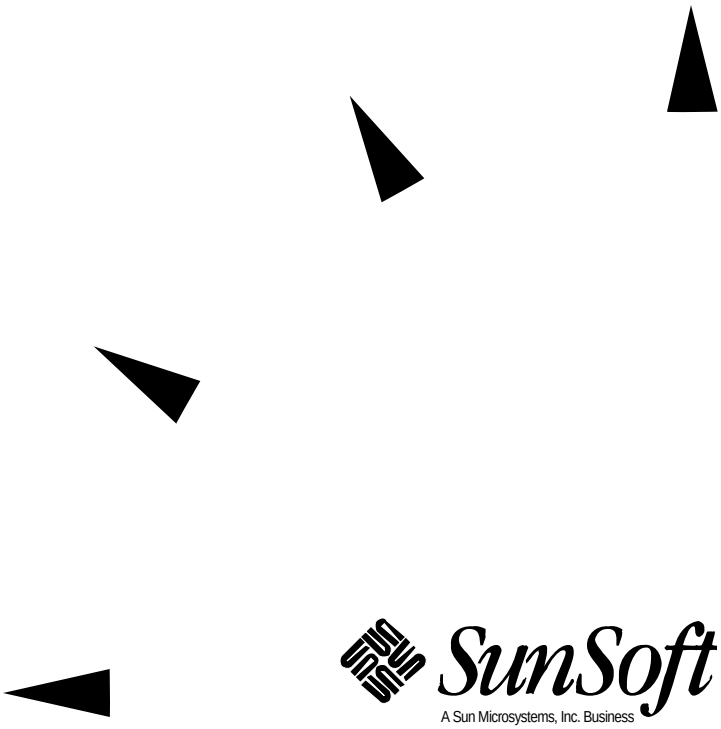
OpenWindows Developer's Guide: User's Guide



2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

SunSoft
A Sun Microsystems, Inc. Business

Contents

Figures

Code Samples

Preface

OpenWindows™ Developer's Guide 3.0.1 *(*referred to throughout this manual as Devguide*)* is a development tool designed to make the interface programmer's job easier. Using it gives you the freedom to create and try user interfaces without writing any code. Because you can create and modify an interface easily, you'll find that you can spend more time designing and testing interfaces, the surest route to better user interfaces for software users. This manual introduces Devguide and shows you how best to use it.

Once you're finished with the interface design, use one of two code generators to generate interface source code and then link the interface to the application. Refer to the programmer's guide appropriate for the code generator you plan to use for details on how to generate code for your application.

Who Should Use This Book

Use this manual if you want to build a user interface with Devguide. Devguide is a powerful programmer's tool. It is also a useful tool for non-programmers, such as user interface designers and project managers, because you can easily create and modify user interfaces without generating any source code.

Devguide can generate code for two toolkits; refer to one of these programmer's guides for details:

- *OpenWindows Developer's Guide: XView Code Generator Programmer's Guide*
- *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide*

Motif Conversion Utilities

Also included with Devguide in the Software Developer Kit are two Motif conversion utilities, `gmf` and `guil`; see the *OpenWindows Developer's Guide: Motif Conversion Utilities Guide* for information about these tools. Use `gmf` to convert Devguide GIL files to Motif C code or `guil` to convert GIL files to UIL format.

SPARC and x86 Differences

Depending on what kind of system you have (SPARC or x86), you may have a different kind of mouse and a different type of keyboard, and devices may be named differently; the differences are described below.

Two- or Three-Button Mouse

Your system may have a two- or three-button mouse. On either type of mouse, the left mouse button is referred to as SELECT and the right mouse button as MENU in this manual. To perform a SELECT function, click the left button; to perform a MENU function, click or press the right button.

On a three-button mouse, the middle button is referred to as ADJUST. To perform an ADJUST function on a two-button mouse, hold down the Shift key and click SELECT (the left button).

Keyboard Alternatives for the Meta and Other Special Keys

Some keyboards do not include the Meta key, the function keys L1 - L10, or a Help key; keyboard alternatives are available for the Meta and Help keys and most of the function keys.

Alternative to the Meta Key

On keyboards without the Meta key (marked as a diamond on most SPARC keyboards), the Alt key is mapped to the Meta key. If you have installed Devguide on an x86 system, use the key labeled "Alt" whenever the Meta key is prescribed.

Note – Functions which require the Alt key itself will *not* work on systems with keyboards that do not include a Meta key.

Alternatives to Function Keys

On keyboards without the L1 to L10 function keys, Meta-key equivalents are available for most of the functions, as shown below. Use the key labeled “Alt” instead of the Meta key on keyboards without the Meta key. The alternatives are:

Cut	Meta-x
Copy	Meta-c
Paste	Meta-v
Find	Meta-f
Props	Meta-i
Undo	Meta-z
Open	Meta-w

There are no keyboard alternatives for the Front and Again keys.

Alternative to the Help Key

On keyboards without a “Help” key, use the key marked “F1” for help.

Device Name Differences

The device name for the CD-ROM on a SPARC system is usually `/dev/dsk/c0t6d0s0`; on an x86 system it is usually `/dev/dsk/c0t6d0p0`.

Your CD-ROM device name may be different, depending on your machine and how you configure it. We use `/dev/dsk/c0t6d0s0` in the examples in this manual; substitute your device name in the command line instructions where appropriate.

Before You Read This Book

Before you can run Devguide as described in this manual, you must first install it on your workstation. To do so, follow the instructions in the *Software Developer Kit Installation Guide.*

For any last-minute additions or corrections to this manual, see Appendix D, "Devguide 3.0.1 Release Notes."

How This Book Is Organized

This manual includes four chapters and four appendixes.

Chapter 1, "Introduction to Devguide," discusses Devguide, its connection to the code generators, what you need to run Devguide, and some naming conventions.

Chapter 2, "Getting Started," shows you how to run Devguide and its on-line help, and presents a quick example of how to create an interface using Devguide.

Chapter 3, "Using Devguide," describes Devguide's base window, working with user interface (UI) elements, and the many Devguide features available for application building.

Chapter 4, "Devguide Reference," describes the UI elements' property windows and property fields.

Appendix A, "Files Provided With Devguide describes the files that get installed on your workstation or server when you install Devguide.

Appendix B, "Porting Devguide 3.0 Applications to Devguide 3.0.1 describes how to port your Devguide applications from a Solaris™ 1.0 environment to a Solaris 2.x environment.

Appendix C, "Adding Devguide to the SPARCworks or ProWorks Environment shows you how to integrate Devguide into SPARCworks or ProWorks.

Appendix D, "Devguide 3.0.1 Release Notes," describes last-minute changes to Devguide that did not get incorporated into this and other Devguide manuals.

This manual concludes with an index to help you easily find information about Devguide.

Related Books

This manual isn't for novice Sun users: it assumes that you've worked with Sun workstations andSolaris 2.x. Before you read this manual, you should be familiar with the UNIX® and SunOS™ operating systems, OpenWindows Version 3.0.1, the OPEN LOOK® user interface and, if you're going to link Devguide interfaces to application software, the toolkit or programming language you wish to use. It's particularly important that you know how to work in the OpenWindows environment, using the pointer and the SELECT, ADJUST, or MENU mouse buttons (or their equivalents) to work with windows, panes, menus, and controls.

For more information about these topics, you can consult:

- *Solaris 2.4 User AnswerBook* for information about the UNIX and SunOS operating systems
- *OpenWindows 3.1 User's Guide* for information on working in the OpenWindows environment
- *The OPEN LOOK Graphical UI Style Guidelines* for information about the elements that constitute user interfaces in the OpenWindows environment and the rules and suggestions for creating OPEN LOOK user interfaces

What Typographic Changes and Symbols Mean

The following table describes the type changes and symbols used in this book.

Table P-1 Typographic Conventions

Typeface or Symbol	Meaning	Examples
`Courier`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. % You have mail.
`Courier Bold`	What you type, contrasted with on-screen computer output	% **su** password:
Palatino Italic	Command-line placeholder: replace with a real name or value	To delete a file, type the following: `rm` *filename*`.`

Table P-1 Typographic Conventions

Typeface or Symbol	Meaning	Examples
	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.
Code samples are included in boxes and may display the following:		
%	UNIX C shell prompt	% *or* system%
$	UNIX Bourne shell prompt	$ *or* system$
#	Superuser prompt, either shell	# *or* system#

Introduction to Devguide 1

You can use Devguide to create user interfaces that conform to the OPEN LOOK user interface. Assemble the elements of a user interface by dragging visual representations of the elements onto the workspace of the monitor screen and putting the elements together. You can assemble windows, control areas, panes, buttons, menus, sliders, and other OPEN LOOK UI elements in your interface and then test their operation together while still using Devguide.

Available Companion Tools

Devguide works with two toolkits: the XView toolkit, and the OPEN LOOK Intrinsics Toolkit (OLIT). The Devguide software you receive includes toolkit code generators that you can use with Devguide. The code generators are:

- GXV – Use GXV when you want to generate an XView application.
- GOLIT – Use GOLIT when you want to generate an OLIT application.

How the Tools and Devguide Fit Together

When you finish assembling the elements of a user interface, save the interface into one or more Guide Interface Language (GIL) files. You can recall the GIL files later for modification or to generate source code. Use one of Devguide's companion programs (GXV or GOLIT) to generate window system code from GIL files. Then add your code to complete your application.

1

Naming Conventions

As you create elements in Devguide, each interface element stored in a single GIL file must have a unique name. It's very important that you don't assign the same name to more than one element in the interface—if you do, you'll have problems with name clashing (multiply defined symbols) during the compile. You can, however, use the same element name in separate GIL files as long as the name occurs only once in each GIL file. Because windows tend to include parallel elements, it's often easy to give those elements the same name. If you store each window in a separate GIL file, you can use the same name within each window without worrying about compiler errors.

The connections you specify for interface elements are global to an application, and must each have a unique name. A connection should be called by the same name every time you use it, whether it occurs in the same GIL file or across two or more GIL files. See "Handling Events Using Connections" on page 81 for further details on connections.

Getting Started 2

This chapter introduces Devguide by giving you a hands-on tour. You see how to start Devguide, how to use Devguide's on-line help, and how to put together a simple user interface using Devguide. When you're finished, you should know your way around Devguide well enough to start building your own simple interfaces or to move on to the following chapters to learn more about Devguide's features.

Installing Devguide

Before you run Devguide, make sure Devguide and a companion program (GXV or GOLIT) are installed on your workstation or server. If they aren't installed, you or your system administrator should follow the instructions in the *Software Developer Kit Installation Guide.* The installation procedure uses the CD that comes with your manual set to install Devguide, the two code generators, and related files.

Changing Environment Variables to Run Devguide

Once you've installed Devguide, you must set the appropriate environment variables—`GUIDEHOME`, `PATH`, and `HELPPATH`—before you can run Devguide and the code generators.

- `GUIDEHOME` – Sets Devguide's home directory
- `PATH` – Tells your shell where to find Devguide, GXV, and GOLIT for execution

- HELPPATH – Tells your window system toolkit where to find Devguide- and application-generated help files

If you use the C shell, you'll add GUIDEHOME and HELPPATH to the file named .login in your home directory, and will most likely find PATH in the .login or .cshrc files. If you use the Korn or Bourne shell, you'll work with GUIDEHOME, PATH and HELPPATH in the .profile file.

C Shell Users

If you're a C shell user, use emacs, vi, or another text editor to add a GUIDEHOME line to the contents of the .login file in your home directory. The line starts
setenv GUIDEHOME and follows with a space and the pathname to Devguide's home directory, which is the destination directory you used during installation. For example, if you used the destination directory /usr/devguide during installation, the line you add will read:

```
setenv GUIDEHOME /usr/devguide
```

Now use your text editor to add a HELPPATH line to the contents of the .login file.The line starts setenv HELPPATH and follows with a space and a series of two directories that are searched in order to find help files:

```
setenv HELPPATH ${GUIDEHOME}/lib/locale:${GUIDEHOME}/lib/help
```

Once Devguide's home directory has been set, use your text editor to look through the contents of .login and .cshrc for a line that starts with set path = (and follows with a collection of pathnames, each separated by a space, and ended with a closing parenthesis. This is a typical set path line:

```
set path = (. ~/bin /usr/ucb /bin /usr/bin /usr/local/bin)
```

Once you find the path line, use your text editor to add $GUIDEHOME/bin to the end of the line, just before the closing parenthesis. Put a space before the pathname to separate it from the other pathnames. Add a backslash if you

need more than one line for your path. For example, if you added the Devguide pathname to the sample `set path` line presented above, the line would now read:

```
set path = (. ~/bin /usr/ucb /bin /usr/bin\
           /usr/local/bin $GUIDEHOME/bin)
```

When you're finished editing and saving these files, you must exit the window environment and log in again to put these environment variables into effect.

Korn and Bourne Shell Users

If you're a Korn or Bourne shell user, use `emacs, vi`, or another text editor to add two `GUIDEHOME` lines to the contents of the `.profile` file in your home directory. The first line starts `GUIDEHOME=` and follows with the pathname to Devguide's home directory, which is the destination directory you used during installation. For example, if you used the destination directory `/usr/devguide`, the lines you add will read:

```
GUIDEHOME=/usr/devguide
export GUIDEHOME
```

Note – If `PATH` or `HELPPATH` is not already in your environment, you must export it, as with `GUIDEHOME` above.

Now use your text editor to add a `HELPPATH` line to the contents of the `.profile` file. The line starts `HELPPATH=` and follows with a series of two directories that are searched in order to find help files:

```
HELPPATH=${GUIDEHOME}/lib/locale:${GUIDEHOME}/lib/help
```

Once Devguide's home directory has been set, look through the rest of the `.profile` file. You're looking for a line in the file that starts `PATH=` and follows with a collection of pathnames separated by colons. This is a typical `PATH` line:

```
PATH=.:$HOME/bin:/usr/ucb:/bin:/usr/bin:/usr/local/bin
```

Once you find the path line, use your text editor to add `:$GUIDEHOME/bin` to the end of the line. For example, if you added the Devguide pathname to the sample `PATH` line presented above, the line would now read:

```
PATH=.:$HOME/bin:/usr/ucb:/bin:/usr/bin:\
       /usr/local/bin:$GUIDEHOME/bin
```

When you're finished editing and saving these files, you must exit the window environment and log in again to put these environment variables into effect.

Starting Devguide

Before starting Devguide, you must be running the window system environment. You can use the File Manager to start Devguide by double-clicking on the Devguide glyph, but it is probably easiest to start Devguide from a shell tool or shell script.

Execute the `devguide` command in a shell tool whenever you want to start Devguide:

```
% devguide &
```

Choosing a Toolkit

After you execute the `devguide` command, Devguide's base window appears. You should make a toolkit choice now if you plan to generate code after you build a user interface. Do this by choosing Devguide... from Devguide's Properties menu. The Devguide Properties window appears.

Set your toolkit preference using the Toolkit menu. XView is the default choice. Certain properties of Devguide are targeted to the OPEN LOOK toolkit you choose. It is important to choose the right toolkit for your application before you begin work. You need to set a toolkit choice only once. Thereafter, Devguide uses your selection as the default toolkit whenever you start Devguide.

Note – If you don't choose a toolkit, Devguide assumes you want to build an XView application, and sets the environment accordingly. See "Customizing Devguide's Properties" on page 52 for a discussion of the Devguide property window.

Quitting Devguide

To quit Devguide, choose Quit from the Devguide Window menu. (Read more about the Window menu in "Window Menu" on page 30.) If you have unsaved work, Devguide presents a notice, shown in Figure 2-1, telling you that you have unsaved edits. Choose Cancel to continue running Devguide, or choose Discard Changes to quit Devguide, discarding your unsaved work. When you quit, the Devguide window disappears along with any UI elements you have laid out on the workspace using Devguide.

Figure 2-1 Unsaved-Edits Notice

2

Devguide Window

When you start Devguide, the window shown in Figure 2-2 appears on the screen.

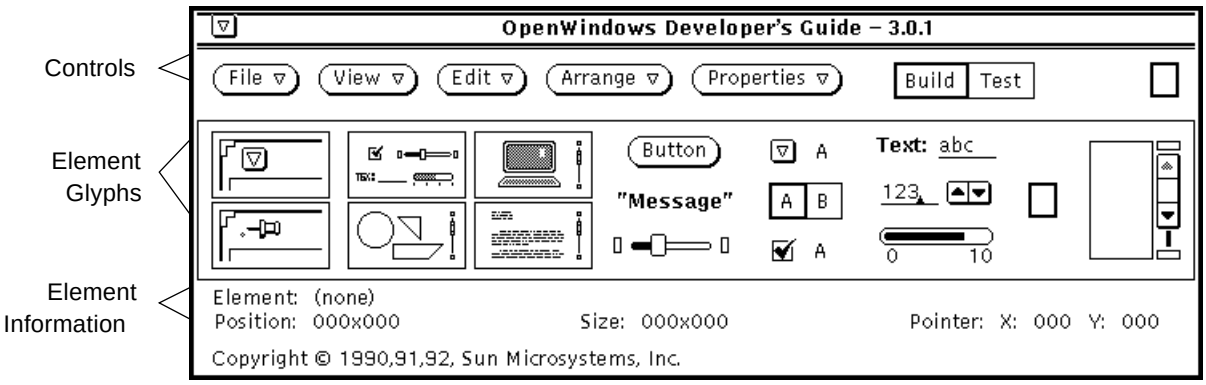


Figure 2-2 Devguide Window

The Devguide window contains the controls and information you need to operate Devguide and the element glyphs you use to add elements to a user interface. Devguide's controls are located in the top of the window, the element glyphs are in the middle of the window, and information about the elements is in the bottom of the window. The middle portion of the window is called Devguide's *palette*. Click SELECT on the abbreviated button in the upper left corner of Devguide's window to iconize the window.

Note – It's important to make a distinction between two types of user interface elements used in Devguide: elements you drag onto the workspace to create an interface, and elements in Devguide's base window (such as menu buttons and menus) that you use to control Devguide. Whenever confusion is possible, this manual uses the term *UI element* (short for User Interface element) for an element you use to build an interface.

Controls

The Devguide controls appear across the top of the Devguide window, as shown in Figure 2-3.

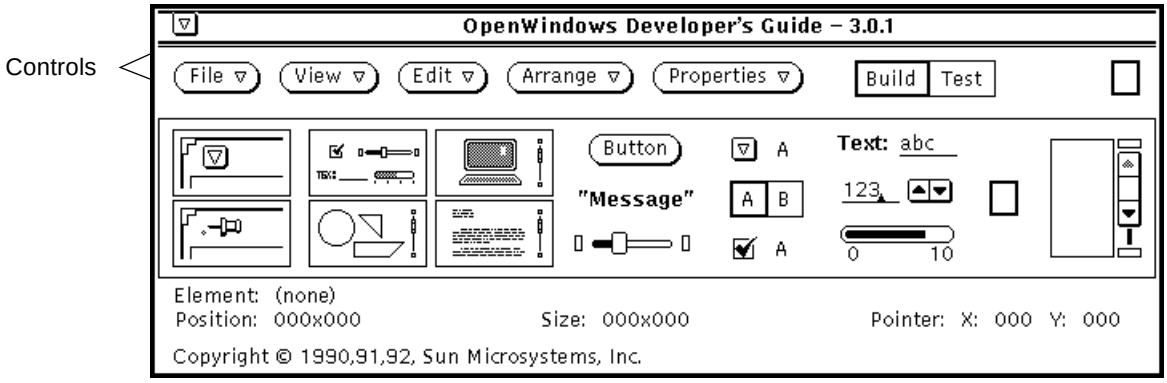


Figure 2-3 Devguide Controls

The Devguide controls include:

- The File menu button, which contains menu items used to load, unload, and save interfaces and projects that you create in Devguide.
- The View menu button, which enables you to scroll through layered panes, open previously dismissed pop-up windows, and bring up the object browser.
- The Edit menu button, which enables you to cut, copy, and paste elements within and between different interfaces.
- The Arrange menu button, which enables you to group, ungroup, and align UI elements, and to use grid alignment options.
- The Properties menu button, which enables you to open windows for editing the properties of individual elements within a user interface.
- The Interface Setting with Build and Test options that set Devguide in build or test mode.

- The Drag and Drop Target, which provides a location for users to drop GIL (Guide Interface Language) files for loading into Devguide. (Do not confuse this with the drag and drop target glyph in Devguide's palette that you can use when you build an application.)

Element Glyphs

The element glyphs appear in Devguide's palette, as shown in Figure 2-4. To create an interface with Devguide, you simply drag element glyphs from Devguide's palette onto the workspace, where each glyph turns into an interface element. You can lay elements down on top of one another or next to one another to assemble the interface you want.

