following table lists some of the UWS addressed problems.

SPR Number	Version	Description
00016	1.1	`xbiff` reference page is incorrect
00039	1.1	VAXstation II mouse support status
00062	2.0	Settable items in `X` and `dx` applications
00070	2.0	`X` Applications and server problem
00072	2.0	`xset(1)` adds slashes to font path
00083	2.0	Dialog box does not track properly
00096	2.0	Segmentation error `F77` from `Xnextevent` call
00098	2.0	Segmentation error `F77` from `Xnextevent` call
00122	2.0	Customized, DECwindows terminates
00134	2.1	`XSelectaSyncInput` at application startup
00138	2.1	Cannot start UNIX System V shell

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-234-1998 using a 1200- or 2400-baud modem from anywhere in the USA, Canada, or Puerto Rico. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital Subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	————	Local Digital subsidiary or approved distributor
Internal*	————	SSB Order Processing - WMO/E15 *or* Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

* For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

ꓤeader's Comments

ULTRIX Worksystem Software
Release Notes
AA-HF07E-TN

ꓒlease use this postage-paid form to comment on this manual. If you require a written reply :o a software problem and are eligible to receive one under Software Performance Report ˎSPR) service, submit your comments on an SPR form.

Γhank you for your assistance.

Please rate this manual:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

What would you like to see more/less of? ______________________________

____

What do you like best about this manual? ______________________________

____

What do you like least about this manual? ______________________________

____

Please list errors you have found in this manual:

Page Description

______ ____

______ ____

______ ____

______ ____

Additional comments or suggestions to improve this manual:

____

____

____

What version of the software described by this manual are you using? __________

Name/Title ______________________________ Dept. __________

Company ______________________________________ Date ________

Mailing Address ____

__________________ Email ________________ Phone __________

---- Do Not Tear – Fold Here and Tape --

digital™

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST–CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OPEN SOFTWARE PUBLICATIONS MANAGER
ZKO3–2/Z04
110 SPIT BROOK ROAD
NASHUA NH 03062–9987

--- Do Not Tear – Fold Here --

**Cut
Along
Dotted
Line**

ULTRIX Worksystem Software

Release Note Addendum

Order Number: AD-HF07E-T1

Product Version:	ULTRIX Worksystem Software Version 4.0
Operating System and Version:	ULTRIX Version 4.0

digital equipment corporation
maynard, massachusetts

Overview

This document contains software and documentation workarounds not documented in the *ULTRIX Worksystem Software Version 4.0 Release Notes*.

Contents

The notes in this addendum discuss:

1 UWS X Servers

The following notes apply to UWS X Servers.

UWS X Servers and Swap Space Configuration

If the system's swap space disk partition fills, an attempt by the server to access a previously unused page in the dynamically allocated portion of its data segment crashes the server with a segmentation fault. This problem can occur at various points during server execution, not just at those related to the allocation of server resources.

The anomaly stems from a peculiarity of ULTRIX virtual memory: Swap space on the disk partition is allocated when the memory is first accessed, and not when the corresponding memory segment is allocated.

Avoid the problem by configuring your system with a larger swap space. Swap space requirements of the applications being used (dependent upon each application's size and complexity) dictate the minimum acceptable swap partition size.

Save-Unders and Backing Store

The server now supports save-unders as well as backing store functionality. Save-unders beautify screen and window refreshing, but increase the execution time of windowing applications using drop-down menus. Run your server both with and without save-unders to decide if a significant performance penalty occurs when using save-unders in your hardware and software configuration. By default, save-unders are disabled on the `Xtm2d` server and enabled on the `Xcfb` server. The availability of save-unders is determined by a command line option to the server executable. The command to invoke the server is usually located in the `/etc/ttys` file. Edit the line in the `/etc/ttys` file where the server is invoked to turn save-unders off or on. The server must be restarted for any changes to take effect.

