L

APPENDIX L

SunView Applications under OpenWindows

If you hang around Sun users for long enough, you'll eventually hear them talking about a window system called "SunView". Before the emergence of X11 and NeWS, most users of Sun Workstations used Sun's own windowing system, called SunView. Indeed, many old-time Sun aficionados will mourn the passing of SunView. It was small and fast[1], which meant you could do useful work on a Sun-3/50 with a mere(!) four megabytes of main storage. And on my SparcStation 1 (the classic SPARC machine) running SunView, *cmdtool* starts up in about one second, while an XView *cmdtool* or an *xterm* takes slightly over two seconds[2]. Most of this overhead is probably due to the networked nature of communication between the program, the X server, and the X window manager. SunView was a full-featured system, with its own (the original) version of *shelltool/cmdtool*, desk accessories such as *perfmeter*, etc. There was even (in SunOS 4.1.1) a SunView *fileview* program, which is similar to the File Manager discussed in Chapter 4, *Using the OPEN LOOK File Manager*. But despite its speed, SunView wasn't able to muster support in the days of networked window systems, so it has had to pass on to that great swap device in the sky.

But wait; SunView lives on! To make OpenWindows serve as a complete working environment for users of modern graphical programs, Sun Microsystems provides full compatibility with applications written for their first-generation window system, SunView. OpenWindows 2.0 introduced, and 3.x continues, the tradition of providing reasonable backwards compatibility for SunView applications. While few new applications are being developed for SunView, if you are making extensive use of a variety of programs on your Sun, you may meet up with a SunView application. Thus you should at least know the basics of running SunView applications under OpenWindows. This appendix shows you

1. But still not as fast as MGR, the free software window system from Steve Uhler of Bellcore. It's fast, but there is no real body of software accumulating for it as there is for The X Window System.

2. All cases were timed after one of these windows was started, to rule out disk- and cache-induced delays

how to use SunView applications under Sun's version of OpenWindows: what works, what doesn't, and what problems can crop up.

L.1 The Similarities

Figure L-1 shows a fairly typical native SunView window; this is how workstation screens looked before The X Window System came along. Like an X display, the screen has two *shelltool* windows, a small one for console messages and a larger one for working in, and a *perfmeter* and a clock. As well, this screendump shows the root menu pulled down to show the items for starting clock windows.