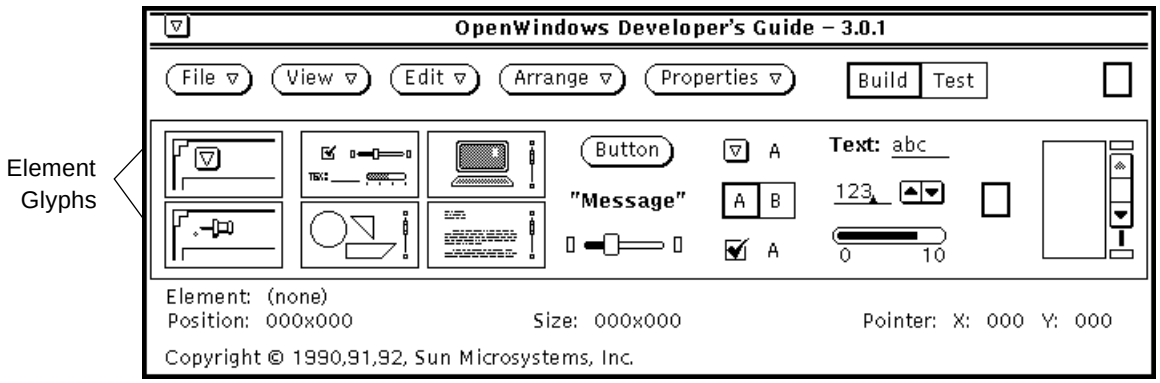
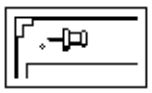


Figure 2-4 Devguide Element Glyphs

These element glyphs include:

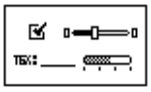
Windows

The Base Window glyph places a base window on the workspace when you drag it there.



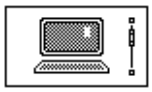
The Pop-up Window glyph places a pop-up window on the workspace when you drag it there.

Panes



The Control Area glyph places a control area in an interface when you drag it onto a base window or pop-up window.

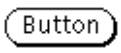
The Canvas Pane glyph places a canvas pane in an interface when you drag it onto a base window or pop-up window.



The Term Pane glyph places a term pane in an interface when you drag it onto a base window or pop-up window.

The Text Pane glyph places a text pane in an interface when you drag it onto a base window or pop-up window.

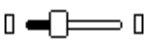
Controls



The Button glyph places a button in an interface when you drag it onto a control area. You can place an *Abbreviated Menu Button* in an interface by first dragging the Button glyph onto the workspace, bringing up the Buttons property window, and altering the button's properties. A button becomes a *menu button* when you attach a menu to it.

"Message"

The Message glyph places a message in an interface when you drag it onto a control area.

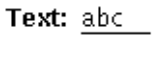


The Slider glyph places a slider in an interface when you drag it onto a control area.

The Stack Setting glyph places a stack setting in an interface when you drag it onto a control area.

The Exclusive Setting glyph places an exclusive setting in an interface when you drag it onto a control area.

The Check Box Setting glyph places a check box setting in an interface when you drag it onto a control area. You can place a *Nonexclusive Setting* in an interface by first dragging any other Setting glyph onto the workspace, bringing up the Settings property window, and altering the setting's properties.



The Alphanumeric Text Field glyph places an alphanumeric text field in an interface when you drag it onto a control area.

The Numeric Text Field glyph places a numeric text field in an interface when you drag it onto a control area. You can place a *Multiline Text Field* in an interface by first dragging any other Text Field glyph onto the workspace, bringing up the Text Fields property window, and altering the text field's properties.

The Gauge glyph places a gauge in an interface when you drag it onto a control area.

The Drop Target glyph places a drag and drop target in an interface when you drag it onto a control area.

The Scrolling List glyph places a scrolling list in an interface when you drag it onto a control area.

Element Information

The Devguide element information appears along the bottom of the Devguide window, as shown in Figure 2-5.

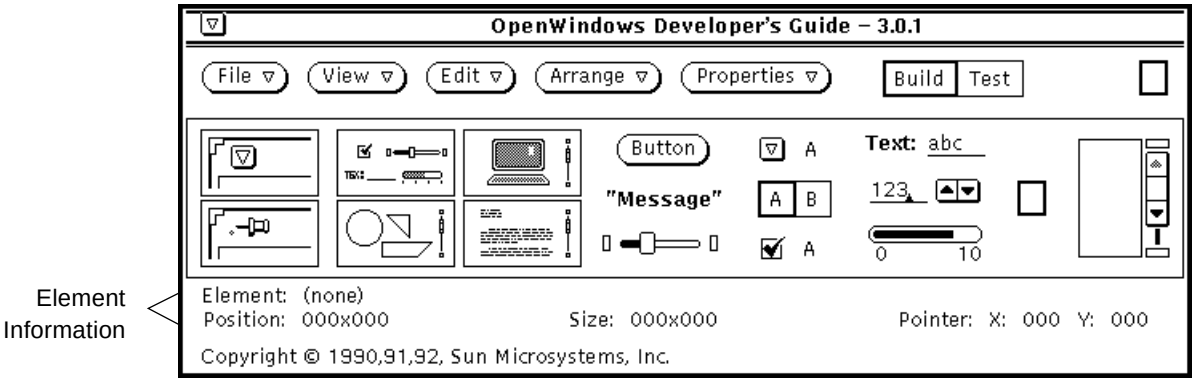


Figure 2-5 Devguide Element Information

The element information area gives you precise information on the position, size, and shape of the current element as well as the precise pointer location.

- The Element field displays the name of the interface element under the pointer.
- The Position field displays the position of the interface element under the pointer. (If the element is in a layer of panes, the layered panes' name and size and relative position of the pane in the layer is also displayed.)
- The Size field displays the size of the interface element under the pointer.
- The Pointer field displays the location of the pointer within an interface window or pane.

On-line Help

If you want information about any of the elements in the Devguide window or in any of the Devguide property windows, you can read the help text about an element simply by moving the pointer over the element in question and pressing the Help key on the keyboard. (The Help key is the F1 key on keyboards that do not have a special Help key.) A Help window like that shown in Figure 2-6 appears.

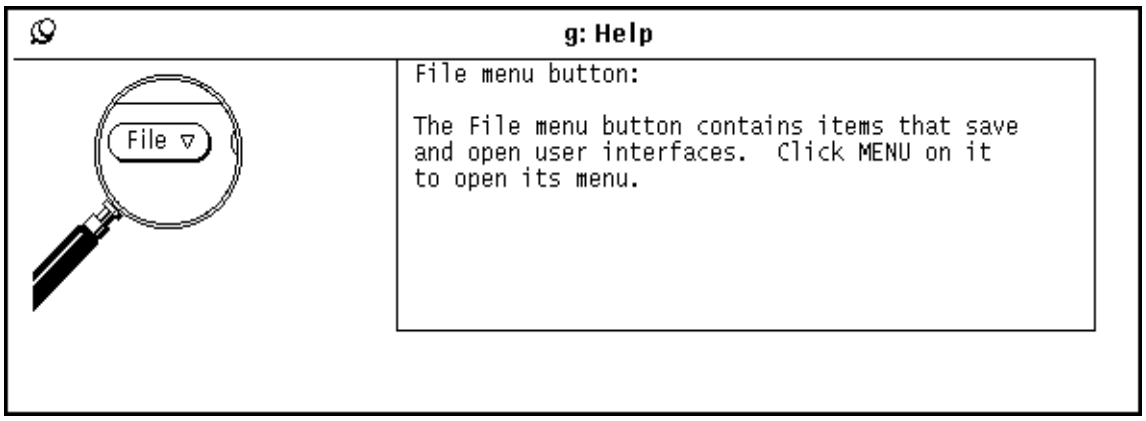


Figure 2-6 Help Window for the File Menu Button

Once you are finished reading the help text about an element in the help window, you can view the help text about another element by moving the pointer over the new element and pressing the Help key once again. The text in the Help window changes to describe the new element. When you are finished reading help text, dismiss the Help window by clicking SELECT on the pushpin in the upper-left corner of the Help window.

Creating Interfaces: A Quick Example

The easiest way to get to know Devguide is to use it to create a very simple user interface. In this quick example, you will:

1. Create an interface by placing a base window in the workspace, adding a control area, and then populating the control area with a button and a setting.
2. Edit the interface by changing the base window's label, changing the button's label, and adding some items to the setting.
3. Test the interface by trying the controls.
4. Save the interface and quit Devguide.

Creating the Interface

First you will place the basic user interface elements in the approriate positions: a base window, a control area, a button, and a setting.

Placing a Base Window

Start your interface by placing a base window on the workspace.

1. **Move the pointer over the Base Window glyph (shown in Figure 2-7) in Devguide's palette.**

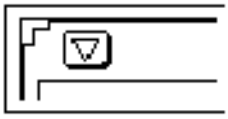


Figure 2-7 Base Window Glyph

2. **Drag the glyph from the palette onto the workspace.**
   To drag, hold down SELECT while you move the pointer, then release SELECT when the pointer is located where you want it. When you drop the glyph, it disappears and a base window appears.

3. **Drag the base window so it appears just to the right of the Devguide window.**
   To drag a window, place the pointer on its header—the strip along the top of the window—then move the pointer while holding down SELECT. Release SELECT when the window is in the location you want.

Adding a Control Area to a Base Window

A base window is the foundation for any user interface. To add controls to a base window, you must first add a control area to the window.

1. **Drag the Control Area glyph from Devguide's palette to the upper left corner of the base window you just created and drop the glyph there.**
   A small control area appears where you dropped the glyph. Because you just placed the control area on the workspace, it is selected and ready for you to resize. Eight small black squares—its resize handles—appear around the control area's borders to show that it's selected. Figure 2-8 shows the control area's resize handles, along with what the control area looks like as it is being stretched to fill the base window. (The oversized control area snaps to fit the window.)

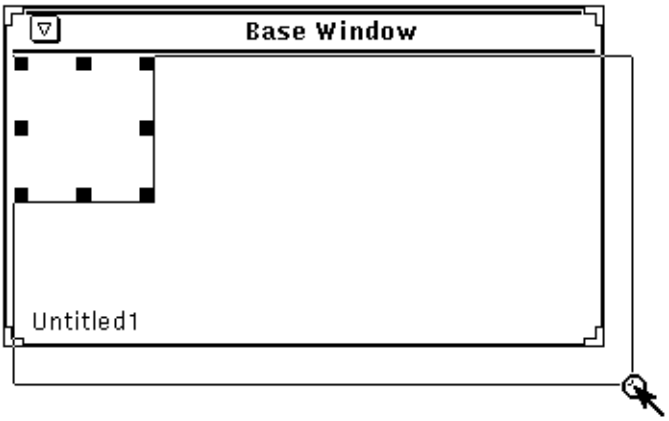


Figure 2-8 Resize Handles; Stretching Out the Control Area

2. **Drag the control area's lower right resize handle down and to the right until you've stretched the control area past the lower right corner of the base window (as shown in Figure 2-8), then release the resize corner.**
   The control area resizes to fill the entire base window with a small strip of space along the bottom of the window—the window's footer.

Adding Controls to the Control Area

Now that you have a base window with a control area, you can add controls to the control area. Add a setting and a button.

1. **Drag the Exclusive Setting glyph from Devguide's palette to the upper half of the control area and drop it there.**
   An exclusive setting appears in the control area.

2. **Drag the Button glyph from Devguide's palette to the bottom half of the control area and drop it there.**
   A button appears in the control area.

Editing the Interface

Once the elements of a user interface are in place, you can edit each individual element to your specifications. To edit an element, you change element attributes in a property window that lists the properties of that element.

To see the property window, you can select the element on the workspace and then use its pop-up menu to open its property window. Or you can use the Properties menu button to open the appropriate property window and then choose the name of the specific element you want to edit. You'll see how to use both methods of editing in the examples that follow. (You can also double-click SELECT on the element to open its property window.)

Labeling the Base Window

The label of the base window you created appears in the window's header; it is labeled "Base Window". To change the label, first open the Base Windows property window and select the base window in the property window's scrolling list.

1. **Move the pointer over the Properties menu button and click MENU.**
   The Properties menu (shown in Figure 2-9) opens.

Figure 2-9 Properties Menu

2. **Click MENU on the Base Windows item in the Windows submenu.**
   The Base Windows property window (shown in Figure 2-10) appears. The Base Windows scrolling list at the top of the property window lists all the base windows in your interface. It should show only “window1” right now. Click SELECT on the name “untitled1::window1” in the Elements list to show the properties for your base window.

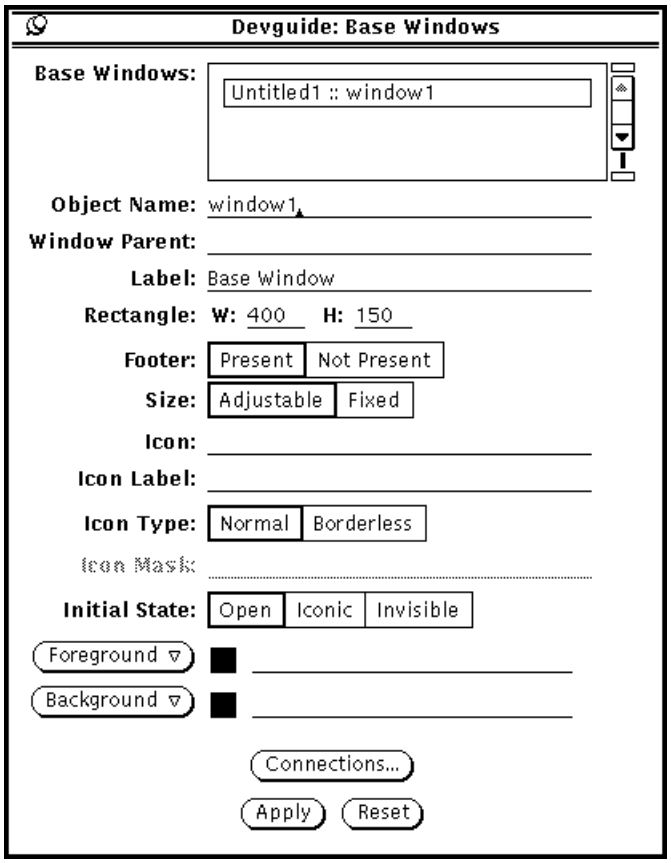


Figure 2-10 Base Windows Property Window

3. **Move the pointer to the end of the characters following the text field Label (the characters should read Base Window) and click SELECT.**
   A small triangular caret appears after the letter “w”.

4. **Press the Delete key until all the characters in the text field are deleted.**

5. **Type the label** `Coffee Machine`**.**

6. **Choose the Apply button at the bottom of the property window to apply the name change you just made.**
   The header of the base window now reads `Coffee Machine`.

7. **Click SELECT on the pushpin in the upper left corner of the property window to dismiss the window.**

Adding Items to the Exclusive Setting

Now add some items to the Exclusive setting you placed in your interface. Select the setting and use its property window:

1. **Move the pointer over the setting and click SELECT. A selection box appears around the setting.**
2. **Click MENU to open the setting's pop-up menu.**
3. **Choose Properties from the menu.**
   The Settings property window (shown in Figure 2-11) appears. The setting's name is selected in the Setting scrolling list in the top of the window, and the rest of the window shows the setting's current properties. The Items scrolling list in the bottom half of the window shows the items currently available in the setting. There are two, each named Item. The top one is already selected for editing, as you can see by the selection box around it.

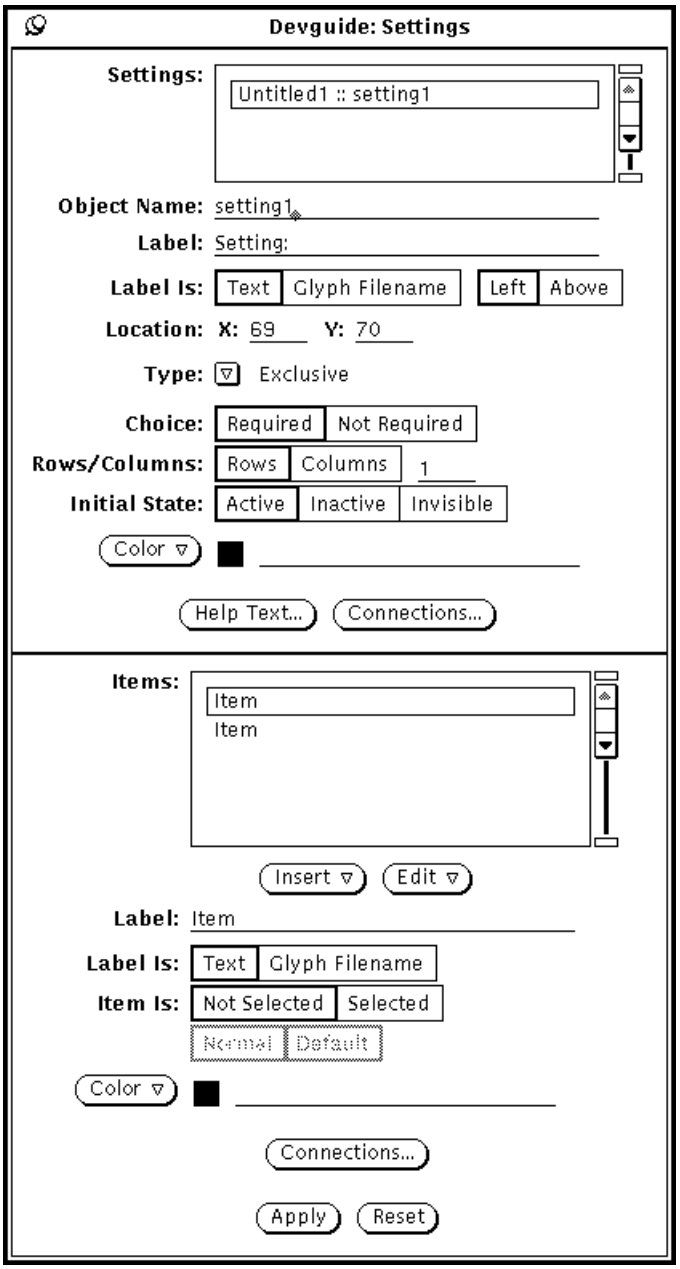


Figure 2-11 Settings Property Window

4. **Click SELECT on the Insert menu button below the Items scrolling list.**
   This chooses the default menu item Bottom in the button menu, which adds a third item—also labeled Item—to the Items scrolling list.

5. **Click SELECT on the top item shown in the Items scrolling list.**
   This selects it for editing, and its label appears in the Label text field below the scrolling list.

6. **Click SELECT just after the "m" in "Item" as it's shown in the Label text field.**
   A caret appears there.

7. **Use the Delete key to delete all the characters in the field.**

8. **Type `W/Cream` in the text field, then click on the second item in the Items scrolling list to select it for editing.**
   The first item's label now reads `W/Cream`. (When you select a new item in the Items scrolling list, this automatically applies edits you just made.)

9. **The second item's label appears in the Label text field; change it to `W/Sugar`, then click SELECT on the third item to select it for editing.**
   The second item's label now reads `W/Sugar`.

10. **The third item's label appears in the Label text field; change it to `Dispense Cup After Coffee`, then click SELECT on any of the other items in the Items scrolling list.**
    The third item's label now reads `Dispense Cup After Coffee`.

11. **Look for the abbreviated menu button in the Settings property window labeled Type; it should be followed by the word Exclusive. Click MENU on the abbreviated menu button (it's labeled with a small, down-pointing wedge).**
    It opens to reveal four different setting types.

12. **Click SELECT on the Nonexclusive item.**
    The menu closes and the Type line now reads Nonexclusive. (This makes the item nonexclusive so the user can choose more than one of the items at a time.)

13. **Click SELECT at the end of the Label text field at the top of the window, then press Delete repeatedly to remove all text from the text field.**
    This removes the setting label so no setting label appears in the interface.

14. **Choose the Apply button at the bottom of the property window to apply the property changes you just made.**
    The setting now offers three different items. If the setting is now off-center in the window, drag it to a new centered location.

15. **Click SELECT on the pushpin in the upper left corner of the property window to dismiss the window.**

Changing the Button Label

Change the label of the button in the bottom half of the window by using its property window.

1. **Move the pointer over the button and double-click SELECT.**
   The button's property window (shown in Figure 2-12) appears, showing the button's current properties.