Enabling Save-Unders

To enable save-unders, invoke the server without the `-su` command line option. For example, to enable save-unders in the `Xcfb` color server, invoke the server with the following command:

```
# Xcfb
```

Disabling Save-Unders and Backing Store

Save-unders can be disabled using the `-su` command line option. Backing store can be disabled using the `-bs` command line option. For example, to disable save-unders and backing store in the `Xcfb` color server, invoke the server with the following command:

```
# Xcfb -su -bs
```

2 DECstation 5000 Model 200PX

The following notes apply only to the DECstation Model 200PX.

PostScript Previewer (DS5000 Model 200PX only)

A problem in the server prevents the PostScript previewer, `dxpsview`, from operating correctly with the default option settings. Prior to selecting a document for display by the previewer, select the Watch Progress option from the Options menu, or scale the page size down by adjusting the Scale Factor in the Sheet Selection menu which is also an option on the Options menu. Please note that the previewer will crash if a document is selected for viewing before one of these two measures is taken.

Line Drawing (DS5000 Model 200PX only)

Lines partially clipped by the physical limits of the screen are not positioned correctly. In most cases, clipped lines are offset one or two pixels from their intended position.

Additionally, thin (zero width) lines with coincident endpoints are not drawn, regardless of the cap style specified.

Application Screen Font (DS5000 Model 200PX only)

Because the DECstation5000 Model 200PX uses a 100 dots-per-inch (dpi) screen, programs using 75 dpi fonts do not display correctly. Only those applications utilizing 100 dpi fonts will operate correctly. Upon execution, every application must examine the display characteristics and choose an appropriate (75 or 100 dpi) font resolution. Moreover, the necessary fonts (in the appropriate resolution) must be available to the application.

Pixmaps (DS5000 Model 200PX only)

If the TURBOChannel is heavily loaded (for example, if there is heavy disk or network traffic), and there are more pixmaps allocated than there is available offscreen memory, then there may be some corruption of the pixmaps.

If the TURBOChannel is heavily loaded, image readback from the framebuffer or a pixmap may have some wrong pixels in it.

3 Allocating a Colormap for Use with Display PostScript

The colormap cells allocated in a color cube or gray-scale ramp must be contiguous. `XAllocColorCells` can be used to allocate contiguous planes, but not contiguous color cells. Using non-contiguous color cells produces undefined results.

The following routine allocates *n* contiguous colormap cells, assuming they exist in the specified colormap. Use this routine, rather than `XAllocColorCells`, when defining a colormap or gray-scale ramp for use with Display PostScript code.

```
static Bool
_AllocContiguousCells(dpy, cmap, pixels, npixels)
Display *dpy;
Colormap cmap;
unsigned long *pixels;          /* filled in by routine */
```

```
int npixels;
{
    unsigned long *waste = (unsigned long *)NULL;
    int nwaste = 0;
    Bool contig = False;
    int status, i;
    unsigned long masks = NULL;

    while (!contig) {
        status = XAllocColorCells(dpy, cmap, False, &masks, 0, pixels,
                npixels);
        if (!status)
            break;   /* can't get enough contiguous cells */
        for (i=0; i < (npixels-1); i++) {
            if (pixels[i] + 1 != pixels[i+1]) {
                /* isn't contiguous, keep trying */
                XFreeColors(dpy, cmap, &pixels[i+1], npixels - (i+1), 0);
                if (!waste)
                    waste = (unsigned long *)malloc((i + 1) *
                                sizeof(unsigned long));
                else
                    waste = (unsigned long *)realloc(waste, (nwaste +
                                (i + 1)) * sizeof(unsigned long));
                bcopy(pixels, waste+nwaste, (i+1) * sizeof(unsigned long));
                nwaste += (i+1);
                break;
            }
        }
        if (i == (npixels-1))
            contig = True;
    }
    /* clean up and return 'contig' */
    if (nwaste) {
        XFreeColors(dpy, cmap, waste, nwaste, 0);
        free(waste);
    }
    return (contig);
}
```

Pixmaps Larger Than Size of Screen

Pixmaps that are created that are larger than the size of offscreen memory will cause the server to send an error and cause the client to crash.