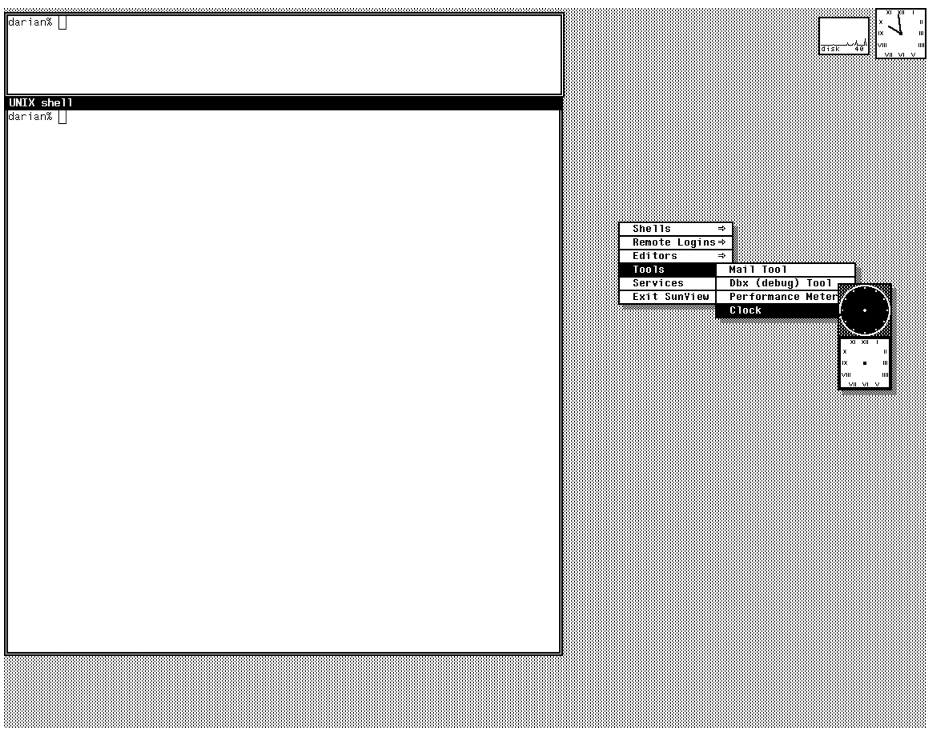

Figure L-1. Typical SunView **Display.**

Notice the following similarities between SunView and OpenWindows:

- there can be arbitrarily many windows, and they overlap in about the same way
- most of the time you do most of your work in one or more shell windows
- *cmdtool/shelltool* is almost identical to its OpenWindows descendants. Even the escape codes in the SunView terminal windows are the same as those for *cmdtool/shelltool* in Appendix E, *Control Sequences for xterm and cmdtool*.

- Cut-and paste is similar, but the terminal pane menu has the added item *send* (combines copy and paste) menu item, which to this day is distressingly absent from the OpenWindows versions (the functionality is there, but not in the menus; see Chapter 2, *Working in the OPEN LOOK Environment*).
- tools like clocks and *perfmeter*s are almost identical to their OpenWindows descendants
- menus are activated by depressing the right button
- root menu, submenus. Menu contents controlled by files, similar in syntax to OPEN LOOK Window Manager
- There are special editors (SunView's *defaultsedit* is the ancestor of OpenWindows's *properties* editor).

L.2 The Differences

The SunView window system was organized quite differently than is X11. It is not a networked window system; it is tightly interwoven with the UNIX kernel, and there is no separate "window manager" client at all. This means that you cannot change the overall behavior nor the "look and feel" of SunView clients, as you can in X11 by changing window managers. And you use "defaults files", not X Resources, to customize SunView programs.

L.2.1 Always at the forefront...

Pity the poor OpenWindows server program! It has to keep track not only of X11 programs' windows, and NeWS programs' windows, but also SunView program windows. X11 and NeWS are similar enough that they can co-exist on one screen. But SunView programs do not respond to the same kind of events, and do not have the same overall behavior, so they cannot really live on the same screen. To sort this out, the OpenWindows server maintains the fiction of an "overlay plane", or separate screen level, to run SunView applications in. The result is that SunView applications will always appear to be "in front of" all X11 and NeWS applications. To visually remind you that a given program is running in SunView mode, the SunView program's windows are surrounded by a fat white border. Figure L-2 is an example of a SunView version of the *dbxtool* program debugger: running in the OpenWindows environment. Notice how the border around the dbxtool window "cuts into" the windows around it, including the Workspace menu, the *xv* control panel being used to record the screen, and the borders of the terminal windows, with a large white space around it. This is your reminder that a SunView application is in use, and that certain things (notably pointer and menu operations) must be done differently. These are described in the next section.