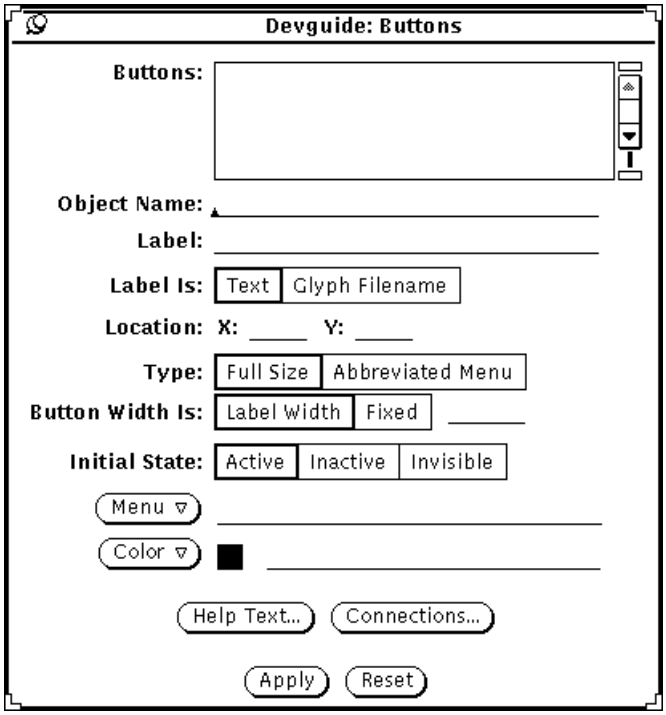


Figure 2-12 Buttons Property Window

2. **Change the contents of the Label text field from Button to `Dispense`. (Change it the same way you changed the base window's label in its property window.)**
3. **Choose the Apply button at the bottom of the property window to apply the name change you just made.**
4. **Click SELECT on the pushpin in the upper left corner of the property window to dismiss the window.**

You've now created the interface shown in Figure 2-13. If your interface looks different, you can resize your base window by dragging its resize corners, and reposition the controls by dragging them to new locations. If you've made a typing error or some other mistake, you can edit properties using the element property windows.

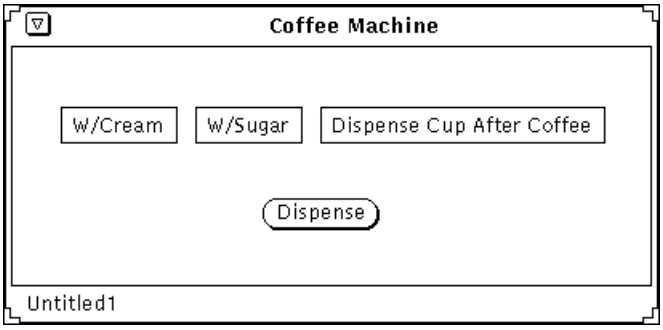


Figure 2-13 Coffee Machine Interface

Testing the Interface

After creating an interface, you may want to see how its elements behave when you use them. One way to do that is to generate the source code for the interface, compile and link the code, and then run the object code file.

A quicker way is to turn on the test mode. In test mode, the elements of your user interface work, for the most part, as they will when a user works with them; that is, the buttons flash, menus appear, and selections set choices. Nothing is connected to the user interface in this example, so the elements don't initiate any actions—coffee won't pour, no cream or sugar is added—but

you can get a feel for the way the interface works in use. Read “Building and Testing” on page 49 and “Handling Events Using Connections” on page 81 to learn about actions that can work in test mode.

1. **Click SELECT on the Test option of the Build/Test setting in the upper right corner of the Devguide window.**
   Devguide enters test mode. Try testing the coffee machine interface now.

2. **Click SELECT on the W/Cream item.**
   If you have a monochrome monitor, a black border appears around the item to show it was chosen. If you have a color monitor, it appears “depressed.”

3. **Click SELECT on the Dispense button.**
   It flashes to show it was chosen.

You can’t edit your interface in test mode. To edit, you must return to build mode, Devguide’s default mode.

4. **Click SELECT on the Build option in the Interface setting. Devguide enters build mode.**

Saving the Interface

When you’re finished working on an interface and want to save it for later editing or source code generation, use the Save As... command in the File button menu.

1. **Click MENU on File at the top of Devguide’s base window.**
   The File button menu appears.

2. **Choose Save As... from the menu.**
   The menu disappears and the Save As file chooser window (shown in Figure 2-14) appears.

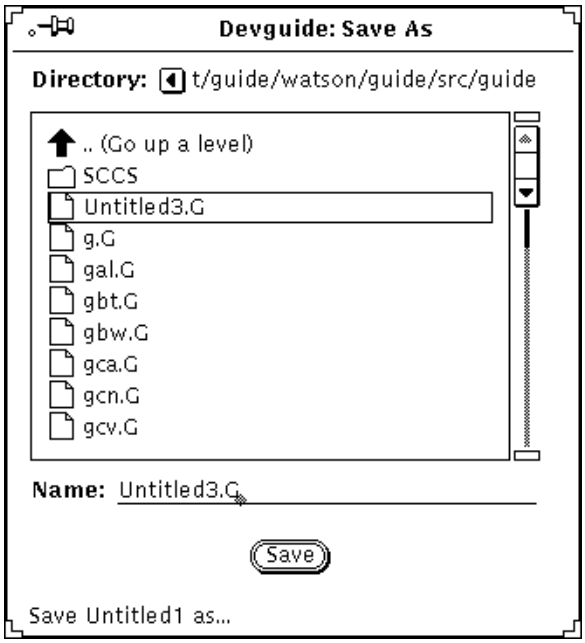


Figure 2-14 Save As File Chooser Window

3. **Move the pointer over the Save As file chooser window, double-click SELECT on the line labeled Name, then type `coffee` to name this interface.**

4. **Choose the Save button to save the interface to Devguide's resident directory using the file name `coffee`.**
   Note that the Directory text field shows the current directory where Devguide will save the interface. The window closes after the save is complete, unless you have pinned it to the workspace.

You have now saved the interface as a GIL file. The interface is stored under the file name `coffee.G`. Devguide appends `.G` to any GIL file it creates, unless you include this extension in the file name. (Devguide appends `.P` to the name of any project file you create, if you have not done so. See "Assembling Interfaces into a Project" on page 87 to learn about creating projects in Devguide.)

Once you're finished working on an interface and have saved it, you can quit Devguide.

Generating Source Code

Now that you've created and saved a user interface, you can use one of Devguide's companion toolkit code generators (GXV, or GOLIT) to generate source code for the user interface. To do this, you can first open a shell tool in the workspace if it's not already open, and then use the appropriate code generator command. As an example, here is how to generate source code for your interface using the `gxv` command:

```
% gxv coffee
```

Alternatively, you can use the Code Generator Tool. See "Using the Code Generator Tool" on page 101 for details.

Compiling Object Code and Running the Results

You can invoke the `make` command yourself, by entering `make` in the shell:

```
% make
```

The executable file for our example is called `coffee`.

Alternatively, you can use the Code Generator Tool to compile and run this application. See "Using the Code Generator Tool" on page 101 for more information.

See the manual in this documentation set for your toolkit code generator for details on generating and compiling source code, and for setting code generator options in the Code Generator Tool.

That's it for this quick and easy introduction to Devguide. Before you leave the scene of your activities, take a few seconds to clean out the files you generated: you should remove all of the `coffee` files, including the generated `Makefile`, from the current directory. If you want to keep these files, create a separate subdirectory in which to store them. If you're ready for more user interface creation, you'll enjoy the next chapter, where you'll learn to use many of Devguide's features.

2

Using Devguide 3

This chapter provides information on using each part of Devguide. It includes:

- Opening, saving, editing, testing, and loading interfaces
- Using Devguide's Properties window to customize your environment
- Using Devguide glyphs to build a user interface
- Using the Help Editor to add help text to interface elements
- Using the Color Chooser to set interface element colors
- Using the Menu Editor to add menus to a user interface
- Using the Groups Property window to create and modify groups
- Using the Connections Manager window to create, modify, and remove connections
- Using the Project Organizer and Interface Browser to create and modify interfaces and projects
- Using Devguide to create Layered Panes
- Using the Code Generator Tool to generate and compile code, and run applications
- Printing windows

3

Basic Devguide Operations

This section explains how to use the Devguide Window menu and the controls in the top third of Devguide's base window.

Window Menu

To display the Window menu (shown in Figure 3-1), click MENU on the Devguide window's abbreviated menu button (in the window's upper left corner), or in its header, footer, or side borders.

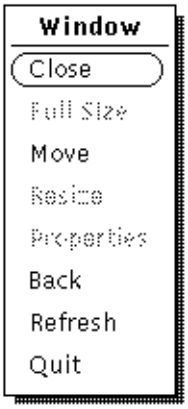


Figure 3-1 Window Menu

Click SELECT on the Devguide window's abbreviated menu button to activate the default item from the Window menu. This usually closes the Devguide window and reduces it to an icon on the workspace because Close is the default Window menu choice. To reopen the window, double-click SELECT on the Devguide icon.

The items in the menu perform these functions:

- Close – Closes the Devguide window and reduces it to an icon.
- Full Size – This feature is disabled because Devguide's base window is a fixed size. Full Size is a generic Open Look function for manipulating windows.

- Move – Moves the Devguide window in mouseless mode. If you choose this, the corners of the window are highlighted. You can then use the arrow keys to move the window pixel by pixel. Hold down Control and the window moves in bigger jumps. Press return when you are done with the move.
- Resize – This feature is disabled because Devguide's base window is a fixed size. Resize is a generic Open Look function for manipulating windows.
- Properties (usually disabled) – If available, this brings up a window that allows you to set properties for the Devguide base window. (Do not confuse this with the property window discussed in "Customizing Devguide's Properties" on page 52!).
- Back – Moves the Devguide window behind any other windows in its area on the screen.
- Refresh – Redraws the contents of the Devguide window, restoring any parts of the window that are not completely restored after an action that obscures part of the window.
- Quit – Unloads all Devguide windows and quits Devguide. If you choose Quit while you have unsaved work, Devguide displays a Notice stating "You have unsaved edits." If you want to quit without saving your work, choose the Discard Changes button. If you want to go back to Devguide to save your work, choose Cancel. This Notice is shown in Figure 2-1 on page 7.

Working with Files

Click MENU on File to display the File menu shown in Figure 3-2. The File Menu contains items that enable you to create, load, save, rename, and unload files and projects.

Figure 3-2 File Menu

Many items on the File menu open up a *file chooser*, or open a window with menu buttons whose items open up a file chooser.

File Chooser

The file chooser is a standard dialog box for file operations. A scrolling list of files and directories in your current directory is displayed when you open the file chooser. Double-clicking SELECT on a directory descends into that directory and displays a list of the files in the directory. Double-clicking on “go up a level” displays the files and subdirectories in the parent directory of your current directory.

An *action button* is located at the bottom of the file chooser dialog box. It is typically labeled Load or Save. Double-clicking on a file name or typing in a file name and pressing the action button loads or saves the file. If the file’s full path name is longer than the line provided, a scrolling button appears and you can continue typing the name on the line.

The file chooser appears when you choose Load... or Save As... from the File menu. It also appears when you choose Save if you haven’t named the file yet.

Load...

Load... loads one or more previously saved user interfaces. When you choose Load..., the file chooser (shown in Figure 3-3) appears. The file chooser window contains a list of all `.P` and `.G` files and all subdirectories in your current directory. Double-clicking on a directory descends into that directory and lists all its `.P` and `.G` files and all its subdirectories.

Devguide appends a `.P` extension to any project file that you save, and `.G` to any interface (GIL) file that you save, if you do not include the extension in the file's name.

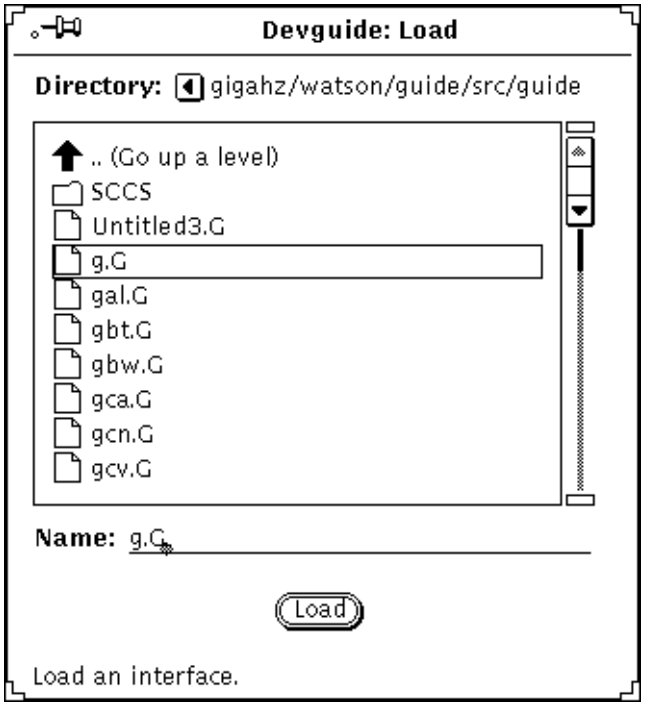


Figure 3-3 Load File Chooser

You can load a `.G` file by double-clicking on the file name if it is visible in the file chooser. If a file name is not visible, you can use the scrollbar to find it or type its full path name into the Name field at the bottom of the file chooser and click Load.

Double-clicking on a `.P` file opens up the *Project Organizer* window. This window contains an icon for each `.G` file contained in the project.These .G files are not initially loaded into Devguide. You can load a `.G` file by double-clicking on its icon. You can also open the Project Organizer for a `.P` file by typing its full path name into the Name field at the bottom of the file chooser,

and clicking Load. To load the project, double-click on each `.G` icon present in the window. The Project Organizer is discussed in “Assembling Interfaces into a Project” on page 87.

When you load a `.G` file, Devguide opens the interface, displaying its elements on the workspace and its name in the lower right corner of the Devguide window. The file chooser then disappears. You can pin the file chooser to the workspace if you want it to remain open.

New Interface...

To create your first interface (`.G` file) after starting Devguide, you can

- Drag element glyphs from the palette, or
- Choose New Interface...

You must choose New Interface... to create a new interface after the first one. To begin building each new interface, drag a base window or pop-up window glyph onto the workspace, or create a menu.

New Interface... opens up the *Interface Browser* window. Each new interface receives a temporary name, such as `Untitled1`. As you build an interface, a tree representation of your interface appears in the Interface Browser. The Interface Browser is discussed in “Assembling Interfaces into a Project” on page 87.

Save

Save saves one or more user interfaces. It is enabled when you have loaded at least one `.G` file. Save has a submenu containing all currently loaded `.G` files as items. The submenu contains a default item labeled “All” when more than one `.G` file appears in the submenu list. See Figure 3-4.

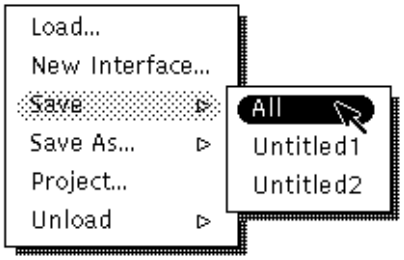


Figure 3-4 Save Submenu

When you choose Save, Devguide saves your interface as a GIL file with the extension `.G`. It turns the previously saved version of the GIL file into a backup file with the extension .BAK. For example, if you save an interface named `face`, Devguide saves it in a file named `face.G`. The file containing the last saved version of the interface is renamed from `face.G` to `face.G.BAK`. If you save a revised interface and then decide you don't like your revisions, you can always reload the `.G.BAK` file to undo your revisions. Files with a `.BAK` extension do not appear in the file chooser's window. You must enter the name into the Name field to load these files.

If you have not yet saved an interface and no other `.G` files are currently loaded, choosing Save brings up the Save As... file chooser (discussed below). If more than one interface appears on the workspace, choose the default Save submenu choice All to save all the interfaces. A Notice appears for each interface that hasn't been saved yet.

You can save a project file, such as `foo.P`, by choosing All from the Save submenu, provided all `.G` files contained in `foo.P` are loaded.

Save As...

Save As... saves a new user interface or lets you save a previously saved user interface or project file under a new file name. It is enabled when you have loaded a `.P` file or at least one `.G` file. (Only one `.P` file can be loaded at any time.) Save As... has a submenu containing all currently loaded `.G` files as items. It looks like the Save submenu shown in Figure 3-4 above, except that it is labeled Save As instead of Save.

When you choose an item in the Save As... submenu, the Save As file chooser appears (shown in Figure 3-5). The window contains a list of all `.P` and `.G` files and all subdirectories in your current directory. Double-clicking on a directory descends into the directory and lists all its `.P` and `.G` files and all its subdirectories. Type in the new name of your file on the line at the bottom of the file chooser.

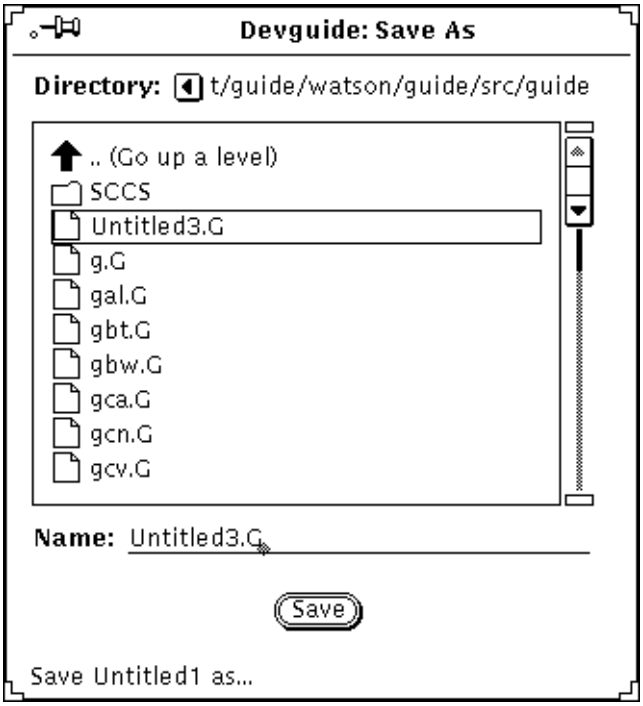


Figure 3-5 Save As File Chooser

If you haven't previously set a path name and a file name, Devguide proposes `Untitled` (with a number appended) as the interface file name and uses the current directory as the file's directory. Note that you must use standard operating system conventions when entering a path name or file name. Note also that when Devguide saves the interface, it appends `.G` to whatever file name you enter, if you have not already done so. Devguide appends `.P` to whatever file name you enter when you save a project file, if the file name you provide does not include the `.P` extension.

If you try to save an interface using a file name that already exists, Devguide displays a Notice window asking if you want to overwrite an existing file. To proceed and overwrite the existing file, choose the Save button. To cancel the save and protect the existing file, choose the Don't Save button. Note that if you do decide to overwrite an existing file, Devguide turns the existing file into a backup file as described in the previous section about the Save command.

Project...

To create a new project (`.P` file) or modify an existing one, choose Project... from the File menu. When you choose Project..., the Project Organizer window appears. The window contains all currently loaded `.G` files. You can create a new project if you have not loaded a project already. The Project Organizer is discussed in "Assembling Interfaces into a Project" on page 87.

Code Generator...

To generate and compile GXV or GOLIT code, and to run applications for these code generators, without using the command line, choose Code Generator... from the File menu. When you do this, the Code Generator tool appears. This tool is discussed in "Using the Code Generator Tool" on page 101.

Unload

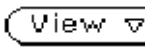
Unload unloads one or more interfaces that are currently open in Devguide, removing all the interfaces' elements from the workspace. It is enabled when you have loaded at least one `.G` file. Unload has a submenu containing all currently loaded `.G` files as items. The submenu contains a default item labeled All when more than one `.G` file appears in the submenu list. Choose All to unload all currently loaded interfaces. The Unload submenu looks like the Save submenu shown in Figure 3-4 on page 35, except that it is labeled Unload instead of Save.

If you unload a file that has been edited but not saved, Devguide displays a Notice window stating, "You have unsaved edits," similar to that shown in Figure 2-1 on page 7. Click SELECT on the Save button to save your work. If you want to unload an interface without saving your work, choose the Discard Changes button. If you want to abandon the unloading process so you can save your work before you unload the interface, choose Cancel.

If you unload a project that has been edited but not saved, Devguide displays a Notice stating that the project has unsaved edits. An edited project is one in which you have added or deleted `.G` files.

Devguide displays a third type of Notice if you edited more than one interface when you choose Unload All. Choose Overwrite All to save all edited files. Choose Overwrite selectively to save certain files. A Notice appears for each file that has been modified.

Viewing Layers, Windows, and Interfaces



Click MENU on View to display the View menu shown in Figure 3-6.

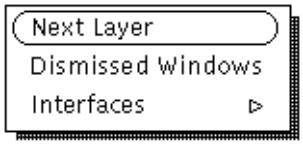


Figure 3-6 View Menu

Next Layer

If you select a pane in a layer of panes, Next Layer brings the next pane to the top of the parent base window. See “Layering Panes” on page 99 for a discussion of layered panes.

Dismissed Windows

Selecting Dismissed Windows returns any pop-up windows to the workspace that you earlier dismissed from the current user interface by choosing Dismiss from their window menu. Note that Dismissed Windows won’t open base windows or restore deleted windows.

You can also recover any dismissed window by selecting it in the Pop-up Windows scrolling list.

Interfaces

Choose Interfaces to display a submenu containing a list of all `.G` files that you have loaded into Devguide (see Figure 3-7). The default choice All appears in the submenu if more than one `.G` file is listed. When you choose a `.G` file name, the Interface Browser for that interface appears on the workspace. When you choose All, an Interface Browser for each listed interface appears on the workspace.