A workaround is always to allocate pixmaps that are the same size as the screen.

5 Execute Permissions for /usr/lib/X11/rgb

The file access permissions of the executable file /usr/lib/X11/rgb shipped with V4.0 prohibit its execution. To restore the access permissions and render the program executable, become super-user and issue the following command:

```
# chmod +x /usr/lib/X11/rgb
```

This problem affects both RISC and VAX versions of UWS Version 4.0.

Note

Versions which have the rgb executable in /usr/lib/ rather than /usr/lib/X11 are not affected by this problem.

6 Terminal Emulators and Text Editors

An application using pseudo ttys can take as long as five minutes to appear initially on the display. Pseudo ttys are used by many types of applications, including terminal emulators and text editors. If an application window does not appear on the display within a minute, start a second version of the same application. This second version of the application will usually come up quickly. When the sluggish version of the application finally appears, you can delete it without harming the second version of the application.

7 User Environment

The following notes apply to the UWS user environment.

Window Manager

The window manager may prevent applications from mapping drawables to the display until an existing application window is unfocused and refocused.

Starting dxmail from User Executive Option on a DECstation 3100

Using dxmail invoked from the User Executive Option disables sending or receiving mail. If you run the csh shell, you can work around this problem by invoking dxmail from the command line after putting /usr/new/mh in your PATH. This solution will not work when using the Bourne shell.

PostScript Previewer

Due to a swap space memory limitation, selecting a scale factor greater than 2.0 may crash the Xserver. To avoid the problem, increase the available swap space on your workstation. This is accomplished by allocating a larger disk partition for swap space or allocating additional disk partitions for swap space. In either case, the kernel must be rebuilt for the change in swap space to occur. For information on allocating swap space, see the *Guide to Configuration File Maintenance*.

8 DwtMainWindow Reference Page (DwtNcolormap Attribute)

The `DwtMainWindow` reference page lists the inherited attribute `DwtNcolormap`, but gives no indication on whether a user created color map can be used instead of the `DwtNcolormap` attribute. Any attempts to use a user created color map will be ignored.

9 Fonts

The following notes apply to fonts.

Terminal Font

Since UWS Version 2.1, line drawing characters, glyphs, were removed from the terminal font and moved into a new font called terminal_dectech.

The following fonts are for glyphs:

```
/usr/lib/X11/fonts/decwin/75dpi/terminal_dectech18.pcf

/usr/lib/X11/fonts/decwin/75dpi/terminal_bold_dectech18.pcf
```

Font Aliases Example File

The following example file illustrates a workaround to interoperability problems that might exist when using Digital applications with other non-Digital X servers.

It works by mapping fonts used by Digital's applications to fonts supplied on the MIT X11 R4 tape. Therefore, the following font families must already be installed on the server for this to work:

- Courier
- Helvetica
- New Century Schoolbook
- Symbol
- Terminal
- Times

Font Aliases

This font aliases file example works with any MIT-based X server (just about all X servers).