L

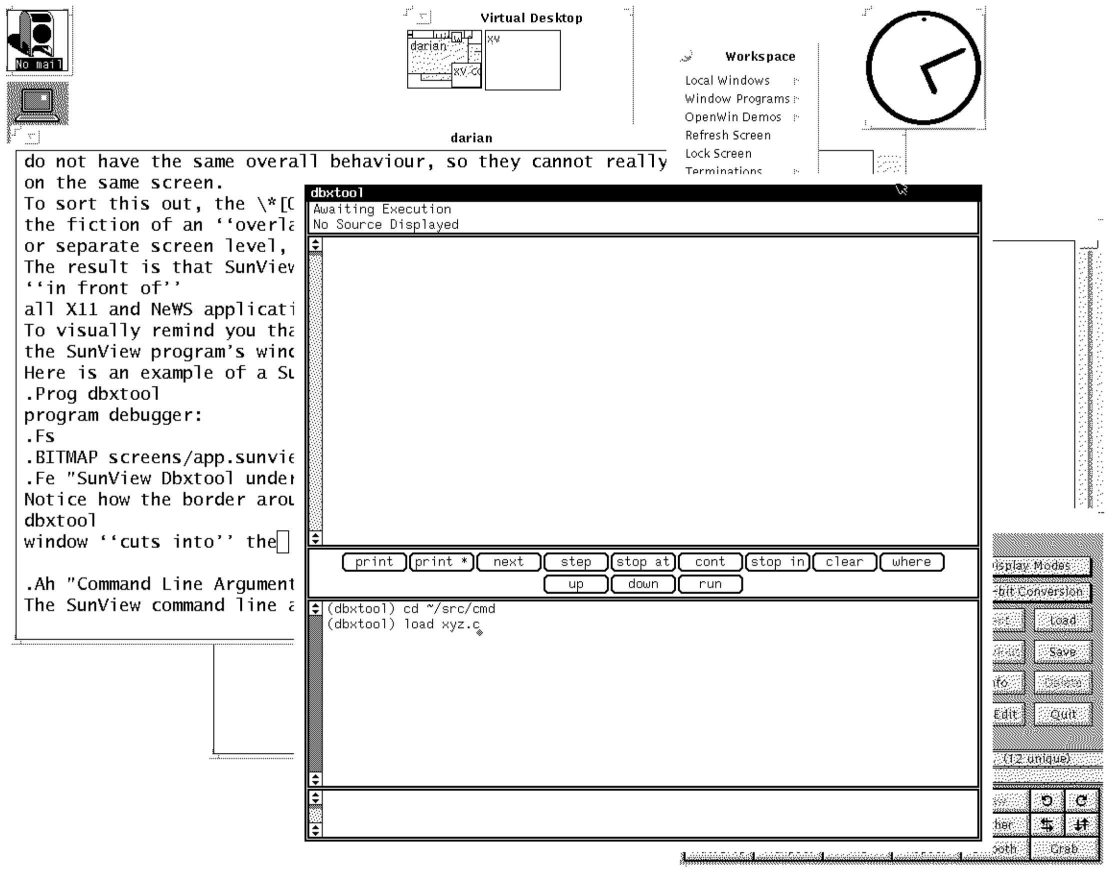

Figure L-2. SunView dbxtool in front of OpenWindows windows.

L.3 Pointer and Menu Conventions

Many of the pointer and menu button conventions are similar or identical to those of OpenWindows. Indeed, when the pointer is not in a SunView window, it behaves "normally". When in a SunView window, titlebar, or icon, however, there are a few differences.

L.3.1 Pointer button conventions

Table L-1 is a summary of the pointer button conventions used when running SunView applications under OpenWindows.

Table L-1. SunView Button Functions

Where	Button	Function
Root window	All Buttons	OPEN LOOK conventions (controlled by window manager)

Table L-1. SunView Button Functions

Where	Button	Function
In Titlebar	Button 1	Front
	Button 2	Move
	Button 3	Window menu
In Application Window	Button 1	Program-defined
	Button 2	Program-defined
	Button 3	Usually client menu

L.3.2 The Root Menu

When the pointer is on the OpenWindows background or root window, its behavior is entirely controlled by the window manager. Running the recommended *olwm* or *olvwm* will cause the normal OPEN LOOK conventions to apply to all keyboard and pointer button activity in the root window.

L.3.3 The Window Menu

Under X11, the "window manager" program provides the "window menu", the menu seen when clicking the MENU button on the titlebar of a window. Under SunView, the user program provides the window menu, so you may see variations from one program to another. The typical program provides the menu shown in Figure L-3, when not iconified.