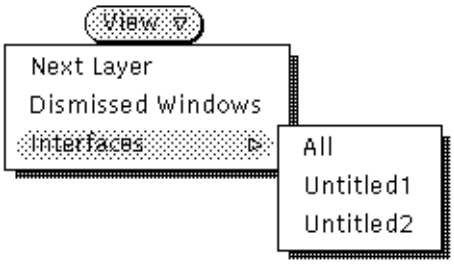
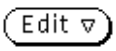


Figure 3-7 Interface Submenu

Editing Interfaces



Click MENU on Edit to display the Edit menu shown in Figure 3-8. This menu contains items that help you move or copy selected elements from a user interface to other interfaces. Note that each of these editing functions is also available in element pop-up menus (Figure 3-20 on page 60).

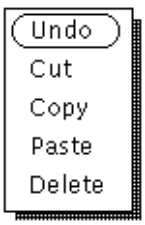


Figure 3-8 Edit Menu

Undo

Undo reverses the last editing action. Use it to retrieve deleted or cut elements, or to remove pasted elements. Note that Undo does not restore the contents of the elements clipboard that were removed by an editing action.

Cut

Cut removes any selected UI elements from the workspace and places them on the element clipboard. Each time you use Cut, you replace the contents of the clipboard with the elements you cut.

Copy

Copy copies any selected UI elements in the workspace and places their copies on the element clipboard. Each time you use Copy, you replace the contents of the clipboard with the elements you copied. Copy *does not* remove any UI elements from the workspace.

Paste

Paste pastes any UI elements currently on the element clipboard into the currently open user interface.

If you have a window on the clipboard, first deselect it, then choose Paste to put the window on the workspace.

If you have a control area or a pane on the clipboard, you must first select a UI window before you can use Paste. When you use Paste, the control area or pane appears in the selected window.

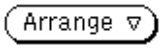
If you have one or more elements on the clipboard, you must first select a UI control area before you can use Paste. When you paste the controls, they appear in the selected control area.

If you have mixed elements on the clipboard—for example, a window with a control area and controls—Paste follows the rules for pasting the highest-level containing element. For example, you can paste a window with a control area and controls anywhere on the workspace, but you can only paste a control area with controls within a selected window.

Delete

Delete removes any selected UI elements from the workspace without placing them on the elements clipboard. You cannot retrieve deleted elements by pasting them from the clipboard.

Arranging Elements



Click MENU on Arrange to display the Arrange menu shown in Figure 3-9. This menu contains items that help you arrange UI elements in a user interface.

Figure 3-9 Arrange Menu

Group

Choosing Group combines all currently selected UI control elements into a group. See “Controls” on page 11 for a list of all UI control elements, and “Grouping UI Elements” on page 75 for a full description of groups.

Ungroup

Choosing Ungroup decomposes the currently selected group into its constituent UI elements. See “Grouping UI Elements” on page 75 for a full description of groups.

Align...

Choosing Align... displays the Align pop-up window shown in Figure 3-10. You can align selected sets of controls using the options in the Align window.

If you want to leave selected controls' vertical and horizontal alignments as they appear, do not select any of the vertical or horizontal alignment options. Deselect any selected option that you do not want to include in your alignment scheme.

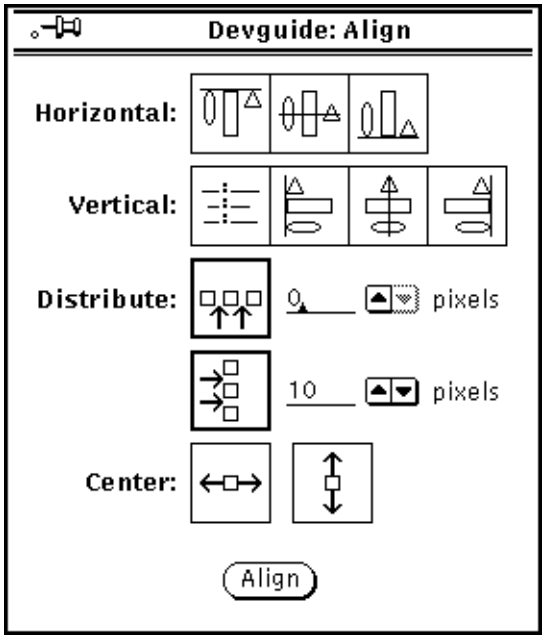


Figure 3-10 Align Pop-up Window

You see four alignment categories in the window:

Horizontal

Moves the selected controls to line them up in a single row. You can choose from three horizontal alignment schemes:

Aligns the UI elements with respect to the top of the highest element.

Aligns the UI elements so that their centers fall on a common (invisible) horizontal line, with no shifting to the left or right.

Aligns the UI elements with respect to the bottom of the lowest element.

Vertical

Moves the selected controls to line them up in a single column. You can choose from four vertical alignment schemes:

Aligns the UI elements with respect to the right side of the topmost element's label. For example, if the labels of all selected elements end with a colon, the colons line up.

Aligns the UI elements with respect to the left side of the leftmost element.

Aligns the UI elements so that their centers fall on a common (invisible) vertical line, with no shifting up or down.

Aligns the UI elements with respect to the right side of the rightmost element.

Distribute

Distributes space evenly between selected UI elements. You can set the space length in pixels using the numeric text fields at the right.

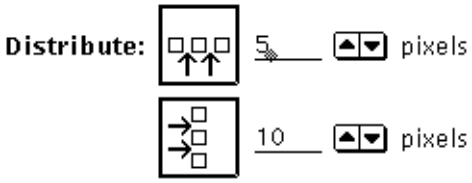


Figure 3-11 Distribute Options in the Align Window

- The top option in Figure 3-11 distributes space evenly between UI elements that are arranged horizontally. This will arrange elements horizontally as well as distribute space evenly if you also choose a Horizontal item.
- The bottom option in Figure 3-11 distributes space evenly between UI elements that are arranged vertically. This will arrange elements vertically as well as distribute space evenly if you also choose a Vertical item.

Center

Centers the selected UI elements with respect to the sides of the parent control area.

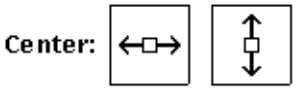


Figure 3-12 Center Options in the Align Window

- The first option in Figure 3-12 centers the bounding box of the selected elements with respect to the left and right borders of the parent control area.
- The second option in Figure 3-12 centers the bounding box of the selected elements with respect to the top and bottom of the parent control area.

You can get the same alignment effect by selecting a set of UI elements, and choosing the Align or Distribute submenus from an element's pop-up menu.

Grid

To set up a placement grid in a control area, choose a grid setting. Click MENU on the Arrange button and click MENU on the Grid item to display the Grid submenu shown in Figure 3-13. It offers grid settings of 1, 2, 4, 8, 10, 12, 16, 32, 48, and 64 pixels. If you choose a setting of two or more pixels, Devguide creates an invisible grid in the control area with grid squares the size of your grid setting. For example, if you choose a grid setting of four pixels, Devguide creates a grid of four-by-four-pixel squares. If you choose a setting of one pixel, you turn the grid off in the control area.

Note – Devguide has a default grid size of eight. That is, every time you start Devguide the grid size is set to eight, unless you customize your Devguide environment. See “Customizing Devguide’s Properties” on page 52 for a discussion of how to change this default value.

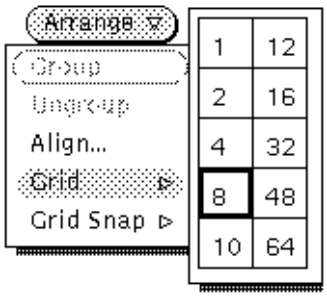


Figure 3-13 Grid Submenu

Any controls already in place in the control area don’t change position when you change the grid setting. However, once you turn the grid on, any new controls you place or move in the control area snap into place, aligned by Devguide so their upper-left corner falls on a grid intersection (as shown in Figure 3-14). The grid is on by default; you can turn it off and on using Grid Snap, described below.

If you choose larger grid settings, the controls will fall into more readily noticeable columns and rows. The Grid size you choose applies to any control area in your interface, during the current session with Devguide.

Note – You cannot display the grid. Figure 3-14 is only an illustration of aligning elements using the grid; the grid is always invisible.

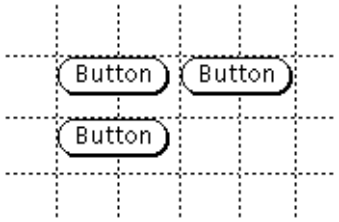


Figure 3-14 Using the Grid to Align Controls

Grid Snap

If a grid size has been chosen, you can use Grid Snap to temporarily turn the grid off and on. The grid is on by default. Click MENU on the Arrange button and click MENU on the Grid Snap item to display the Grid Snap submenu. Two items—On and Off—appear; click SELECT or MENU on your choice:

- On – Turns on the grid.
- Off – Turns off the grid. To reinstate the chosen grid size, choose On.

Editing Features Using Property Windows

Click MENU on Properties to display the Properties menu shown in Figure 3-15. This menu contains items that open windows to customize Devguide's properties and edit the properties of user interface elements.

Figure 3-15 Properties Menu

Selection...

Selection... opens the property window for a single UI element that is selected on the workspace. It is the default menu item for the Properties menu.

Devguide...

Devguide... opens the Devguide Properties window. Use this window to choose a default toolkit and set the default grid size for Devguide.

You can learn more about the Devguide Properties window in “Customizing Devguide’s Properties” on page 52.

Windows

Windows contains items to open base window and pop-up window property windows:

- Base Windows...
- Pop-up Windows...

When you choose one of these element items from the Windows submenu, the Properties menu closes and a property window for that type of element appears on the workspace. See Chapter 4, “Devguide Reference” for element property window descriptions.

Panes

Panes contains items to open property windows for the UI pane elements:

- Control Areas...
- Canvas Panes...
- Term Panes...
- Text Panes...

When you choose one of these element items from the Panes submenu, the Properties menu closes and a property window for that type of element appears on the workspace. See Chapter 4, “Devguide Reference” for element property window descriptions.

Controls

Controls contains items to open property windows for the UI control elements:

- Buttons...
- Messages...
- Settings...
- Text Fields...
- Sliders...
- Gauges...
- Lists...
- Drop Targets...

When you choose one of these element items from the Controls submenu, the Properties menu closes and a property window for that type of element appears on the workspace. See Chapter 4, “Devguide Reference” for element property window descriptions.

Help...

Help... opens the Help Editor window where you can enter and edit help text for elements of the user interface. You can learn how to use the Help Editor window in “Editing Help Text” on page 61.

Menus...

Menus... opens the Menu Editor Window where you can create and edit menus. For details on making and editing menus, read “Making and Editing Menus” on page 65. For details on attaching menus to buttons, canvas panes, control areas, and scrolling lists, read the description of those elements in the next chapter.

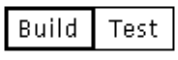
Groups...

Groups... opens the Groups window where you can choose layout, alignment, and anchoring schemes for your group. Read “Grouping UI Elements” on page 75 to learn how to use the Groups window and how to construct groups.

Connections...

Connections... opens the Connections Manager window where you can create and modify connections between UI elements. You can learn about connections and how to use the Connections Manager window by reading “Handling Events Using Connections” on page 81.

Building and Testing



The Interface Setting to the right of the Devguide control menu buttons offers two choices that determine the way UI elements on the workspace behave.

Build

Build is the default choice. When it is set, you can build a user interface, putting together UI elements from the Devguide window and editing them to change their size, position, properties, and other attributes. The UI elements

don’t act as they do when running in your final program. For example, when you click SELECT on a button, it doesn’t highlight and run a function. Instead, it selects the button so you can edit it.

Test

Test helps you test how your user interface works. When Test is set, UI elements act as if they were part of a finished user interface; that is, buttons and settings highlight, menus display, and so forth. You can pin menus to the workspace that are pinnable. Whatever Help text you have written appears. Initial values and states take effect. Connections supported by the XView toolkit work in Test mode if you use the actions and events provided by Devguide. That is, any predefined When and Action menu items for your toolkit that are also supported by the XView code generator work in Test mode.

You can’t edit any interface elements or load and save files when Test is selected, so when you’re finished testing you should return the interface setting to Build.

Using the Drag and Drop Target to Load GIL Files

The drag and drop target to the right of the Interface Setting offers you the option of loading a GIL file into Devguide by dropping the file onto the drag and drop target. Devguide provides four ways to drop a GIL file onto the target:

- Select text and drag it onto the target.
- Select one or more files from the File Manager and drag the file icons onto the target.
- Select one or more Mail attachment GIL files and drag the file icons onto the target.
- Select and drag an entire mail message that includes a GIL file onto the target. The mail header is automatically ignored.

When you perform any of these operations, the GIL files are loaded and the interfaces appear on the workspace.

Note – You can also load a GIL file into Devguide by dropping its file icon into the Project Organizer window. The corresponding interface appears on the workspace. If you have loaded a project, the new GIL file is added to the project.

Understanding the Element Information Area

The area in the Devguide window below the element glyphs is the element information area, shown in Figure 3-16. The information area presents information about elements on the workspace and the pointer's position as you move it over those elements.

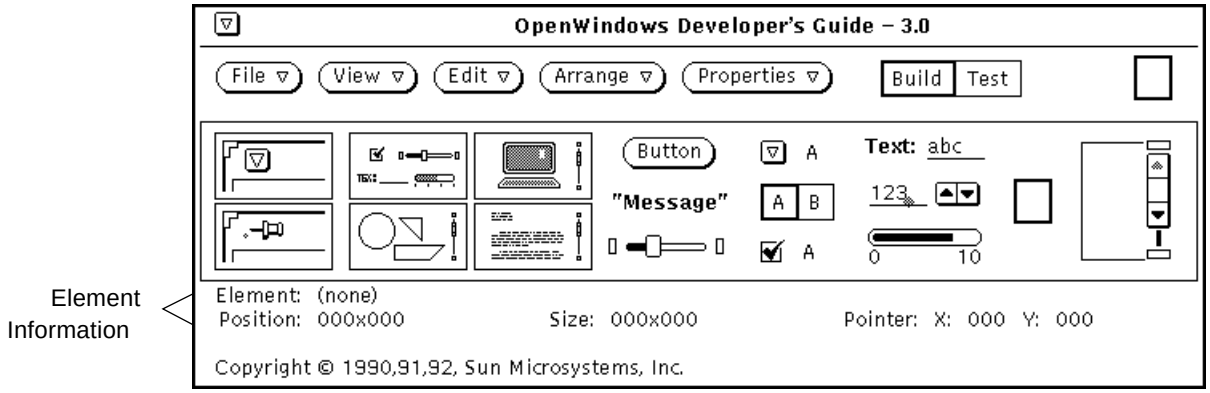


Figure 3-16 Devguide Element Information

Element Field

Element: window1

The Element field is the top message field in the information pane below the UI element glyphs. As you move the pointer around the workspace, the Element field displays the name of any UI element under the pointer. If there is more than one UI element under the pointer, the Element field displays the name of the top element. For example, if you move the pointer over a UI window with a control area full of controls, the Element field might display `window1` when the pointer is over the UI window's background, then `controls1` as the pointer moves over the UI control area, and finally `button1` when the pointer comes to rest on top of a UI button in the control area.

For layered panes, the Element field displays the layered panes' name, the relative position of the pane, and the number of panes in the layer. See "Layering Panes" on page 99 for details on layered panes.

Note that the Element field displays only the name of elements created by Devguide.

Position Field

Position: 254x464

When the pointer is on a UI element, this field gives the position of the element in terms of a pixel address.

- Windows are measured from the upper-left corner of the workspace.
- Panes and control areas are measured from the upper-left corner of the window.
- Controls are measured from the upper-left corner of the control area.

Size Field

Size: 200x128

When the pointer is on a UI element, this field displays its X (width) and Y (height) size in pixels.

Pointer Field

Pointer: X: 0 Y: 0

The Pointer field shows X (horizontal) and Y (vertical) coordinates that give the pointer's current location within the UI window, pane, or control area underneath it. The coordinates are measured in pixels from the upper-left corner of the UI window or pane. For example, as you move a pointer through unoccupied areas of a UI window, the Pointer field displays its coordinates as measured from the upper-left corner of the window. When you move the pointer into a UI pane or control area within the window, the Pointer field displays the coordinates as measured from the upper-left corner of the UI pane or control area instead of from the upper-left corner of the UI window.

Customizing Devguide's Properties

Devguide's Properties window allows you to specify a toolkit and default grid size for Devguide. Choose Devguide... from the Properties menu to display the Devguide Properties window, shown in Figure 3-17.

Note – You should configure Devguide to reflect your toolkit and grid size preferences the first time you start Devguide. After that, your selections are used as default values each time you start Devguide. You can change these values at any time by using Devguide's Properties window.

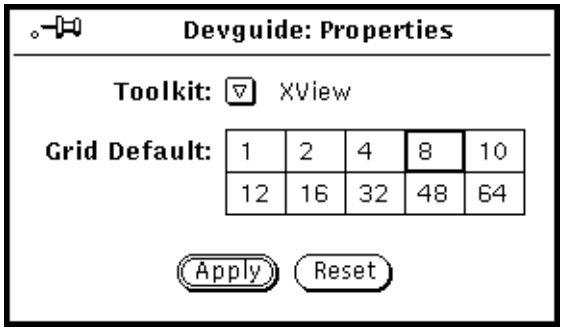


Figure 3-17 Devguide Properties Window

Specifying the Toolkit

Below the Devguide Properties window's header you find an abbreviated menu button labeled Toolkit. The default choice is the XView toolkit. Devguide also supports the OLIT toolkit. When you choose a toolkit, the Connections Manager window's When and Action menus appropriate for that toolkit are loaded. If you do not set a toolkit choice, Devguide assumes you are building an XView application, and configures the Connections Manager window menus accordingly. For further details on connections and the Connections Manager window, see "Handling Events Using Connections" on page 81.

If you plan to generate application code, you should choose a toolkit right after starting Devguide. Once you choose a toolkit, you can apply this choice to the current and future environment by clicking SELECT on the Apply button. Clicking SELECT on the Reset button returns the Toolkit setting to that in effect when you last chose Apply.

You should maintain a single toolkit choice throughout an application. The code generator you use generates code only for those connections it recognizes. For example, suppose you choose OLIT to design an interface and accidentally choose GXV to generate code. GXV then produces XView code and converts any OLIT-specific connections to generic (empty) callbacks.

Setting the Grid Default

The Grid Default setting, shown in Figure 3-17 on page 53, enables you to change Devguide's default grid size of eight. You can do this by selecting one of the available grid size settings. Once you choose a grid size, you can apply this choice to the current and future environment by clicking SELECT on the Apply button.

Clicking SELECT on the Reset button returns the Grid Default values to that in effect when you last chose Apply.

Producing a `.guidedefaults` *File*

If you change either of Devguide's properties at any time, Devguide produces a `.guidedefaults` file and stores the file in your home directory. This file contains the Toolkit and Grid Default selections set for your environment. `.guidedefaults` is a resource file similar to `.Xdefaults`. This file also stores the code generator that is used as the default when the code generator is invoked for the first time in a Devguide session.

Adding Elements to a User Interface

The glyphs in Devguide's palette represent elements you can add to a user interface. This section describes how you use the glyphs to place UI elements on the workspace to build a user interface, and how you edit the UI elements once they're in place.

Placing UI Elements

To place a UI element on the workspace, drag its glyph from the Devguide window. To drag an element glyph, put the pointer on it, press SELECT, then drag it onto the workspace. Release SELECT to drop the glyph—the element it represents then appears on the workspace where you dropped the glyph.

Controls

You can place a control UI element only within a control area. If the control area's grid is turned on, the upper-left corner of the control UI element snaps into place on the nearest grid intersection. See "Arranging Elements" on page 41 for more information about the grid.

Note – Devguide provides no glyph for an abbreviated menu button, a nonexclusive setting, or a multiline text field. To place an abbreviated menu button in your application, drag out the button glyph and set the element's Type to Abbreviated Menu. To place a nonexclusive setting in your application, drag out any setting glyph and set the element's Type menu to Nonexclusive. To place a multiline text field in your application, drag out either text field glyph and set the element's Field Type to Multiline.

Panes

You can place a pane only within a base window or pop-up window UI element. Any border of the pane you place that is within 32 pixels of an existing window or pane border snaps to the existing border.

You can layer panes on top of each other. If you drag any pane glyph directly onto an existing pane, the new pane is positioned and sized to the one underneath it. See "Layering Panes" on page 99 for details.

Sizing Windows and Panes

Window and pane UI elements can be re-sized once you drop them onto the workspace.

Windows

To change the size of a base or pop-up window, move the pointer onto any one of the window's four resize corners and drag the corner until you stretch the window into the size you want. You can also set the precise size of the window using the Rectangle line in the pane's property window.

Panes

To change the size of a pane, first select it. Once selected, the pane shows a boundary that includes resize corners and handles. To move two adjacent sides of the text pane at once, drag the resize corner touching both sides. To move a single side of the pane, drag the resize handle on that side. You can also set the precise size of the pane using the Rectangle line in the pane's property window.