Note

Use of non-Digital X servers with Digital applications is not supported by Digital.

```
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--10-100-75-75-P-59-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--10-100-75-75-P-56-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--12-120-75-75-P-67-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--14-140-75-75-P-80-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--14-140-75-75-P-77-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--18-180-75-75-P-103-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--18-180-75-75-P-98-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--24-240-75-75-P-138-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--24-240-75-75-P-130-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--8-80-75-75-P-49-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--8-80-75-75-P-46-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--10-100-75-75-P-59-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--10-100-75-75-P-57-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--12-120-75-75-P-69-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--12-120-75-75-P-67-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--14-140-75-75-P-81-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--14-140-75-75-P-78-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--18-180-75-75-P-103-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--18-180-75-75-P-98-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--24-240-75-75-P-138-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--24-240-75-75-P-130-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--8-80-75-75-P-49-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--8-80-75-75-P-47-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--10-100-75-75-P-61-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--10-100-75-75-P-60-ISO8859-1
```

```
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--18-180-75-75-P-105-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--18-180-75-75-P-103-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--24-240-75-75-P-140-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--24-240-75-75-P-138-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--8-80-75-75-P-51-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--8-80-75-75-P-50-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--10-100-75-75-P-61-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--10-100-75-75-P-60-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--12-120-75-75-P-71-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--12-120-75-75-P-69-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--18-180-75-75-P-103-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--18-180-75-75-P-104-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--24-240-75-75-P-139-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--24-240-75-75-P-138-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--8-80-75-75-P-51-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--8-80-75-75-P-50-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--10-100-75-75-P-60-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--10-100-75-75-P-60-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--14-140-75-75-P-81-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--18-180-75-75-P-106-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--18-180-75-75-P-103-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--24-240-75-75-P-139-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--24-240-75-75-P-137-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--8-80-75-75-P-50-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--8-80-75-75-P-50-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--10-100-75-75-P-60-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--10-100-75-75-P-60-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--14-140-75-75-P-81-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--18-180-75-75-P-105-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--18-180-75-75-P-104-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--24-240-75-75-P-140-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--24-240-75-75-P-136-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--8-80-75-75-P-50-ISO8859-1
```

```
-Adobe-"New Century Schoolbook"-Medium-I-Normal--8-80-75-75-P-50-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--10-100-75-75-P-61-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--10-100-75-75-P-66-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--12-120-75-75-P-73-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--12-120-75-75-P-77-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--14-140-75-75-P-85-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--14-140-75-75-P-87-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--18-180-75-75-P-109-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--18-180-75-75-P-113-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--24-240-75-75-P-144-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--24-240-75-75-P-149-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--8-80-75-75-P-51-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--8-80-75-75-P-56-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--10-100-75-75-P-62-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--10-100-75-75-P-66-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--12-120-75-75-P-74-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--12-120-75-75-P-76-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--14-140-75-75-P-85-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--14-140-75-75-P-88-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--18-180-75-75-P-109-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--18-180-75-75-P-111-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--24-240-75-75-P-144-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--24-240-75-75-P-148-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--8-80-75-75-P-52-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--8-80-75-75-P-56-ISO8859-1