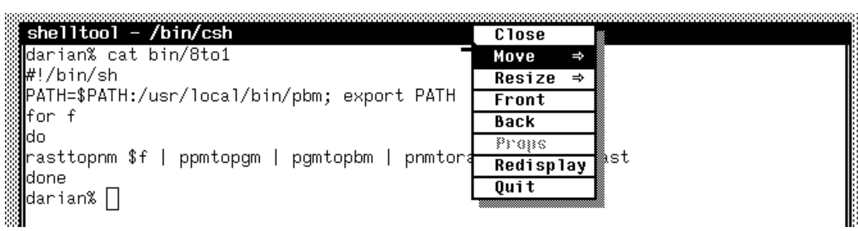

Figure L-3. Typical SunView **Window Menu.**

Most of these entries have the same meaning as they do under OPEN LOOK.

One difference is in what happens when you click on an icon - click-to-select (single-click) actually opens the icon, rather than just selecting it as most X window managers do.

L

L.3.4 Keyboard Shortcuts

Most of the common keyboard shortcuts are the same, including the common **L5, L7, L6, L8, L10, L4,** and **HELP.** See Appendix K, *OPEN LOOK Mouseless Operations* for list of the standard Sun keyboard shortcuts.

L.4 SunView Customization Files

As with OpenWindows, there are numerous files used to control the way programs operate. For each of these files, there is both a system-wide set of files (in */usr/lib*) and a per-user set in a user's home directory. A SunView user might have as many as five files to customize her environment, as shown in Table L-2. Some of these are used when SunView programs run under OpenWindows, and some are not.

Table L-2. SunView Files

Name	Function	Used under OpenWindows?	OpenWindows Analog
~/.defaults	Set defaults for various programs	Yes	*.Xdefaults*
~/.rootmenu	Workspace Menu	No	*~/.openwin-menu*
~/.suntools	Programs to run	No	*~/.xinitrc* or *~/.openwin-init*
~/.text_extras_menu	Menu for *Extras* in text windows	Yes	*~/.text_extras_menu*
~/.textswrc	Customize all "textsw" windows	Yes	*~/.textswrc*

L.4.1 The Defaults files vs. X11 Defaults

The SunView system used "defaults" files to let the user customize the behavior of individual window programs. Their syntax is simpler than that of X Resources—there is no "Widget Hierarchy", for example—but quite different Here is a section of one such file:

```
//Set/askbcc "No"
 $Enumeration ""
 $Help "Enables/disables prompting user for 'bcc' field when sending."
 No "No"
 No/$Help "Do NOT automatically prompt user for 'bcc' field when
           sending."
 Yes "Yes"
 Yes/$Help "Automatically prompt user for 'bcc' field when sending."

//Set/bell "0"
 $Help "Number of times to ring the audible bell when new mail arrives."
```

Part of the structure here is to dynamically control the *defaultsedit* program. This program, shown in Figure L-4,: was used to customize the behavior of SunView programs for an