Moving UI Elements

To move a UI element already in place, simply drag it to the desired position.

Windows

To move a base or pop-up window to a new location, drag it by its header or border.

Panes

After initial placement, panes follow grid snap when you move them. To move a pane UI element to a new location:

- Use the Move Pointer to drag the pane to its new location. The Move pointer, shown in Figure 3-18, appears after you select the pane and move the pointer between resize corners near the pane's border.

Figure 3-18 Move Pointer

- Use the resize corners to first stretch one corner into the new location you want, and to then stretch the other corners to fit the pane into the new location.

Note that you can't move a pane to any location outside of a UI base window or pop-up window.

Controls

To move a control UI element to a new location, simply drag it. Controls follow grid snap when you move them. You can also move a control by selecting it and using the keyboard arrow keys. You can align a particular control to other controls by using the Align item in the control's pop-up menu.

Note that you can't move a control to any location outside of a UI control area.

Cutting, Copying, and Pasting UI Elements

To cut or copy and paste UI elements, first select the element and choose cut or copy from its pop-up menu. For panes and control UI elements, select the target element and choose paste from its pop-up menu. The pane or control element (or a copy) appears on the target element. For base windows and pop-up windows, first deselect the window, then choose paste from the window's pop-up menu. The window (or a copy) is pasted on the workspace.

Use the editing commands in the UI element pop-up menu:

- Cut removes the UI element and its contents from the workspace and places it on the clipboard.
- Copy places a copy of the UI element and its contents on the clipboard.
- Paste places a copy of the clipboard contents onto appropriate areas of the workspace. The usual placement restrictions apply; for example, you can only paste a control UI element into a control area. You must first select the parent object into which to paste a UI element.
- Delete removes the UI element from the workspace without placing it on the clipboard.
- Undo reverses the last editing action.

You can cut, copy, and paste groups that you create. For more information about groups, see "Grouping UI Elements" on page 75.

If you cut (or copy) and paste two elements that are linked via a connection, the connection is cut (or copied) and pasted, too. For example, suppose you define a connection from button1 to pop-up1, and you copy the button and pop-up to make button2 and pop-up2, respectively. Then button2 is linked to pop-up2 via a connection with the same When and Action menu choices as the connection for button1 and pop-up1. For more information about connections, see “Handling Events Using Connections” on page 81.

Selecting and Deselecting Elements

You must select a UI element before you can edit it. You can select a single element either by clicking SELECT on the element in the workspace, or by opening a property window for that type of element and clicking SELECT on the element’s name in the scrolling list at the top of the window. Once you select an element, its name appears selected in the property-window scrolling list.

You can select a group of elements you have defined by selecting any element in the group. (See “Grouping UI Elements” on page 75 for a discussion of groups.)

To deselect a selected element, click ADJUST on it.

Note – If you have a two-button mouse, the ADJUST function is performed by holding down the Shift key and clicking the SELECT (left) button.

Note – When a window is selected, the word SELECTED appears in its right footer. When a canvas or pane is selected, it appears with resize handles. When other UI elements are selected, they appear with a box drawn around them.

Selecting Multiple Elements

If you want to dismiss, cut, copy, paste, delete, or move more than one UI element at a time, you can select several elements at once, using a technique called adjusted selection. You can use either one of two adjusted selection methods.

The first method is to select elements on the workspace by clicking ADJUST on them. Whenever you select an element using ADJUST, all previously selected elements remain selected, allowing you to select many elements at once. To deselect a single selected element (from a collection of selected elements, for example), click ADJUST on it. The element is deselected while all other selected elements remain selected. To deselect all elements but one, click SELECT on a single element. The element is selected while all previously selected elements are deselected. Click SELECT on the workspace to deselect all UI elements.

To use the second adjusted selection method, drag a bounding box across the UI controls you want to select in the control area. To drag the bounding box, move the pointer into a clear area within a control area or the workspace, then hold down SELECT, and drag the pointer directly across the controls you want to select. As you drag, an outline (bounding box) stretches to enclose the controls. When you completely enclose the controls you want, release SELECT and the enclosed controls are selected. See Figure 3-19 below for an example.

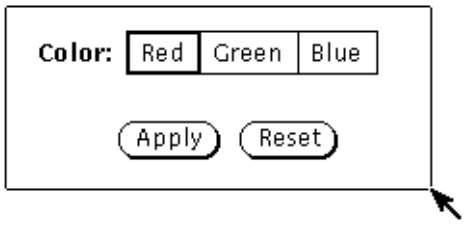


Figure 3-19 Selecting Using a Bounding Box

Once you select a collection of elements, each element indicates that it is selected. You can then press MENU to see a pop-up menu for the elements. If you select Cut, Copy, Paste, or Delete, Devguide performs your editing action on all the elements in the collection. If you use the Align... pop-up, Devguide aligns all of the controls in the collection to the grid in their control area. If you use adjusted selection to select a set of UI elements, you can drag the entire set to a new location by putting the pointer over any one of the elements, holding down SELECT, and dragging the pointer.

Opening UI Element Property Windows

Property windows are where you make changes to customize individual UI elements. You can open a property window in several ways.

- Double-click SELECT on a UI element to open its property window.
- Click SELECT on a UI element to select it, press MENU to display its pop-up menu (shown in Figure 3-20), and choose Properties to open its property window.

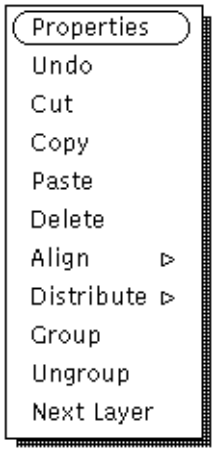


Figure 3-20 Devguide UI Element Pop-up Menu

- Click SELECT on a UI element to select it, and press the Properties menu button in the Devguide main window.
- Pop up the Properties menu (shown in Figure 3-15 on page 47) from the Devguide main window and choose the appropriate property window from the menu.

Note – The application area of a window can be entirely covered by a control area, text, term, or canvas pane. If so, you can access the Windows property window by double-clicking on the window's footer. (The window's header is under the control of the window manager. It cannot be used to access the window property window.)

Adding Glyphs and Icons to UI Elements

Many UI elements can have glyphs associated with them, such as for the element's label. Base windows have icons associated with them. In either case, the format of the file containing the glyph or icon image depends on the toolkit code generator you use.

Devguide supports two bitmap file formats: Sun Icon and Mono X bitmap (XBM). By toolkit, the file you specify *must* be:

- GXV – in Sun Icon or XBM file format
- GOLIT – in XBM file format

Editing Help Text

All interface elements except base windows and pop-up windows can include help text. Help text appears when you put the pointer over an element and press the Help (or F1) key.

To add help text to an element or to edit existing help text, use the Help Editor window shown in Figure 3-21.

Open the Help Editor window in either of two ways:

- Choose Help... from the Properties menu in the Devguide window.
- Choose the Help Text... button in an element properties window.

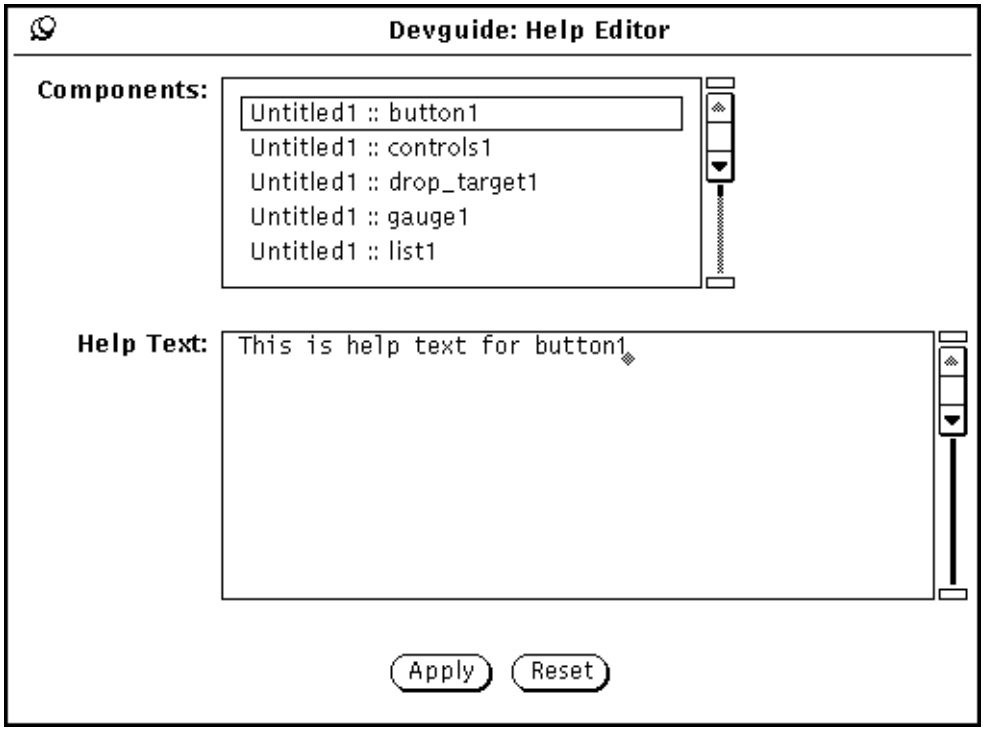


Figure 3-21 Help Editor Window

When Devguide creates source code, it includes help text attached to each element for which you created help text.

Components

The top of the Help Editor window is the Components scrolling list, which contains the names of all of the elements in the current user interface that can include help text. If you open the Help Editor window from an element's property window, the name of the element is automatically selected. To create or edit help text for any of the other elements, simply select the name of the element from the list.

Help Text

The text pane labeled Help Text in the bottom of the window contains the help text for the element selected in the Components list. It includes a vertical scrollbar you can use to scroll through the text, and also has a standard text pop-up menu. To see the pop-up menu, put the pointer on the text pane and click MENU. The pop-up menu contains five submenus that offer file handling, text views, text editing, text finding features, and extra features. All of these are standard features described in the *OpenWindows User's Manual.*

Choosing a Color

You can set the colors for each element in an interface. Windows, panes, and control areas each have a foreground color and a background color; controls have simply a color. To set any of these colors in an element, open the property window for the element and look for the Foreground, Background, or Color menu button. These three buttons have the same menu, shown in Figure 3-22.

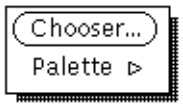


Figure 3-22 Color, Foreground, and Background Menu

You can choose a color in two ways:

- You can choose a color from a palette of colors displayed from the Palette submenu. Devguide then enters the color name into the text field to the right of the button. (This submenu is disabled if you have a monochrome monitor.)
- You can use the Color Chooser window. You can enter a color name directly into the text field to the right of the button if you know the proper color name. A rectangle of color appears to the left of the line.

The Palette submenu is an accelerator for the Color Chooser. Use the Color Chooser if you want to view each available color with its associated name.

You must click SELECT on the property window's Apply button to set the colors on the screen.

To display the palette of colors, press MENU on the Color, Foreground, or Background menu button and open the Palette submenu. Then click MENU over your color choice. The color's name appears on the line by the menu button, and a rectangle of color appears to the left of the line.

To open the Color Chooser window (shown in Figure 3-23), click MENU on Chooser..., or click SELECT on the Foreground, Background, or Color button in a property window.

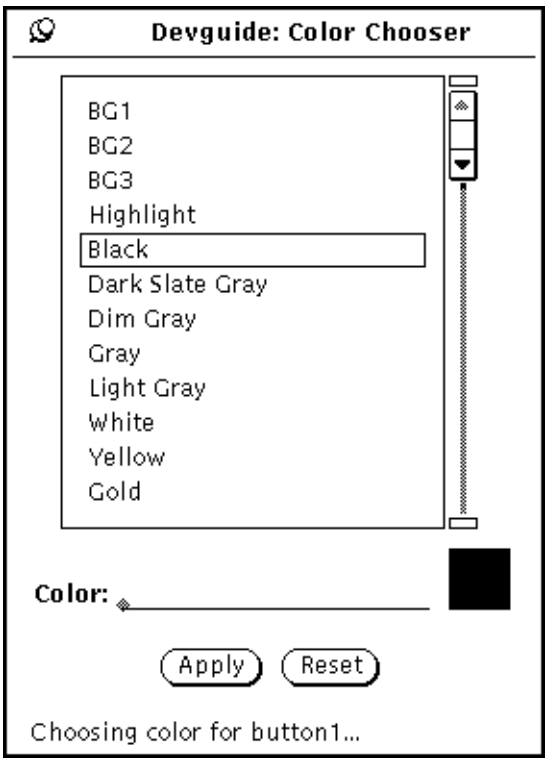


Figure 3-23 Color Chooser Window

The scrolling list at the top of the window offers you the full choice of colors. Scroll through the list to find a color you want, then click SELECT on it; the color name appears in the Color text field below the scrolling list, and the

square at the end of the text field turns the shade of the color you selected (unless you have a black and white monitor, in which case it remains black). You can also select a color by entering its name directly in the Color text field.

Once you have a color chosen, you can apply it to the selected UI element by clicking SELECT on the Apply button. The color name is inserted in the appropriate text field in the element's property window, and you see the element change to the color you set. If you decide you don't like your new color, choose the Reset button; the color reverts to the color that was set when you first opened the Color Chooser window.

Making and Editing Menus

A menu displays a set of options to a user so the user can choose one of them. There are two types of menus:

- A pop-up menu, which can appear anywhere on the workspace not covered by a control, and can be attached to different interface elements. A user usually displays a pop-up menu by moving the pointer onto an element and pressing MENU.
- A button menu, which is attached to a menu button. A menu button is a type of button that includes a menu mark to show that a menu is attached. A user displays a button menu by moving the pointer onto the menu button and pressing MENU.

The options displayed within a menu are called items, and can be labeled with text or with an image, as shown in the example in Figure 3-24.

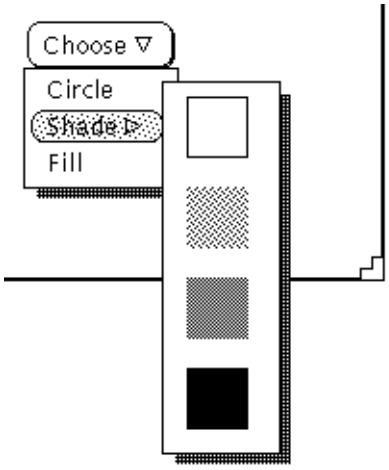


Figure 3-24 Button Menu with Submenu; Text and Image Items

When a user chooses an item, the item can do one of two things:

- Initiate an action by calling a procedure from the main body of the program
- Open a submenu with items of its own

Menus can have as many layers of submenus as you wish to add. Any item that opens a submenu is followed by a menu mark to indicate that a submenu is attached.

Menu Editor Window

To create and edit menus, items, and submenus, use the Menu Editor window shown in Figure 3-25. To display the window, choose Menus... from the Properties menu in Devguide's base window. You can also open this window by choosing New... from the Menu button menu on a control area, canvas pane, button, or scrolling list property window.

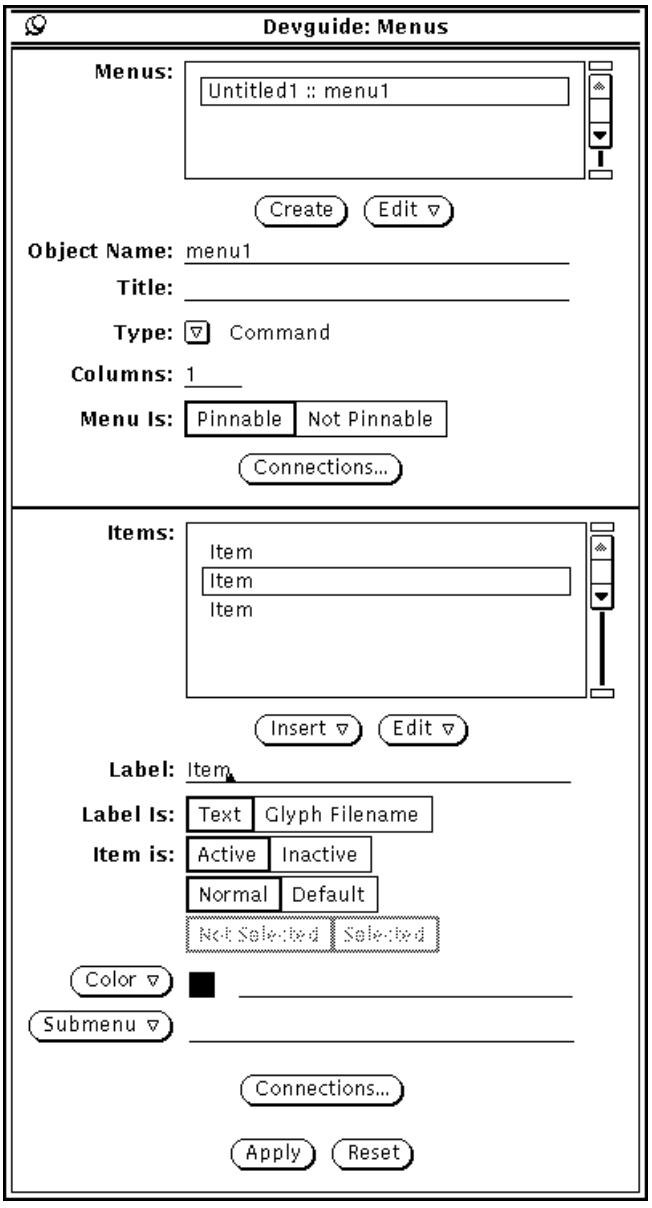


Figure 3-25 Menu Editor Window

Creating, Editing, and Moving Menus

The top half of the window contains the Menus list, a scrolling list that lists any menus created for the currently open user interface. It should be empty when you first start a new interface. Once you create menus, they appear in the list, where you can select any menu by putting the pointer on its name and clicking SELECT. The text fields and other controls below the Menus scrolling list (called the Menu controls) show and set the properties of the currently selected menu.

To add, delete, and move menus in the Menus list, use the Insert and Edit buttons below the list.

Create Button

Each time you click on the Create button, you create a new menu and add it to the list. It appears in alphabetical order within the other menus in the list. The menu controls below the Menus list display the properties of the new menu.

Edit Button

The Edit button below the Choices scrolling list offers items that cut, copy, paste, and delete menus within the Menus list. Use these items to move menus from one location to another in the list:

- Cut – Removes the selected menu from the scrolling list and places it in the menu clipboard. Cutting a menu removes all of the menu's items as well as the menu.
- Copy – Puts a copy of the selected menu into the menu clipboard. Copying a menu copies all of the menu's items and properties.
- Delete – Removes the selected menu from the scrolling list without putting it in the menu clipboard. The contents of the clipboard remain untouched when you choose this item. Deleting a menu removes all of the menu's items as well as the menu.

Setting Properties of the Selected Menu

Use the top half of the Menu Editor window to set the properties of the currently selected menu in the Menus list.

Object Name

Object Name establishes the name that Devguide gives the menu in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the menu.

Title

Contains the label that appears in the menu.

Type Menu Button

The Type menu button is an abbreviated menu button that offers you a choice of three different types of menus:

- Command – The most common type of menu, where each item initiates a procedure (a command) or calls up a submenu.
- Exclusive – A menu which presents items listing possible settings for a situation. When you choose an item, you turn on that setting and automatically turn off any other possible settings.
- Nonexclusive – A menu which presents items listing possible settings for a situation. When you choose an item, you turn on that setting. You can turn on as many settings in the menu at one time as you wish.

To choose one of the menu types, click MENU on the abbreviated menu button to open its menu. Click SELECT on whichever menu type you want to set. The menu closes, and your choice is displayed to the right of the menu button.

Columns

Columns contains the number of columns used to display items in the menu. The default setting is 1, where all items appear in a single column.

If you change the text field to a value of more than one, the menu's items appear in multiple columns. For example, if you set Columns to 3, the items appear in three columns. When the window system arranges the items in the menu, it uses the order of the items in the Items list, starting at the top of the first column and filling it to the bottom, then starting at the top of the next

column and filling it to the bottom. Figure 3-26 shows a menu with three columns. The numbered items show how the window system arranges the items.

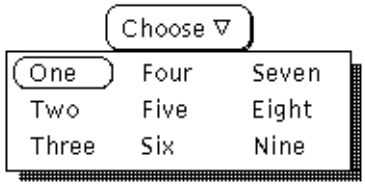


Figure 3-26 Multicolumn Menu

Menu Is

Adds a pushpin to or removes a pushpin from a menu.

- Pinnable – Menu appears with a pushpin and menu is pinnable
- Not Pinnable – Menu appears without a pushpin

Connections...

The Connections Manager window appears. You can established connections between the menu and other existing UI elements.