-"Bigelow & Holmes"-Menu-Medium-R-Normal--10-100-75-75-P-56-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--10-100-75-75-P-60-ISO8859-1
-"Bigelow & Holmes"-Menu-Medium-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--12-120-75-75-P-70-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--10-100-75-75-P-62-ISO8859-1
-Adobe-Times-Bold-R-Normal--10-100-75-75-P-57-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--12-120-75-75-P-75-ISO8859-1
-Adobe-Times-Bold-R-Normal--12-120-75-75-P-67-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--14-140-75-75-P-90-ISO8859-1
-Adobe-Times-Bold-R-Normal--14-140-75-75-P-77-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--18-180-75-75-P-112-ISO8859-1
-Adobe-Times-Bold-R-Normal--18-180-75-75-P-99-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--24-240-75-75-P-149-ISO8859-1
-Adobe-Times-Bold-R-Normal--24-240-75-75-P-132-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--8-80-75-75-P-52-ISO8859-1
-Adobe-Times-Bold-R-Normal--8-80-75-75-P-47-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--10-100-75-75-P-67-ISO8859-1
-Adobe-Times-Bold-I-Normal--10-100-75-75-P-57-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--12-120-75-75-P-78-ISO8859-1
-Adobe-Times-Bold-I-Normal--12-120-75-75-P-68-ISO8859-1
```

```
-Adobe-"ITC Souvenir"-Demi-I-Normal--14-140-75-75-P-92-ISO8859-1
-Adobe-Times-Bold-I-Normal--14-140-75-75-P-77-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--18-180-75-75-P-115-ISO8859-1
-Adobe-Times-Bold-I-Normal--18-180-75-75-P-98-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--24-240-75-75-P-154-ISO8859-1
-Adobe-Times-Bold-I-Normal--24-240-75-75-P-128-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--8-80-75-75-P-57-ISO8859-1
-Adobe-Times-Bold-I-Normal--8-80-75-75-P-47-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--10-100-75-75-P-56-ISO8859-1
-Adobe-Times-Medium-R-Normal--10-100-75-75-P-54-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--12-120-75-75-P-68-ISO8859-1
-Adobe-Times-Medium-R-Normal--12-120-75-75-P-64-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--14-140-75-75-P-79-ISO8859-1
-Adobe-Times-Medium-R-Normal--14-140-75-75-P-74-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--18-180-75-75-P-102-ISO8859-1
-Adobe-Times-Medium-R-Normal--18-180-75-75-P-94-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--24-240-75-75-P-135-ISO8859-1
-Adobe-Times-Medium-R-Normal--24-240-75-75-P-124-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--8-80-75-75-P-46-ISO8859-1
-Adobe-Times-Medium-R-Normal--8-80-75-75-P-44-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--10-100-75-75-P-59-ISO8859-1
-Adobe-Times-Medium-I-Normal--10-100-75-75-P-52-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--12-120-75-75-P-69-ISO8859-1
-Adobe-Times-Medium-I-Normal--12-120-75-75-P-63-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--14-140-75-75-P-82-ISO8859-1
-Adobe-Times-Medium-I-Normal--14-140-75-75-P-73-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--18-180-75-75-P-104-ISO8859-1
-Adobe-Times-Medium-I-Normal--18-180-75-75-P-94-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--24-240-75-75-P-139-ISO8859-1
-Adobe-Times-Medium-I-Normal--24-240-75-75-P-125-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--8-80-75-75-P-49-ISO8859-1
-Adobe-Times-Medium-I-Normal--8-80-75-75-P-42-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-8-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-180-75-75-C-11-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Medium-R-Normal--28-280-75-75-C-16-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--36-360-75-75-C-22-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-8-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-180-75-75-C-11-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-Normal--28-280-75-75-C-16-ISO8859-1
```

```
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--36-360-75-75-C-22-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-"Double Wide"--14-140-75-75-C-16-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Bold-R-"Double Wide"--18-180-75-75-C-22-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-"Double Wide"--14-140-75-75-C-16
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Bold-R-"Double Wide"--18-180-75-75-C-22
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-8
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-180-75-75-C-11
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Bold-R-Normal--28-280-75-75-C-16
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--36-360-75-75-C-22
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Bold-R-Narrow--14-140-75-75-C-6-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Bold-R-Narrow--18-180-75-75-C-7-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-Narrow--28-280-75-75-C-12-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Bold-R-Narrow--36-360-75-75-C-14-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-Narrow--14-140-75-75-C-6-DEC-DECtech