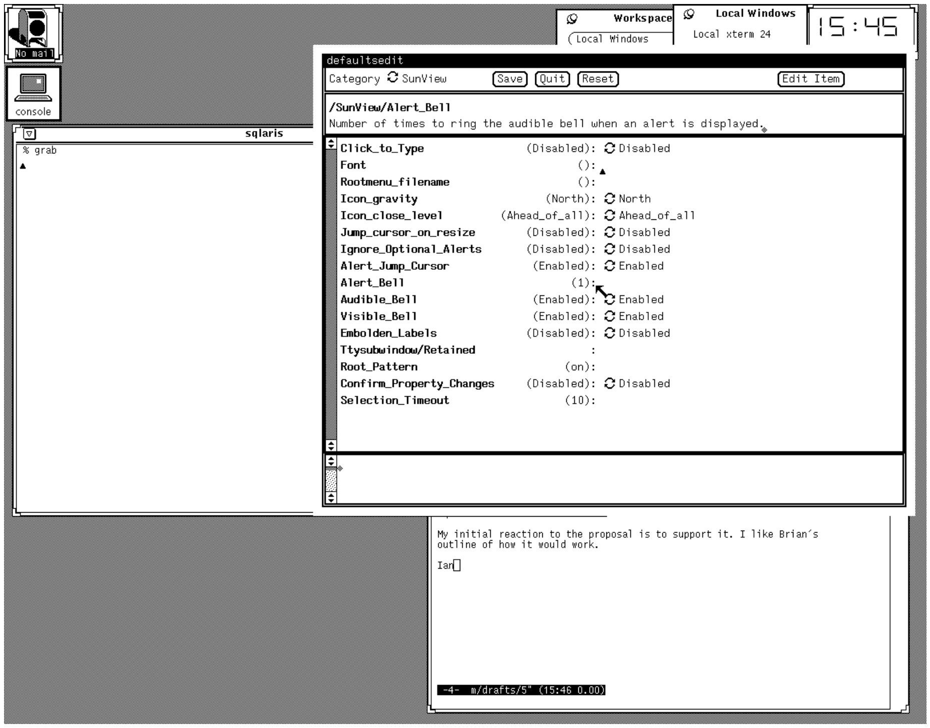

Figure L-4. Defaultsedit (SunView) in action

individual user, like the *properties* program described in Chapter 13, *Customizing olwm*. It only saved the changed values, so a user's *~/.defaults* file might look like this

```
SunDefaults_Version 2
/Mail/Set/editmessagewindow     "Yes"
/Mail/Set/hold  "Yes"
```

For users who may have existing Defaults files, there are two programs. The first, *input_from_defaults,* is normally run automatically, to read your Defaults files and apply them to the running copy of OpenWindows (see the next section for an example). The other, *convert_to_Xdefaults,* is invoked manually, and only once, when you are moving from SunView to X11 or OpenWindows. Its function is to convert a Defaults file to the corresponding X Resources format. This program is normally run with its standard output redirected into a file that will become (or be appended to) your *.Xdefaults* file. See the Reference Page in Part Three if you need to use this program.

L.5 SunView Controls

The use of buttons, menus, and other controls in SunView is very similar to that of OPEN LOOK. This is not surprising, since historically SunView is one of the most direct ancestors of the OPEN LOOK GUI. However there are a few differences that should be noted:

1. The SunView scrollbar is much more like the Athena (*xterm*) scrollbar (see Appendix A, *The xterm/olterm Terminal Emulator*) than like the OPEN LOOK scrollbar described in Chapter 2, *Working in the OPEN LOOK Environment*.
2. Instead of "abbreviated menu buttons", SunView programs uses "cyclical buttons". For example, in the *defaultsedit* window in Figure L-4, clicking the left button on the little circle beside the word *Category* will change the main category from SunView Controls through several other major categories.

By and large, however, a user of the OPEN LOOK GUI will have little trouble getting used to SunView programs.

L.6 Command Line Arguments

The SunView command line arguments are a subset of those discussed in Chapter 11, *Command Line Options*, in particular, in Table 11-2. For example, here is the list of options from the SunView version of *cmdtool*:

Table L-3. SunView Command Arguments

Flag	Long Flag	Argument Type	Notes
-Ww	-width	text columns	
-Wh	-height	text lines	
-Ws	-size	x y (pixels)	
-Wp	-position	x y	
-WP	-icon_position	x y	
-Wl	-label	(quoted) string	
-Wi	-iconic		
-Wn	-no_name_stripe		
-Wt	-font	filename	
-Wf	-foreground_color	red green blue	0-255 (no color-full color)
-Wb	-background_color	red green blue	0-255 (no color-full color)
-Wg	-set_default_color	same	(apply color to subwindows too)
-WI	-icon_image	filename	
-WL	-icon_label	(quoted) string	
-WT	-icon_font	filename	

Table L-3. SunView Command Arguments

Flag	Long Flag	Argument Type	Notes
-WH	-help		

As you can see, these options are almost all the same as the like-named options discussed in Chapter 12, but many of the ones discussed there are not accepted in SunView applications. In particular there is no *-display* option: SunView windows will only appear on the screen of the computer they are running on. And there is no "color name database" (see Section 11.5, "Specifying Color" on Page 289) so you have to specify colors by their numeric RGB values.