Inserting and Editing Menu Items

The bottom half of the window contains the Items list, a scrolling list that lists any items contained in the currently selected menu in the Menus list. It has a menu button labeled “Items” to its left. It should be empty when you first start a new interface, and should remain empty until you create a menu and add items to that menu. Once you create items, they appear in the list where you can select any single item by putting the pointer on its name and clicking SELECT. The text fields and other controls below the Items scrolling list (called the Item controls) show and set the properties of the currently selected item.

To add, delete, and move items in the Items list, use the Insert and Edit buttons below the Items list.

Insert Button

The Insert button offers items that insert a new menu item at different locations in the scrolling list. Choose one to insert a new item where you want it:

- Before – Inserts a new item before the selected item
- After – Inserts a new item after the selected item
- Top – Inserts a new item at the top of the scrolling list
- Bottom – Inserts a new item at the bottom of the scrolling list

Devguide names each item you insert “Item.”

Edit Button

The Edit button offers items that cut, copy, paste, and delete items within the scrolling list. Use these items to move items from one location to another:

- Cut – Removes the selected item from the scrolling list and places it in the item clipboard.
- Copy – Puts a copy of the selected item in the item clipboard.
- Paste – Puts the contents of the clipboard in the scrolling list. The four items in its submenu offer Before, After, Top, and Bottom, which insert the clipboard contents in the same locations as the corresponding items in the Insert button menu.
- Delete – Removes the selected item from the scrolling list without placing it in the item clipboard. It leaves the contents of the item clipboard intact.

Setting Properties of the Selected Menu Item

Use the bottom half of the Menu Editor window to set the properties of the currently selected item in the Items list.

Label

Contains the label (or glyph) that appears as the item’s label. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the item is labeled with text or with a glyph. See “Adding Glyphs and Icons to UI Elements” on page 61 for information on file formats.

- Text – Inserts the text entered in the Label text field as the item’s label.
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the item’s label.

Item Is

Item Is determines three important properties of menu items that become effective when the menu first appears. A default item appears in the menu enclosed within a lozenge. If the menu is attached to a button and the user clicks SELECT on the button, the user chooses the default item. Select Normal to ensure that an item isn’t the default item.

- Active – Makes the item active (able to be selected)
- Inactive – Makes the item inactive
- Normal – Makes the item a normal item
- Default – Makes the item a default item
- Not Selected – Makes the item be not selected
- Selected – Makes the item be selected

Note that only one item in each menu can be a default item.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color chooser. Color sets the menu’s color.

Submenu

Use Submenu to attach an existing menu to the item selected in the Items scrolling list. Submenu contains two items, New... and Names. Names has an attached submenu that lists existing menu names. Any name you choose from the Names menu is entered on the text field to the right of the Submenu button.

This menu becomes a submenu of the item selected in the Items scrolling list. It is attached so the submenu opens when the user selects the item in the finished interface.

You can enter the name of an existing menu in the Submenu text field, instead of choosing it from Names. If the menu you want to attach to the selected item does not yet exist, you must create it first. You then attach it as a submenu as just described.

New... is disabled because it is not applicable to the Menu Editor window.

Connections...

The Connections Manager window appears. You can establish connections between the menu item and other existing UI elements.

Applying Editing Changes to Menus and Menu Items

To apply the changes you made to a selected menu or menu item, click on the Apply button. Note that if you select a new menu or menu item without clicking on the Apply button, Devguide automatically applies all the editing changes you made to the previously selected menu or menu item.

Leaving the Menu Editor Window

To apply any changes you made in the Menu Editor window, choose the Apply button at the bottom of the window. If you dismiss the Menu Editor window (by clicking SELECT on its pushpin) without using the Apply button to make changes, your changes aren’t made to the currently selected menu or item. If you reopen the Menu Editor window, you’ll see your changes still in place, but unapplied.

To get rid of unapplied changes, choose the Reset button. It resets Menu and Item controls to their settings at the time you last chose the Apply button or chose a new menu from the Menu list.

Attaching a Menu to a UI Element

Once you create a menu in the Menu Editor window, you can attach it to a UI control area, canvas pane, scrolling list, or button. To do so, display the pane or control element's property window, press the Menu button, and select the menu's name from the Names submenu. Devguide enters the menu name on the line to the right of the Menu button. If you know the name of the menu you want, you can enter that name on the line yourself. When you click Apply on the element's property window, the menu is attached to the element.

Alternatively, you can display the selected element's property window, and click the Connections... button, or choose Connections... from the Properties menu. Choose the element as the Source, the menu as the Target, the When as Menu, and the Action as Show. Click Connect to establish the connection.

Detaching a Menu From a UI Element

To detach a menu from a UI element, either:

- Erase the menu's name from the Menu line in the element's property window and click SELECT on Apply, or
- Open the Connections Manager window, select the connection that attaches the menu to the UI element, and click Disconnect.

Viewing a Menu and Its Contents

If you want to see a menu with its items and submenus as it will appear in your finished user interface, enter Test mode by selecting the Test option in Devguide's window. Move the pointer over the element to which the menu is attached, then click MENU. The menu appears, displaying its items in text or images, as you labeled them. Any items with an attached submenu show a menu mark. To see the submenu, move the pointer over the item and click MENU. The submenu and all of its items appear. You can, of course, open submenus from within submenus, and continue down to as many menu layers as you created in the Menus Properties window.

Grouping UI Elements

A group is a collection of UI elements in your user interface that is treated as one unit. Once the elements within the group are positioned as you wish, you can move the group around, and the relative positioning of the individual objects remains the same.

You can place constraints on the objects in a group such as spacing and alignment. You can also place external constraints on the group, by *anchoring* it to another object. By supplying this information, code generators and toolkits can use *relative layout*; that is, one object is positioned in relation to another.

Creating a Group

To create a group, you first select the UI elements to compose a group. (Use a bounding box, or SELECT the first element and include the rest using ADJUST.) Choose Group from either the Arrange menu or from an element's pop-up menu to form the group. Use the Groups property window to give the group a name.

Determining a Group's Bounding Box

A group's bounding box is determined by the layout scheme you choose. For As Is, Vertical, and Horizontal, the bounding box is the smallest box that fits around all the group elements.

A matrix's bounding box is made up of cells. The height of a cell is the height of the tallest group element. The width of a cell is the width of the widest group element. A matrix's bounding box is *m* cells long and *n* cells high, where *m* is the number of elements in the matrix's longest row, and *n* is the number of elements in the matrix's longest column. Each element in the matrix is placed within a cell.

Ungrouping and Modifying Groups

To ungroup, first select the group by clicking SELECT inside the bounding box which surrounds the group elements. Then choose Ungroup from either the Arrange menu or from an element's pop-up menu.

To add elements to a group, you must first ungroup. The individual elements remain selected. Then extend the selection to the additional elements you wish present in the group. Finally, create a new group as described above.

You can delete elements from a group and still retain them as part of your interface. To do this, you must first ungroup. Then deselect any group element you wish removed from the group. Finally, create a new group as described above.

Editing the Properties of a Group

Click MENU on Groups... from Devguide's Properties menu to display the Groups window, shown in Figure 3-27. You can name a group, and modify the group's layout and alignment using the Groups window.

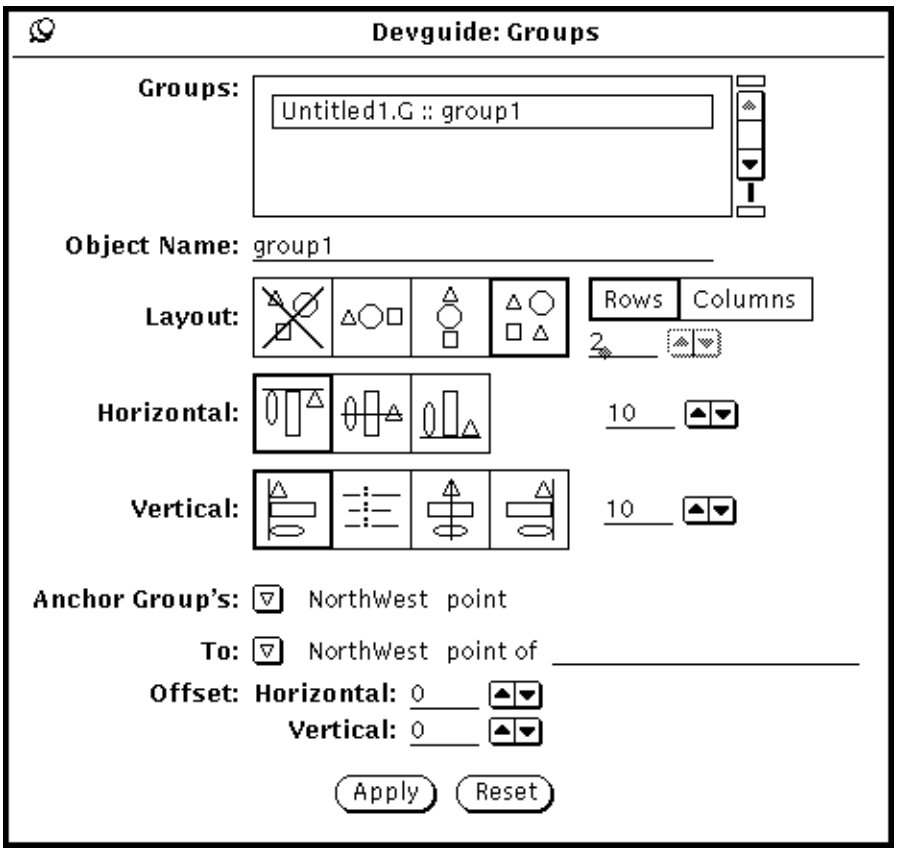


Figure 3-27 Groups Window

Groups List

A scrolling list appears at the top of the Groups window. This lists any groups created for the currently open user interface. It should be empty when you first start a new interface. Once you create groups, they appear in the list. You can select any single group by putting the pointer on its name and clicking SELECT. The Object Name field and layout information particular to that group appear in the Groups window.

Object Name

Object Name establishes the name that Devguide gives the group in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the group.

Layout

You can choose from four schemes to set the group's layout: As Is, Row, Column, and Matrix.

As Is

As Is preserves the layout of the group as you have positioned elements on the workspace. The position of a group element is fixed with respect to the upper-left corner of the group's bounding box. If you later change fonts, internationalize your application, or if element sizes change for some reason, the group elements may overlap. To avoid this situation, use one of the relative layout schemes discussed below.

Row

Row places all group elements in a horizontal line. The elements are evenly spaced across the line. You set the number of pixels between elements using the numeric text field to the right of the Row setting. The default is 10 pixels.

Column

Column places all group elements in a vertical line. The elements are evenly spaced down the line. You set the number of pixels between elements using the numeric text field to the right of the Column setting. The default is 10 pixels.

Matrix

Matrix places the group elements in a matrix format. You must specify a fixed number of rows or columns for the layout. The minimum number of rows (columns) allowed is 2. The maximum number of rows (columns) allowed is the number of elements in the group.

You set a fixed number of rows or columns by using the Rows/Columns exclusive setting and the numeric text field to the right of the matrix layout. You set the number of pixels between horizontally and vertically placed cells using the numeric text fields to the right of the Horizontal and Vertical settings.

To arrange the group elements into a matrix, Devguide first orders the elements in the group. The matrix is filled using a row-fill-first technique if you specify a fixed number of rows, and a column-fill-first technique if you specify a fixed number of columns. In row-fill-first, the matrix rows are created one by one, starting with the first element in the group ordering. In column-fill-first, the matrix columns are created one by one, starting with the first element in the group ordering.

- Rows – For a fixed number of rows, the number of columns in the group is determined by the number of group elements divided by the number of rows. If this number is an integer, it is used for the number of columns. If not, another column is created. This new column contains as many elements as the remainder indicates.
- Columns – For a fixed number of columns, the number of rows in the group is determined by the number of group elements divided by the number of columns. If this number is an integer, it is used for the number of rows. If not, another row is created. This new row contains as many elements as the remainder indicates.

Horizontal

You can choose from three horizontal alignment schemes. The Horizontal setting is active only when you choose a horizontal or matrix layout for the group. If you choose a matrix layout, the horizontal alignment you choose will be applied row by row. You can adjust the number of pixels between elements using the numeric text field to the right of the setting.

Aligns the UI elements with respect to the top of the highest element in the row.

Aligns the UI elements so that their centers fall on a common (invisible) horizontal line, with no shifting to the left or right.

Aligns with respect to the bottom of the lowest element in the row.

Vertical

You can choose from four vertical alignment schemes. The Vertical setting is active only when you choose a vertical or matrix layout for the group. If you choose a matrix layout, the vertical alignment you choose will be applied column by column. You can adjust the number of pixels between elements using the numeric text field to the right of the setting.

Aligns with respect to the left side of the leftmost element in the column.

Aligns with respect to the label of the leftmost element in the column.

Aligns the UI elements so that their centers fall on a common (invisible) vertical line, with no shifting up or down.

Aligns with respect to the right side of the rightmost element in the column.

Anchors

Devguide defines compass points for the bounding box of any UI element or group. The compass points are: Northwest (NW), North (N), Northeast (NE), West (W), Center (C), East (E), Southwest (SW), South (S), Southeast (SE). You can anchor any compass point on a group's bounding box to any other group or UI element in your interface using anchors. This allows you to define relative layouts between the group and any object in your interface.

- Anchor Group's – An abbreviated menu button is used to choose a compass point on a group. Choose a point on the group that you want to anchor to a point on another object in your interface.

- To – An abbreviated menu button is attached to a setting stack containing compass points for an object's bounding box. Fill in the object's name on the line to the right of the menu button. Use the stack to choose a point on the object's bounding box that you want to use as an anchor for the group.
- Offset – Once you have chosen an object and anchor points for the group and the object, you must choose a relative offset between compass points. You set a Horizontal and Vertical component to the offset using the numeric text fields provided.

As an example, if you want to anchor the North (top) of one group, group1, to the bottom of another group, group2, you would choose:

From: **`North`** point

To: **`South`** of **`group2`**

Offset: Horizontal: **`0`**

Vertical: **`10`**

This will place group1's top edge 10 pixels below the bottom edge of group2.

Devguide's coordinate system puts (0,0) at a window or pane's Northwest compass point. Vertical displacement increases downward; horizontal displacement increases to the right. You can assign a negative number to either Horizontal or Vertical (or both). A negative Horizontal displacement represents a shift to the left relative to a fixed point. A negative Vertical displacement represents an upward shift.

Handling Events Using Connections

To make your interface handle events, you specify *connections* in Devguide. When one object (called the *Source*) receives an event, a connection triggers a second object (called the *Target*) to respond by performing a specified action. For example, you might want to specify that when SELECT is clicked (the event) on a button (the Source), a popup (the Target) appears (the Action).

Use the Connections Manager window (shown in Figure 3-28 on page 83) to define a connection. Your toolkit choice determines the Connections Manager window's configuration. Use Devguide's Properties window (discussed in "Customizing Devguide's Properties" on page 52) to choose a toolkit before you start assembling your interface. The default choice is XView.

The events and actions recognized by the toolkit you choose are described in the toolkit's programming guide. See "Who Should Use This Book" on page xvii for the programming guide titles.

You can find the list of acceptable connection configurations for a given object in the `.config` file for your toolkit. This file is in the `$GUIDEHOME/lib` subdirectory included with the Devguide software. See Appendix A, "Files Provided With Devguide" to read about Devguide's included files.

You can open the Connections Manager window in one of three different ways:

- Drag and Link – Use Meta-SELECT to make a connection between two objects. Once you press the Meta key, you can select a Source object and drag a connection line with attached plug to a Target object. (You must keep the tip of the plug inside the desired target's bounding box.) This procedure brings up the Connections Manager window. The source and target objects are selected in the Source and Target scrolling lists (described below). You can use drag and link for any objects except menus and list items.

Note – Use the key marked "Alt" if there is no Meta key (usually marked as a diamond) on your keyboard.

- Property windows – Each UI element property window has a Connections... button. The Connections Manager window appears when you click this button. The object that is selected in the property window is selected in the Source and Target scrolling lists in the Connections Manager window. Notify is chosen from the When menu, and CallFunction is chosen from the Action menu. Insert a function name to create a notify handler.
- Properties menu – Choose Connections... from Devguide's Properties menu to open the Connections Manager window. The first time you do this, all base windows in the currently loaded interfaces are listed in the Source and Target scrolling lists (described below). When you subsequently choose Connections... from the Properties menu, the Connections Manager window will appear as it did the last time it was pinned to the workspace.

Note – You can resize the Connections Manager window to see entire names of items in all scrolling lists.

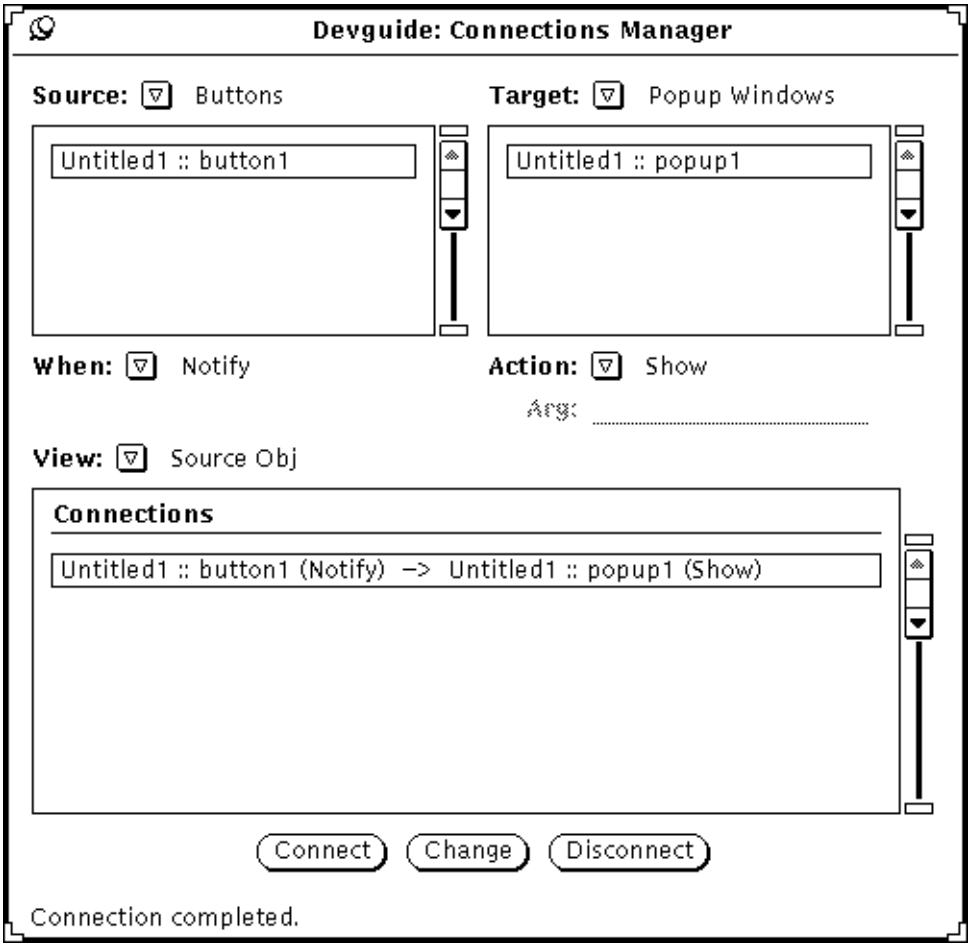


Figure 3-28 Connections Manager Window

Choosing the Source of a Connection

In the upper-left corner of the Connections Manager window you see an abbreviated menu button labeled Source, with a scrolling list underneath. The Source menu contains all UI element types, as well as menu items and setting items. When you choose a Source type, all objects of that type that are in currently open interfaces are listed in the scrolling list below the menu button.

The default choice for the menu is Base Windows. If you open the Connections Manager window using a property window or by using drag and link, you have already specified a source. The appropriate object type will be chosen from the menu, and the source you chose will be selected in the scrolling list. Otherwise, choose the menu item appropriate to the type of connection you wish to make. Then select the item in the scrolling list that you want as the source of the connection.

Choosing the Target of a Connection

In the upper-right corner of the Connections Manager window you see an abbreviated menu button labeled Target, with a scrolling list underneath. The contents of the Target menu depend on the source you choose. The Target menu contains the list of acceptable targets for a given source object. At most it will contain all UI element types, as well as menu items and setting items. This is the case, for example, if you choose a button as the source of a connection. When you choose a menu item, all objects of that type that are in currently open interfaces are listed in the scrolling list below the menu button.

The default choice for the menu is Base Windows. If you open the Connections Manager window using drag and link, you have already specified a target. The appropriate object type will be chosen from the menu, and the target you chose will be selected in the scrolling list. Otherwise, choose the menu item appropriate to the type of connection you wish to make. Then select the item in the scrolling list that you want as the target of the connection.

If you open the Connections Manager window using the Connections... button from a property window, the UI element you selected in the window will appear selected as both source and target for a new connection. The target should be changed if you want to make a direct connection to a different UI element.