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Bold-R-Narrow--18-180-75-75-C-7
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Bold-R-Narrow--28-280-75-75-C-12
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Bold-R-Narrow--36-360-75-75-C-14
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Bold-R-Wide--14-140-75-75-C-12-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Bold-R-Wide--18-180-75-75-C-14-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Bold-R-Wide--14-140-75-75-C-12
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Bold-R-Wide--18-180-75-75-C-14
-DEC-DECtech -DEC-Terminal-Bold-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Medium-R-"Double Wide"--14-140-75-75-C-16-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
```

```
-Bitstream-Terminal-Medium-R-"Double Wide"--18-180-75-75-C-22-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Medium-R-"Double Wide"--14-140-75-75-C-16
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Medium-R-"Double Wide"--18-180-75-75-C-22
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-8
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-180-75-75-C-11
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Medium-R-Normal--28-280-75-75-C-16
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--36-360-75-75-C-22
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Medium-R-Narrow--14-140-75-75-C-6-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Medium-R-Narrow--18-180-75-75-C-7-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Medium-R-Narrow--28-280-75-75-C-12-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Medium-R-Narrow--36-360-75-75-C-14-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Medium-R-Narrow--14-140-75-75-C-6
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Medium-R-Narrow--18-180-75-75-C-7
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Medium-R-Narrow--28-280-75-75-C-12
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Medium-R-Narrow--36-360-75-75-C-14
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-DEC-Terminal-Medium-R-Wide--14-140-75-75-C-12-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-Bitstream-Terminal-Medium-R-Wide--18-180-75-75-C-14-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-ISO8859-1
-DEC-Terminal-Medium-R-Wide--14-140-75-75-C-12
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Bitstream-Terminal-Medium-R-Wide--18-180-75-75-C-14
-DEC-DECtech -DEC-Terminal-Medium-R-Normal--14-140-75-75-C-80-DEC-DECtech
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--14-100-100-100-P-80-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--14-100-100-100-P-76-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--17-120-100-100-P-93-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--17-120-100-100-P-88-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--20-140-100-100-P-104-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--20-140-100-100-P-100-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--25-180-100-100-P-138-ISO8859-1
```

```
-Adobe-Helvetica-Medium-R-Normal--25-180-100-100-P-130-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--34-240-100-100-P-183-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--34-240-100-100-P-176-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-R-Normal--11-80-100-100-P-59-ISO8859-1
-Adobe-Helvetica-Medium-R-Normal--11-80-100-100-P-56-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--14-100-100-100-P-81-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--14-100-100-100-P-78-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--17-120-100-100-P-88-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--20-140-100-100-P-103-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--20-140-100-100-P-98-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--25-180-100-100-P-138-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--25-180-100-100-P-130-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--34-240-100-100-P-184-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--34-240-100-100-P-176-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Book-O-Normal--10-80-100-100-P-59-ISO8859-1
-Adobe-Helvetica-Medium-O-Normal--11-80-100-100-P-57-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--17-120-100-100-P-93-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--20-140-100-100-P-105-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--20-140-100-100-P-105-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--25-180-100-100-P-140-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--25-180-100-100-P-138-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--34-240-100-100-P-182-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--34-240-100-100-P-182-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-R-Normal--11-80-100-100-P-61-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--11-80-100-100-P-60-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--17-120-100-100-P-93-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--20-140-100-100-P-103-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--20-140-100-100-P-103-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--25-180-100-100-P-139-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--25-180-100-100-P-138-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--34-240-100-100-P-183-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--34-240-100-100-P-182-ISO8859-1
-Adobe-"ITC Avant Garde Gothic"-Demi-O-Normal--11-80-100-100-P-61-ISO8859-1