L.7 Setup

In order to use the SunView compatibility mode, there are some programs that you must run. If you use the provided OpenWindows-specific startup scripts, this is done automatically. If not, you should copy a few lines from the various system startup files. The provided *openwin-sys* file contains these lines:

```
# Load SunView defaults and invoke SunView/XView selection service
# if NOSunView != 1
if [ -z "$NOSunView" -o "$NOSunView" -ne 1 ];
 then input_from_defaults
 sv_xv_sel_svc &
fi
```

And the provided *.xinitrc* file (*/usr/openwin/lib/Xinitrc*) includes this:

```
# SunView binary compatibility is default mode.
 [ -z "$NOSunView" -o "$NOSunView" -ne 1 ] && eval `svenv -env`
```

These both test the shell variable NOSunView and, if it is not present, fire up various programs that are part of the SunView compatibility mode. If you maintain your own *.xinitrcc* file, you should add these entries to it. Leaving out the testing, all you really need is:

```
input_from_defaults# get sunvew defaults
sv_xv_sel_svc &# enable cut-n-paste btwn SunView and X11
eval `svenv -env`# setup "environment variables" for SunView compat.
```

Alternately, you can run obtain the first two commands by running the provided "system" file from within your *.xinitrc* file, just by including this line in your *.xinitrc* file.

```
 $OPENWINHOME/lib/openwin-sys # OpenWin "system" initialization
```

L.8 "OPEN LOOK/SunView"

To further confuse you, a group in Sun has created an interim toolkit called "OPEN LOOK SunView" (or vice versa). This was to allow developers to build SunView applications that conform to the OPEN LOOK specification. These run as SunView applications under OpenWindows, but use more-or-less the OPEN LOOK conventions, not those shown in the tables above. As far as can be determined, this toolkit has not been distributed outside Sun. However there are two products that are known to use it, and there may be others:

1. The "DeskSet Environment for SunView", part number DSK-1.0-4-34R-5, which gives SunView users OPEN LOOK versions of File Manager, cm, Mailtool, calctool, textedit, printtool, tapetool, snapshot, iconedit, perfmeter, binder, cmdtool, shelltool, console and Sun's clock.
2. Sun's SunNet License Platform's *nladmin* (network license administration) tool.

Unless you use either of these packages, you are unlikely to run into the "SunView OPEN LOOK".

L.9 Future Directions

The SunView compatibility mode provides a useful transition tool to allow users of OpenWindows to run any of the hundreds of SunView applications that they may have available.This compatibility works in OpenWindows Release 3.1 under SunOS 5.1, but will end in an unspecified future release of SunOS[1]; applications that are created under SunOS4.x *must* be dynamically linked if they are to run on SunOS 5.x/Solaris 2. You should convert your applications to X11 as soon as possible.

If you have the source code to an application available, there are (at least) two approaches. One method of converting SunView applications to X11 is to use Sun's XView toolkit; XView is intentionally similar to the SunView toolkit, to facilitate just such conversion. Simple SunView applications can usually be converted using a program provided with OpenWindows (called *convert_to_xview*, and described in the manual page in Part Three of this guide). Sophisticated applications will require some programmer time to convert. XView is described more fully in Volume Seven, *XView Programming Manual*.

Another approach is to use Sun's *devGuide* (Developer's Guide) to re-write the user interface code; GUIDE can generate the user-interface code for XView, for OLIT (an X Intrinsics-based Toolkit that is conceptually closer to Motif), or for The NeWS Toolkit. Either of these approaches – *convert_to_xview* or *devGuide* – will allow your programming staff to update older applications to work with X11 and/or OpenWindows, so that programs you have been using under SunView can continue to be used on The X Window System.

1. One can imagine that some future release of OpenWindows will replace the SunView application emulation with a mode that runs MicroSoft Windows applications in the same kind of "overlay plane" without having to start up SunPC and then MS-Windows.