Choosing the Event for a Connection

Below the Source scrolling list you find an abbreviated menu button labeled When. The items in the When menu vary depending on the source type you choose for a connection (for example, Buttons). The When menu choices comprise the list of all events that can be recognized by the source object.

Choose the menu item appropriate to the type of connection you wish to make.

Choosing the Action for a Connection

Below the Target scrolling list you find an abbreviated menu button labeled Action. The items in the Action menu vary depending on the target type you choose for a connection (for example, Buttons). The Action menu choices comprise the list of all actions that can be initiated by the target object.

Choose the menu item appropriate to the type of connection you wish to make. Some action types require an argument. If so, the Arg text field is enabled. The text field's label changes to describe the kind of argument required. For example, CallFunction requires a function name, so the label changes to Function.

Defining Callbacks

All action types displayed as Action menu choices are predefined by your toolkit, except CallFunction and ExecuteCode. For CallFunction, specify the callback name that initiates the connection's action. If you choose ExecuteCode, Devguide displays a pop-up window labeled Define Callback. You can enter the code you wish to include as the action response in the Define Callback pop-up window.

Viewing Lists of Connections

The items in the View Menu determine the types of connections that are displayed in the Connections scrolling list.

The default choice for the View Menu is Source Obj. Those connections having the item selected in the Source scrolling list as the source of the connection are displayed.

All connections are displayed if you choose All from the View menu. When you choose a Menu item other than Source Obj or All, all connections with a source of that type are displayed in the scrolling list.

Connections Scrolling List

Below the View Menu you find a scrolling list labeled Connections. The Connections scrolling list displays a connection description in the following format:

sourcename (*eventname*) —> *targetname* (*actionname*)

where *sourcename* is the full name of the source object, *eventname* is the name of the event you choose, *targetname* is the full name of the target object, and *actionname* is the name of the action you choose. An object's full name includes the interface name and the object name separated by two colons.

Establishing a Connection

To establish a connection, you must select a source object from the Source scrolling list, a target object from the Target scrolling list, an event from the When menu, and an action from the Action menu. Once you have done that, you click Connect. The newly made connection appears in the Connections scrolling list.

Modifying a Connection

Select the connection you wish to change in the Connections scrolling list. Alter the connection, by changing its source, target, event, and/or action. Then click Change. The connection is now modified in the Connections scrolling list.

Deleting a Connection

You can eliminate an established connection by selecting it in the Connections scrolling list and clicking Disconnect.

Tying a Pop-Up Window to a Button

Many interfaces include a pop-up window tied to a button. Use Devguide to create both the pop-up window and the button to which it's tied. Choose the button as the Source of the connection and the pop-up as the Target. Then choose Notify from the When menu and Show from the Action menu. Click the Connect button to establish the connection.

Attaching a Notify Handler to an Object

To attach an XView Notify Handler to an object, establish a connection with that object as both the Source and Target. Choose Notify from the When menu and CallFunction from the Action menu. Insert the Notify Handler's name on the Arg line.

Connecting Objects in Different Interface Files

You can establish a connection between any object in any interface file, within the constraints defined by your toolkit. You *must* create a project if you connect an object in one interface file to an object in a different interface file. The project file contains the information necessary to implement the connection. When you establish this kind of connection, save your project file. Read "Assembling Interfaces into a Project" below to learn how to make a project.

Attaching a Menu to a UI Element

You can attach a menu to a control area, canvas pane, button, or scrolling list using connections. Use Devguide to create both the menu and the UI element to which it's attached. Choose the UI element as the Source of the connection and the menu as the Target. Then choose Menu from the When menu, and Show from the Action menu. Click the Connect button to establish the connection. You can also attach a menu to an element by using the Menu button on the UI element's property window.

Assembling Interfaces into a Project

Devguide allows you to group multiple `.G` (GIL) files for your application into a *project*. The term *interface* applies to the portion of your application created by a single `.G` file (and to your entire application). As you create and save interfaces using Devguide, a project serves as a convenient organizational tool. You can create a more modular application using projects.

The easiest way to construct a project is if you decide in advance how to organize your `.G` files. The first interface can be built by dragging elements onto the workspace, and then saving your work into a `.G` file. Each subsequent interface must be initiated by choosing New Interface... from Devguide's File menu. Once you do that, you can drag elements onto the workspace. Notice that the lower right corner of Devguide's base window tells you the name of

the currently edited interface. All elements you drag onto the workspace go into that interface. Once you finish with an interface, save it and repeat the process until your application is complete. Choose New Interface... from Devguide's File menu, build an interface, and save it to a `.G` file.

Devguide manages a project by creating a `.P` file. This file contains a list of all the `.G` files of the interfaces that comprise the project. It also contains any connections that you make between interfaces (the connection's source and target are in different interfaces).

This section discusses the operations you perform to create, modify, load, and save interfaces and projects.

Creating and Modifying Projects

If you use Devguide to create an interface with more than one window, it is generally a good idea to create each window as a separate interface and save the window in its own GIL file. You'll find that separating a large interface into one window per GIL file keeps the file sizes manageable during the compile. You can then create a project containing all the GIL (`.G`) files necessary for your interface. You can either compile them separately, or all at once. Devguide produces the correct makefile for your project.

To help keep your files straight during interface editing and compiling, you should create a single directory to store all the source code files for a single program: the GIL files for the interface, the toolkit-generated source code files for the interface, your own custom source code files for the software underlying the interface, and the `Makefile` that ties the files together during the compilation.

Using the Project Organizer

To make a project from your `.G` files, choose Project... from Devguide's File menu, or choose Load... and provide the name of a currently existing project. (See "Working with Files" on page 31 for further details on the File menu.). The Project Organizer window appears, containing an icon for each interface that is currently loaded. You must choose New from the Project Organizer's Project menu to create a new project. (Recall that only one project at a time can be open.) You can save this new project by choosing Save or Save As from the

Project menu. You can add existing `.G` files to the project by choosing Add from the Project Organizer's Interface menu. You can delete `.G` files appearing in the window by choosing Remove from the Interface menu.

The Project Organizer Window presents a visual representation of a project's contents. If you have loaded a project, the Project Organizer window contains an icon for each `.G` file contained in the project. Any other currently loaded interface is also considered part of an open project. The upper-right corner of the file appearing in an icon is folded over if the corresponding `.G` file is loaded. Figure 3-29 shows a project with two interfaces.

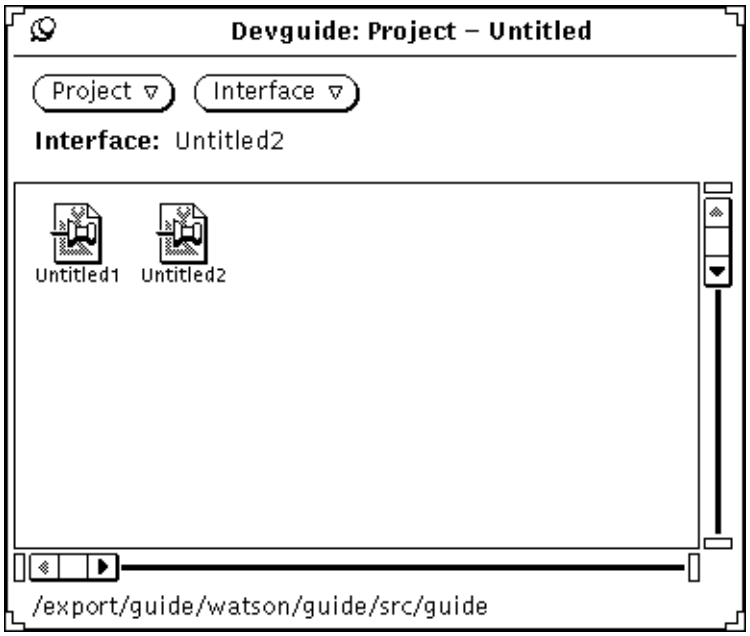


Figure 3-29 Project Organizer

The name of the currently loaded project is displayed in the Project Organizer's header. The header displays "None" if you haven't yet loaded a project. It displays "Untitled" if you have chosen New from the Project menu. The window's footer displays either the current directory or the directory of the project which is currently open.

The Project Organizer window contains a Project menu button and an Interface menu button. The items in these menus are described below.

Working with Projects

Click MENU on Project to display the Project menu shown in Figure 3-30.

Figure 3-30 Project Menu

The Project menu contains items that allow you to create a new project, or load, save, or unload an existing project.

Load...

Load... loads an existing project. It is disabled if a project is currently open. When you choose Load..., the Load file chooser appears. Double-click SELECT on a project file name, or enter the full path name of a project file and click Load.

New

New creates a new project. It is disabled if a project is currently open. When you choose New, the Project Organizer's header changes from "None" to "Untitled." An icon for every `.G` file you have already loaded appears in the Project Organizer window. These files are all considered as part of your new project.

Save

Save saves the current project file, as well as all `.G` files contained in the project. It is disabled if you have not yet created or loaded a project.

Save As...

Save As... saves a new project or renames an existing project. It is disabled if you have not yet created or loaded a project. When you choose Save As..., the Save Project As file chooser appears. Enter the project's new name (including proper directory if necessary). When you save a project, its name appears in the Project Organizer's window header.

Unload

Unload unloads the project whose name appears in the window header. It is disabled if you have not yet created or loaded a project.

Saving Projects

Devguide generates two files when you save a project named `projname`:

- `projname.P` – contains a list of all `.G` files and all connections between interfaces in your project.
- `projname.make` – contains a list of all the `.G` files that constitute your project. It is written in the `Makefile` format and is included in the Makefile generated when you run a toolkit code generator on your project file.

Devguide generates new `.P` and `.make` files every time you save your project.

Working with Interfaces

Click MENU on Interface to display the Interface menu, shown in Figure 3-31.

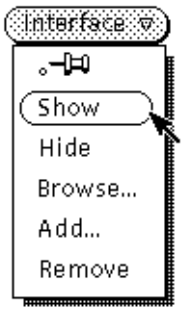


Figure 3-31 Interface Menu

The Interface menu contains items that allow you to add and remove `.G` files from the current project, and to perform operations pertaining to a `.G` file whose icon is selected in the Project Organizer window. You can select an icon by putting the pointer over it and clicking SELECT.

Show

Show loads the `.G` file whose icon is selected in the Project Organizer window. It is disabled if you have not selected an icon. You can also load a `.G` file by double-clicking on the file's icon.

Hide

Hide unloads the `.G` file whose icon is selected in the Project Organizer window. It is disabled if you have not selected an icon or if the selected icon's `.G` file is not loaded.

Browse...

Browse opens the Interface Browser for the interface whose icon is selected in the Project Organizer window. It is disabled if you have not selected an icon.

Add...

Add allows you to add `.G` files to the current project. It is never disabled. When you choose Add..., the Add Interface file chooser appears. The file chooser window contains a list of all `.G` files and subdirectories in your current directory. Double-clicking on a directory descends into the directory and lists

all its .G files and subdirectories. You can add a .G file to your project by double-clicking on the filename, or by entering its full path name on the line provided and clicking Add. If you add a .G file to your project, the Project Organizer window's header informs you that the project was modified.

Note – Add... does not load a .G file; it just adds the file to your project.

Remove

Remove removes a .G file whose icon is selected in the Project Organizer window from the current project. It also unloads the file if it is loaded. The file's icon is removed from the window. If you remove a .G file from your project, the Project Organizer window's header informs you that the project was modified.

Project Organizer Window Pop-up Menu

The Project Organizer window has a pop-up menu. It has the same items contained in the Interface menu. The pop-up menu is included for your convenience. It is sometimes quicker and easier to perform operations on .G files using the pop-up menu than to use the Interface menu.

Dropping a File into the Project Organizer Window

You can load a GIL file into Devguide by dropping its file icon into the Project Organizer window. The corresponding interface appears on the workspace. If you have loaded a project, the new GIL file is added to the project.

Creating and Modifying Interfaces

If you are unsure about the structure of your application, you can build several windows and/or menus, and divide them into separate files later on. This is easy to do using Interface Browsers (one is shown in Figure 3-32 on page 95). If your entire application is in one interface, first open the Interface Browser for that interface. Then choose New Interface... from Devguide's File menu. Now you can drag whatever windows or menus you want from the application's Interface Browser into the Browser for your new interface. Continue to divide off pieces of your interface in this manner, saving as you go along.

If you decide that one interface is too large, you can move part of it to another interface in the same way. Open the Interface Browser for the interface you want to edit. Then choose New Interface... and drag objects into the empty Interface Browser window.

Using the Interface Browser

The Interface Browser window displays a tree or icon representation of the interface whose file name appears in the window's header.

Open an Interface Browser by choosing one of:

- New Interface... from Devguide's File menu (for a new interface).
- Browse... from the Project Organizer's Interface menu. You must first select an interface icon in the Project Organizer's window.

The default view of the tree representation puts the root (a base window, pop-up window, or menu) at the window's left side, and the tree expands horizontally to the right. Each node in the tree is displayed as a UI element glyph, with the name of the element appearing below the glyph. A tree appears for each base window, pop-up window, and menu that is a part of the interface being represented.

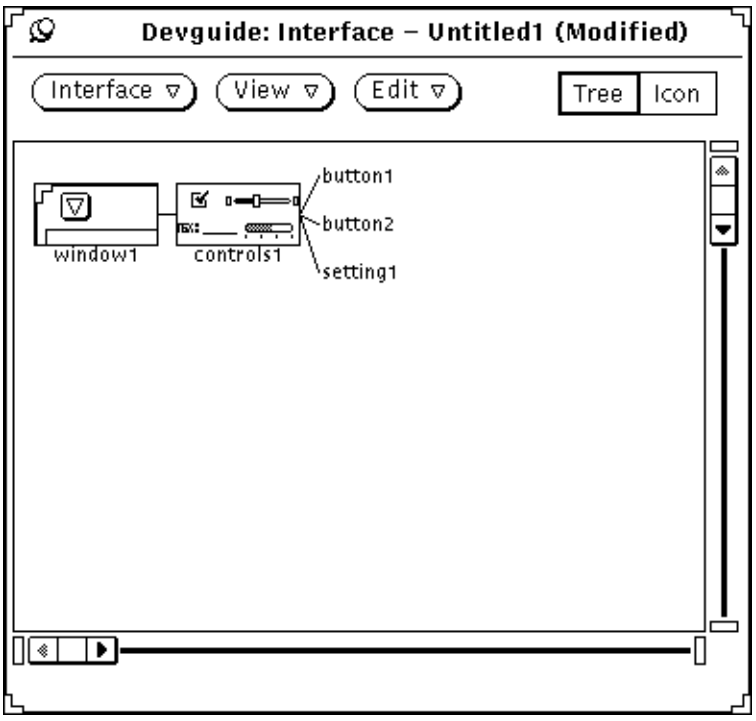


Figure 3-32 Interface Browser

The Interface Browser window contains an Interface menu button, a View menu button, and an Edit menu button. The items in these menus are described below. The Tree/Icon exclusive setting allows you to view an interface as a tree structure as a whole (Tree), or as separate icons in one level of the tree at a time (Icon).

Saving and Unloading Interface Files

Click MENU on Interface to display the Interface menu, shown in Figure 3-33.

Figure 3-33 Interface Menu

The Interface menu contains items that allow you to save, rename, or unload a `.G` file.

Save

Save saves the current interface. If the interface hasn't been saved yet, choosing Save opens up the Save As file chooser.

Save As...

Save As... allows you to name your interface for the first time, or rename it if it has already been saved. When you choose Save As..., the Save As file chooser appears.

Unload

Unload unloads the current interface.

Viewing Portions of the Interface Hierarchy

Click MENU on View to display the View menu, shown in Figure 3-34.

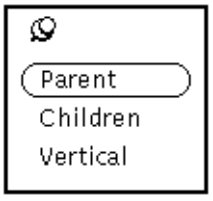


Figure 3-34 View Menu

First select an object in the Interface Browser window by clicking SELECT. When you do this, the object is also selected in the actual interface. The View menu items apply to the object that you select. Use the first two items to shift up and down an interface tree in either Tree or Icon mode.

Parent

Parent shifts the view of the tree if necessary, so that the selected object's parent becomes visible. (The view remains fixed if the selected object's parent is already visible.) It is disabled if no object is selected, or if you select the root of a tree. In Icon mode, Parent displays the icons of all elements on the same level as the parent of the selected object.

Children

Children shifts the view of the tree, so that the selected object and its children are visible. It is disabled if no object is selected, or if you select a leaf of a tree. In Icon mode, only the children of the selected object are visible.

Vertical

Vertical displays all trees in the Interface Browser window vertically. The roots are at the top of the window and the trees expand downward. When you choose Vertical, the Vertical view menu item toggles to Horizontal (the default representation). Vertical is disabled in Icon mode.

Editing Using Interface Browsers

Click MENU on Edit to display the Edit menu, shown in Figure 3-35. This menu contains items that help you copy or remove elements from a user interface, and move them to other interfaces. When you make changes to an interface, the Interface Browser window header informs you that the interface was modified. Performing editing operations on icons in Interface Browsers causes the corresponding interfaces to be updated accordingly.

Note that each of these editing items is also available in UI element pop-up menus and Devguide's Edit menu button.

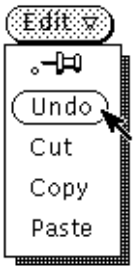


Figure 3-35 Edit Menu

Undo

Undo reverses the last editing action. Use it to retrieve deleted or cut elements, or to remove pasted elements. Note that Undo does not restore the contents of the elements clipboard that were removed by an editing action.

Cut

Cut removes any selected UI elements from the workspace and places them on the element clipboard. Each time you use Cut, you replace the contents of the clipboard with the elements you cut.

Copy

Copy copies any selected UI elements in the workspace and places the copies on the element clipboard. Each time you use Copy, you replace the contents of the clipboard with the elements you copied.

Paste

Paste pastes any UI elements currently on the element clipboard into the currently open user interface. If you have a window on the clipboard, you must first deselect the window. Then choose Paste from the Edit menu of the Interface Browser into which you want to paste.

If you have a control area or a panel on the clipboard, you must first select a UI window icon before you can use Paste. When you use Paste, the control area or panel appears in the selected window, and the Interface Browser is updated accordingly.

If you have one or more control elements on the clipboard, you must first select a UI control area icon before you can use Paste. When you paste the controls, they appear in the selected control area, and the Interface Browser is updated accordingly.

If you have mixed elements on the clipboard—for example, a window with a control area and controls—Paste follows the rules for pasting the highest-level containing element. For example, you can paste a window with a control area and controls anywhere on the workspace, but you can only paste a control area with controls within a selected window.

Interface Browser Window Pop-up Menu

The Interface Browser window has a pop-up menu. It has the same items contained in the Edit menu. The pop-up menu is included for your convenience. It is sometimes quicker and easier to perform edits using the pop-up menu than to use the Edit menu.

Dragging Objects Between Interface Browser Windows

If you create more than one interface on the workspace, you can edit any of them by selecting the appropriate window or menu in the interface. You can add or remove any objects from an interface by dragging or deleting. You can cut, copy, and paste any object from one interface to another. You can also use the Interface Browser to cut, copy, and paste objects, as described in “Editing Using Interface Browsers” on page 97. You can drag and drop windows and menus from one Interface Browser to another. This facilitates moving top level objects between interfaces. When you edit an interface, its Interface Browser window is updated accordingly. If you decide you do not want an interface as part of your project, you can delete it from the project by choosing Remove from the Project Organizer’s Interface menu.

Layering Panes

You can place multiple panes in the same area of the workspace. You can layer all four types of panes (text, term, canvas, and control area), using any combination. You can layer panes by placing a pane’s glyph anywhere on top of an existing pane. This configuration is sometimes referred to as *layered panes* or *a layer of panes*.

If you want to add a new pane to a layer of panes, you must drag its glyph directly onto the top of the layer. A pane is not included in a layer of panes if you first drag its glyph onto a different portion of the workspace, and later place the pane on top of the layer.

Sizing Layered Panes

Any pane you place on top of a pane or an existing configuration of layered panes is sized to the existing pane(s). If you want the layered panes to be different sizes, you can resize them by changing the W and H values of the Rectangle property in the appropriate property window(s). If you resize one pane using its resize handles, all layered panes assume that same size.

You can have layered panes in which multiple adjacent panes comprise the layers. For example, one layer can have two adjacent panes that fill a window, another layer can be one pane the size of the window. One way to do this is to first create the layer consisting of the two adjacent panes. Next, drop a pane glyph onto one of the existing panes, and resize it to fill the window by setting W and H values in the pane's property window.

Moving Layered Panes

If you drag the top pane in a layer of panes to change its location on the workspace, the entire layer of panes is dragged with the pane. You can move an individual pane by changing the X and Y values of the Rectangle property in its property window. Once you create a layer of panes, move it as a whole.

Scrolling through Layered Panes

You can scroll through layered panes by selecting the top pane and choosing Next Layer from Devguide's View Menu. Next Layer is the default item on the View menu. It also appears in a UI element's pop-up menu.