-Adobe-Helvetica-Bold-O-Normal--11-80-100-100-P-60-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--14-100-100-100-P-81-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--17-120-100-100-P-89-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--17-120-100-100-P-91-ISO8859-1
```

```
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--19-140-100-100-P-106-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--20-140-100-100-P-103-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--24-180-100-100-P-139-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--25-180-100-100-P-136-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--33-240-100-100-P-180-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--34-240-100-100-P-181-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-R-Normal--11-80-100-100-P-60-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-R-Normal--11-80-100-100-P-60-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--14-100-100-100-P-81-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--19-120-100-100-P-89-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--20-140-100-100-P-105-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--20-140-100-100-P-104-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--24-180-100-100-P-140-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--25-180-100-100-P-136-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--33-240-100-100-P-181-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--34-240-100-100-P-182-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Book-O-Normal--11-80-100-100-P-60-ISO8859-1
-Adobe-"New Century Schoolbook"-Medium-I-Normal--11-80-100-100-P-60-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--14-100-100-100-P-85-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--14-100-100-100-P-87-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--17-120-100-100-P-99-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--19-140-100-100-P-109-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--20-140-100-100-P-113-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--24-180-100-100-P-144-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--25-180-100-100-P-149-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--33-240-100-100-P-184-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--34-240-100-100-P-193-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-R-Normal--11-80-100-100-P-61-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-R-Normal--11-80-100-100-P-66-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--14-100-100-100-P-85-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--14-100-100-100-P-88-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--17-120-100-100-P-99-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--19-140-100-100-P-109-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--20-140-100-100-P-111-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--24-180-100-100-P-144-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--25-180-100-100-P-148-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--33-240-100-100-P-184-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--34-240-100-100-P-193-ISO8859-1
-Adobe-"ITC Lubalin Graph"-Demi-O-Normal--11-80-100-100-P-62-ISO8859-1
-Adobe-"New Century Schoolbook"-Bold-I-Normal--11-80-100-100-P-66-ISO8859-1
-"Bigelow & Holmes"-Menu-Medium-R-Normal--13-100-100-100-P-77-ISO8859-1
```

```
-Adobe-Helvetica-Bold-R-Normal--14-100-100-100-P-82-ISO8859-1
-"Bigelow & Holmes"-Menu-Medium-R-Normal--16-120-100-100-P-92-ISO8859-1
-Adobe-Helvetica-Bold-R-Normal--17-120-100-100-P-92-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--14-100-100-100-P-90-ISO8859-1
-Adobe-Times-Bold-R-Normal--14-100-100-100-P-76-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--17-120-100-100-P-94-ISO8859-1
-Adobe-Times-Bold-R-Normal--17-120-100-100-P-88-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--20-140-100-100-P-112-ISO8859-1
-Adobe-Times-Bold-R-Normal--20-140-100-100-P-100-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--25-180-100-100-P-149-ISO8859-1
-Adobe-Times-Bold-R-Normal--25-180-100-100-P-132-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--34-240-100-100-P-191-ISO8859-1
-Adobe-Times-Bold-R-Normal--34-240-100-100-P-177-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-R-Normal--11-80-100-100-P-62-ISO8859-1
-Adobe-Times-Bold-R-Normal--11-80-100-100-P-57-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--14-100-100-100-P-92-ISO8859-1
-Adobe-Times-Bold-I-Normal--14-100-100-100-P-77-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--17-120-100-100-P-98-ISO8859-1
-Adobe-Times-Bold-I-Normal--17-120-100-100-P-86-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--20-140-100-100-P-115-ISO8859-1
-Adobe-Times-Bold-I-Normal--20-140-100-100-P-98-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--25-180-100-100-P-154-ISO8859-1
-Adobe-Times-Bold-I-Normal--25-180-100-100-P-128-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--34-240-100-100-P-197-ISO8859-1
-Adobe-Times-Bold-I-Normal--34-240-100-100-P-170-ISO8859-1
-Adobe-"ITC Souvenir"-Demi-I-Normal--11-80-100-100-P-67-ISO8859-1
-Adobe-Times-Bold-I-Normal--11-80-100-100-P-57-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--14-100-100-100-P-79-ISO8859-1
-Adobe-Times-Medium-R-Normal--14-100-100-100-P-74-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--17-120-100-100-P-85-ISO8859-1
-Adobe-Times-Medium-R-Normal--17-120-100-100-P-84-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--20-140-100-100-P-102-ISO8859-1
-Adobe-Times-Medium-R-Normal--20-140-100-100-P-96-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--25-180-100-100-P-135-ISO8859-1
-Adobe-Times-Medium-R-Normal--25-180-100-100-P-125-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--34-240-100-100-P-174-ISO8859-1
-Adobe-Times-Medium-R-Normal--34-240-100-100-P-170-ISO8859-1
-Adobe-"ITC Souvenir"-Light-R-Normal--11-80-100-100-P-56-ISO8859-1
-Adobe-Times-Medium-R-Normal--11-80-100-100-P-54-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--14-100-100-100-P-82-ISO8859-1
-Adobe-Times-Medium-I-Normal--14-100-100-100-P-73-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--17-120-100-100-P-88-ISO8859-1
-Adobe-Times-Medium-I-Normal--17-120-100-100-P-84-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--20-140-100-100-P-104-ISO8859-1
-Adobe-Times-Medium-I-Normal--20-140-100-100-P-94-ISO8859-1
```

```
-Adobe-"ITC Souvenir"-Light-I-Normal--25-180-100-100-P-139-ISO8859-1
-Adobe-Times-Medium-I-Normal--25-180-100-100-P-125-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--34-240-100-100-P-177-ISO8859-1
-Adobe-Times-Medium-I-Normal--34-240-100-100-P-168-ISO8859-1
-Adobe-"ITC Souvenir"-Light-I-Normal--11-80-100-100-P-59-ISO8859-1
-Adobe-Times-Medium-I-Normal--11-80-100-100-P-52-ISO8859-1
-DEC-Terminal-Medium-R-Normal--14-100-100-100-C-8-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-11-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Medium-R-Normal--28-200-100-100-C-16-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--36-280-100-100-C-22-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-Normal--14-100-100-100-C-8-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-11-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-Normal--28-200-100-100-C-16-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--36-280-100-100-C-22-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-"Double Wide"--14-100-100-100-C-16-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Bold-R-"Double Wide"--18-140-100-100-C-22-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-"Double Wide"--14-100-100-100-C-16-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Bold-R-"Double Wide"--18-140-100-100-C-22-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Bold-R-Normal--14-100-100-100-C-8-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-11-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Bold-R-Normal--28-200-100-100-C-16-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--36-280-100-100-C-22-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Bold-R-Narrow--14-100-100-100-C-6-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Bold-R-Narrow--18-140-100-100-C-7-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-Narrow--28-200-100-100-C-12-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Bold-R-Narrow--36-280-100-100-C-14-ISO8859-1
```

```
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-Narrow--14-100-100-100-C-6-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Bold-R-Narrow--18-140-100-100-C-7-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Bold-R-Narrow--28-200-100-100-C-12-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Bold-R-Narrow--36-280-100-100-C-14-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Bold-R-Wide--14-100-100-100-C-12-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Bold-R-Wide--18-140-100-100-C-14-ISO8859-1
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Bold-R-Wide--14-100-100-100-C-12-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Bold-R-Wide--18-140-100-100-C-14-DEC-DECtech
-Bitstream-Terminal-Bold-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Medium-R-"Double Wide"--14-100-100-100-C-16-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Medium-R-"Double Wide"--18-140-100-100-C-22-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Medium-R-"Double Wide"--14-100-100-100-C-16-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Medium-R-"Double Wide"--18-140-100-100-C-22-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Medium-R-Normal--14-100-100-100-C-8-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-11-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Medium-R-Normal--28-200-100-100-C-16-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--36-280-100-100-C-22-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Medium-R-Narrow--14-100-100-100-C-6-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Medium-R-Narrow--18-140-100-100-C-7-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Medium-R-Narrow--28-200-100-100-C-12-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Medium-R-Narrow--36-280-100-100-C-14-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Medium-R-Narrow--14-100-100-100-C-6-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Medium-R-Narrow--18-140-100-100-C-7-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
```

```
-DEC-Terminal-Medium-R-Narrow--28-200-100-100-C-12-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Medium-R-Narrow--36-280-100-100-C-14-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-DEC-Terminal-Medium-R-Wide--14-100-100-100-C-12-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-Bitstream-Terminal-Medium-R-Wide--18-140-100-100-C-14-ISO8859-1
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-ISO8859-1
-DEC-Terminal-Medium-R-Wide--14-100-100-100-C-12-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
-Bitstream-Terminal-Medium-R-Wide--18-140-100-100-C-14-DEC-DECtech
-Bitstream-Terminal-Medium-R-Normal--18-140-100-100-C-110-DEC-DECtech
```