Devguide's Element field displays the name of the top pane when you move the pointer on top of the pane. In addition, the Element field provides a parenthetical message identifying the layered pane's name, the pane's relative position, and the total number of layered panes.

An example of this notation is:

Element: controls1 (layered_pane1, layer2 of 3)

In this case the pointer is over the control area named `controls1`, which is the second in three layered panes. The layer of panes is named `layered_pane1`.

Using the Code Generator Tool

Use Devguide's code generator tool to generate and compile code, and to run applications for these code generators, without having to type commands at the command line. The code generator tool's main window is resizeable.The inner pane is a command tool.

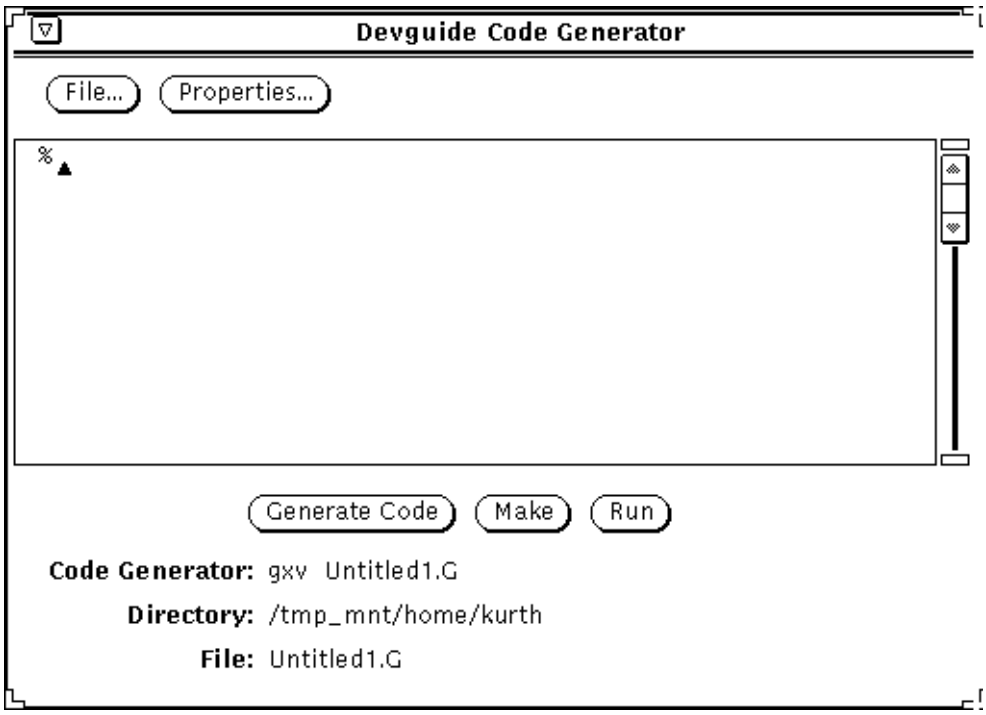


Figure 3-36 Code Generator Tool Main Window

Click SELECT on the File... button to display a file chooser pop-up window. Use this to select a GIL or project file. Click SELECT on the Properties... button to display a pop-up window to set make and run-time arguments, and code generator properties. Click SELECT on the Generate Code button to generate

code from your GIL or project file. Click SELECT on the Make button to compile source code and link object code. Click SELECT on the Run button to run the executable program.

Using the File... Button

When you click SELECT on the File... button, a file chooser pop-up window appears. When you select a file, the main window's Directory and File fields are filled in. The file name also appears after the code generator name in the Code Generator field.

See "File Chooser" on page 32 for more information about the file chooser.

Using the Properties... Button

When you click SELECT on the Properties... button, the Code Generator Properties pop-up window appears. The GXV Properties pop-up window is shown in Figure 3-37 on page 104.

Note – The code generator window can remain open regardless of whether the Devguide window is open.

Use the Code Generator abbreviated menu button to select the GXV, GXV++, or GOLIT code generator.

Note – If an application has already used a particular code generator to generate code from a GIL file, use the same code generator for that application. For example, if you used GXV, continue to use GXV. If you use a different code generator, you will obtain unusable hybrid results.

The options for your chosen code generator appear below the RunTime Arguments text field. The `-scale large` argument is one example of a run-time argument. The `clear` argument is one example of a `make` argument. See the appropriate code generator Programmer's Guide for further information on these options.

Note – The makefile contains all the property options that you used the last time you generated code. If you load-in an application with a makefile, all of the code generator options contained in the makefile are loaded into the tool. When you press the Apply button on the property sheet, the makefile is updated.

Check Use Maketool if you want to use the SPARCworks™ or ProWorks™ Maketool. Type in make or runtime arguments in the appropriate text fields.

Note – The Maketool option works only if you have installed the SPARCworks or ProWorks software and updated your path accordingly.

Click SELECT on the Apply button to effect the changes. Click SELECT on the Reset button to clear the propery sheet. (You must select Apply for the clearing to take effect.

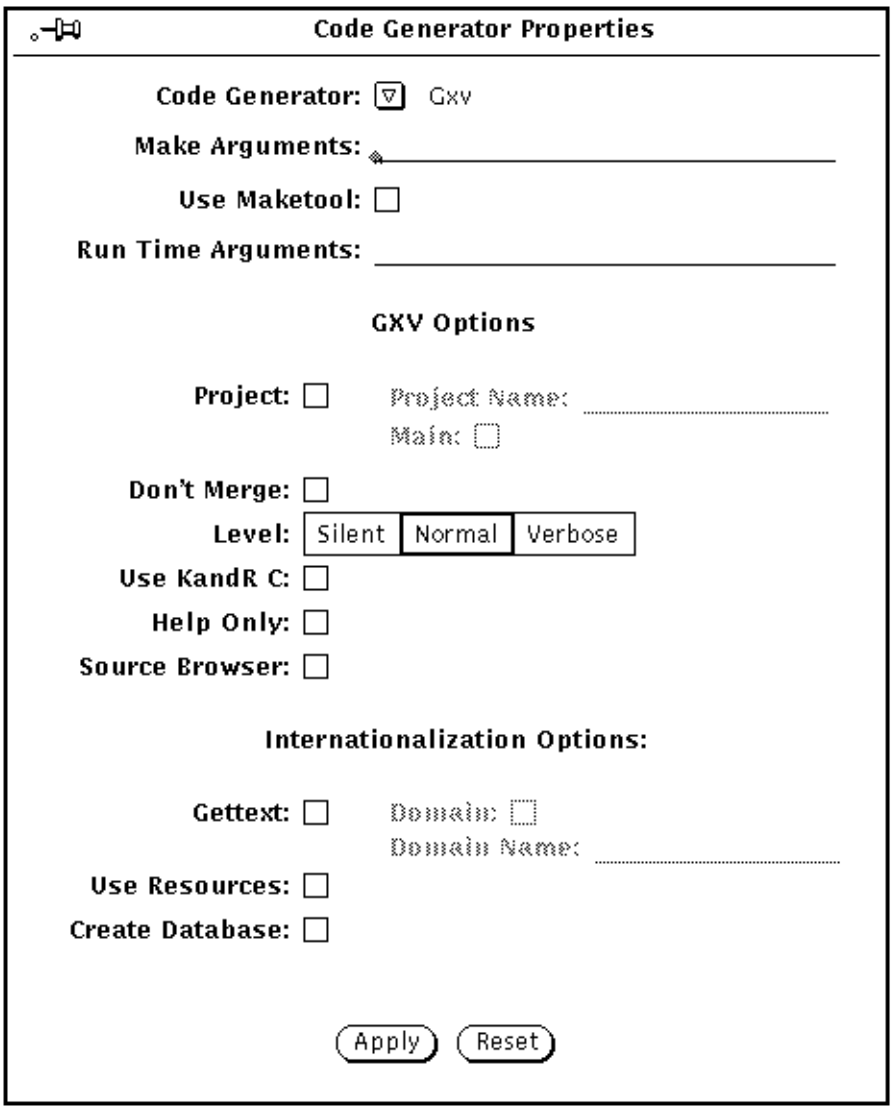


Figure 3-37 Code Generator Properties Pop-up Window

Printing Windows

If you want to capture a window and send its image to a printer, use Snapshot. See *Solaris 2.4 Introduction* for more information on Snapshot.

3

Devguide Reference 4

The glyphs in Devguide's palette represent elements you can add to a user interface. This chapter describes each UI element represented by a glyph, working from the windows to the panes and on to the controls. Each section contains:

- A brief description of a UI element, along with a picture of its glyph
- The UI element's property window
- A description of each field in the UI element's property window

Menus, which aren't represented by a glyph, are described in their own section in "Making and Editing Menus" on page 65.

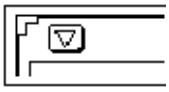
Note – Use Build mode to set UI element property values. Some properties take effect when you first apply them. Other properties take effect when you compile code or put Devguide in Test mode, such as window footers and pop-up window pushpins.

Refer to sections in Chapter 3, "Using Devguide" for details on performing these tasks:

- For placing, sizing, moving, selecting, cutting, and pasting UI elements, see "Adding Elements to a User Interface" on page 54.
- For choosing appropriate glyph and icon file formats depending on your toolkit code generator, see "Adding Glyphs and Icons to UI Elements" on page 61.

- For setting element colors, see “Choosing a Color” on page 63.
- For adding help text, see “Editing Help Text” on page 61.
- For establishing connections, see “Handling Events Using Connections” on page 81.

Base Windows



A base window is the principal window for an application. It can contain any number of panes, which display information or provide controls. Start each application you build with a base window.

Editing the Properties of a Base Window

To edit the properties of a UI base window, select it and open its property window, shown in Figure 4-1. To do this, double-click SELECT on the window or its footer (not on its header or border). See “Opening UI Element Property Windows” on page 60 for other methods of opening property windows.

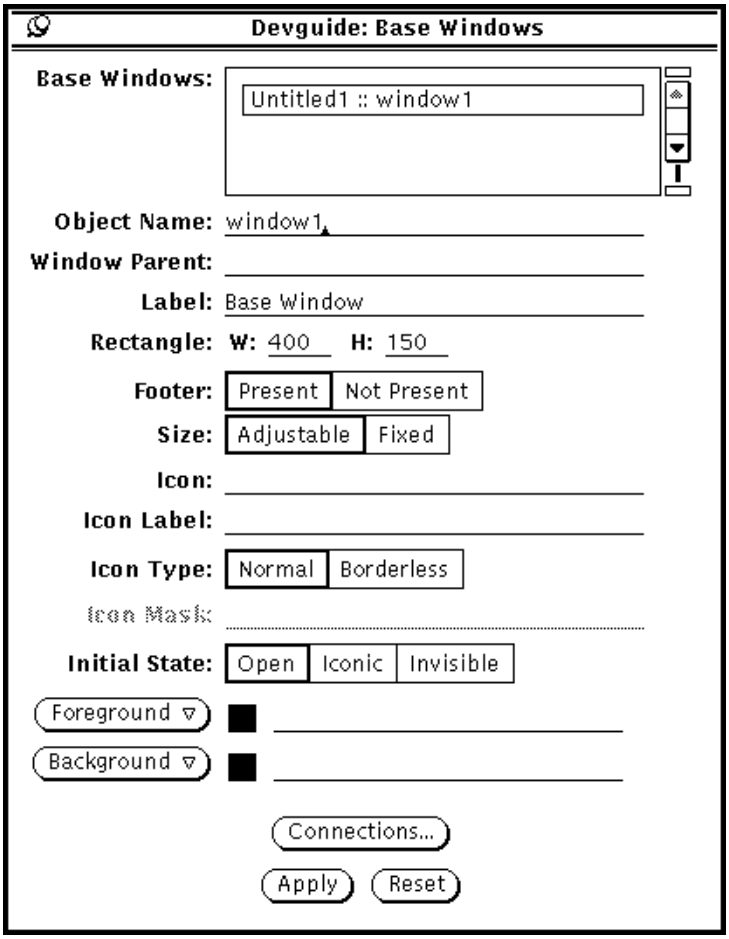


Figure 4-1 Base Windows Property Window

Base Windows Scrolling List

The scrolling list at the top of the property window lists each of the base windows in the current interface. To set the properties of a base window, click SELECT on the window's name. The name is boxed to show that it's selected. The controls in the rest of the property window change to show the current settings of the selected base window.

Object Name

Object Name establishes the name that Devguide gives the base window in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the base window.

Window Parent

The base window's parent.

Label

The Label text field sets the label that appears in the base window's header.

Rectangle

The Rectangle line contains two text fields:

- W – The base window's width in pixels
- H – The base window's height in pixels

To change one of these values, select it, then type the new value. These values update automatically when you make any changes by dragging resize corners or handles.

Footer

There are two possible footer states for base windows.

- Present – Adds a footer to the bottom of the base window
- Not Present – Removes the footer from the bottom of the window

This property takes effect when you enter Test mode or after you compile code.

Size

The Size setting determines whether the base window is resizable or not.

- Adjustable – Sets the base window so the user can resize it by dragging its resize corners

- Fixed – Sets the base window so the user can't resize it

If you turn off the resize corners, you won't see them disappear until you leave Devguide and generate and compile source code for the window.

Icon

Icon contains the file name of the icon you want to attach to the UI base window. Note that you must create, name, and save your own icon before you compile the user interface code so Devguide can attach the icon when you compile the code.

Icon Label

Icon Label contains the label you want to attach to the UI base window's icon. Icon Label is optional.

Icon Type

The icon you choose to display for the base window can have a border, or be borderless.

- Normal – The base window's icon appears on the workspace as a bordered icon; that is, it is a full opaque rectangle.
- Borderless – The icon appears on the workspace without a rectangular border. To create a borderless icon, you must create and specify an icon mask to accompany your icon. This mask is essentially a shadow of your icon image. When you select the Borderless Icon Type, the Icon Mask field is enabled.

Icon Mask

Fill in Icon Mask with the name of the file containing your icon mask. The icon mask is an icon that is a silhouette of your standard icon. Any black pixels in the icon mask block background screen from showing through the standard icon. Any white pixels allow the background to show through.

Initial State

There are three possible initial states for a base window.

- Open – The window appears as an open window when you start the program.
- Iconic – The window appears as an icon when you start the program.
- Invisible – The window is invisible when you start the program.

The user interface must be set so that at least one window is initially open or iconic.

Foreground

Press MENU on the Foreground menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Foreground sets the foreground color of the base window.

Background

Press MENU on the Background menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Background sets the background color of the base window.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the base window and other existing UI elements using the Connections Manager.

Apply

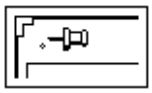
Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Pop-up Windows



A pop-up window, like a base window, defines a section of the workspace where an application displays information, performs functions, and offers controls to the user. A pop-up window is auxiliary to the base window, and never appears as a program's primary window. Pop-up windows typically appear at the user's request to display more information or controls.

Editing the Properties of a Pop-up Window

To edit the properties of a pop-up window, select it and open its property window, shown in Figure 4-2. To do this, double-click SELECT on the window or its footer (not on its header or border). See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

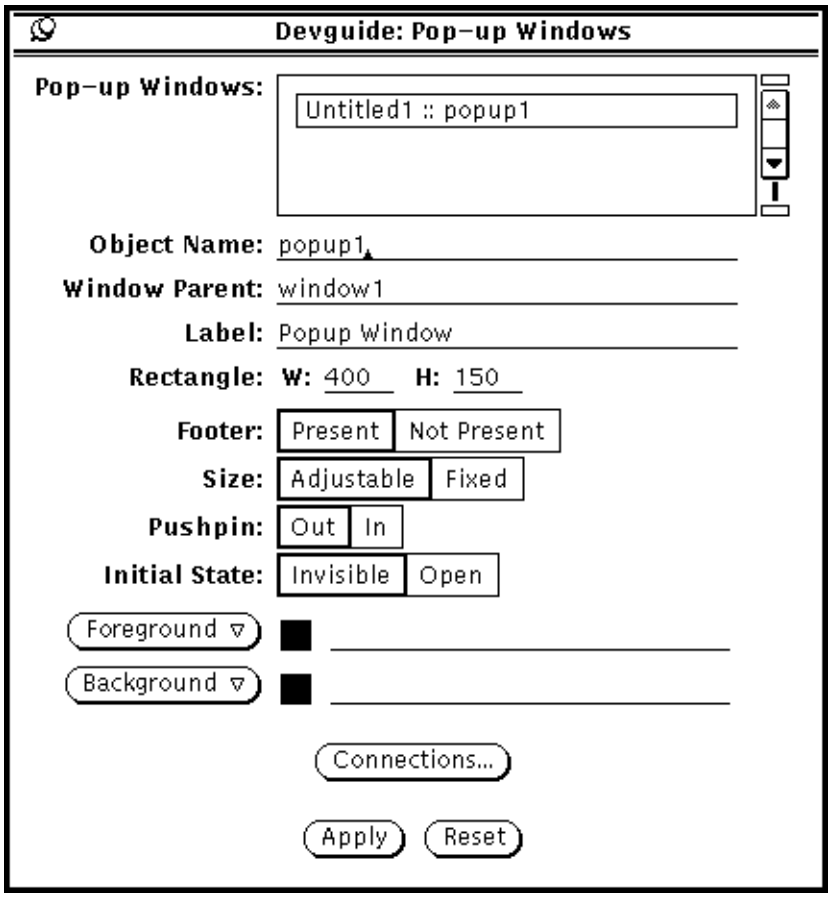


Figure 4-2 Pop-up Windows Property Window

Pop-up Windows Scrolling List

The scrolling list at the top of the property window lists each of the pop-up windows in the current interface. To set the properties of a pop-up window, click SELECT on the window's name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected pop-up window.

Object Name

Object Name establishes the name that Devguide gives the pop-up window in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the pop-up window.

Window Parent

The pop-up window's parent.

Label

Label sets the label that appears in the pop-up window's header.

Rectangle

The rectangle line contains two text fields:

- W – The pop-up window's width in pixels
- H – The pop-up window's height in pixels

To change one of these values, select it, then type the new value. These values update automatically when you make any changes by dragging resize corners or handles.

Footer

There are two possible footer states for pop-up windows.

- Present – Adds a footer to the bottom of the pop-up window
- Not Present – Removes the footer from the bottom of the window

This property takes effect when you enter Test mode or after you compile code.

Size

Size determines whether the pop-up window is resizable or not.

- Adjustable – Sets the pop-up window so the user can resize it by dragging its resize corners

- Fixed – Sets the pop-up window so the user can't resize it

If you turn off the resize corners, you won't see them disappear until you leave Devguide and generate and compile source code for the window.

Pushpin

Pushpin determines whether or not the pop-up window can remain on the workspace indefinitely.

- Out – Sets the pop-up window's pushpin in the out position when the pop-up window first appears. The window disappears when Apply is selected. Clicking SELECT on the pushpin toggles it to the In position.
- In – Sets the pushpin in the in position so the pop-up window will remain on the workspace until the user clicks SELECT on the pushpin.

Initial State

There are two possible initial states for a pop-up window.

- Invisible – The pop-up window is invisible when you start the program.
- Open – The pop-up window appears as an open window when you start the program.

Foreground

Press MENU on the Foreground menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Foreground sets the foreground color of the pop-up window.

Background

Press MENU on the Background menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Background sets the background color of the pop-up window.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the pop-up window and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Dismissing a Pop-up Window

To dismiss a UI pop-up window from the workspace to free up space, first select the window, then choose Dismiss from its Window menu or simply click SELECT on its pushpin. The window disappears from the workspace, but it still remains a part of the user interface you're designing.

Recalling a Pop-up Window

To recall a dismissed UI pop-up window, open the Pop-up Windows property window by choosing Pop-up Windows... from the Properties menu. Select the name of the pop-up window from the Pop-up Windows list; the UI pop-up window appears once again on the workspace. You can also recall all dismissed pop-up windows at one time by choosing Dismissed Windows from the View button on the Devguide window.

Tying a Pop-up Window to Other Elements

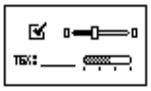
Note that when you create a pop-up window with Devguide, it is unattached to other user interface elements such as buttons. If you want the pop-up window to appear when an external event occurs, you must do one of the following:

- Use connections if your postprocessor provides appropriate built-in actions. See “Handling Events Using Connections” on page 81 for details.
- Add your own code to the code generated by the postprocessor to tie the pop-up window to actions in the user interface.

Setting a Pop-up Window’s Appearance Location

Devguide does not control the pop-up window’s location when it first appears. To set the location, you must specify it in your own custom code.

Control Areas



A control area is a region of a window where the program displays controls such as buttons, settings, and sliders. You must place a control area within a base window or pop-up window before you can start to place controls into the user interface.

Editing the Properties of a Control Area

To edit the properties of a UI control area, select it and open its property window, shown in Figure 4-3. To do this, double-click SELECT on the control area. See “Opening UI Element Property Windows” on page 60 for other methods of opening property windows.

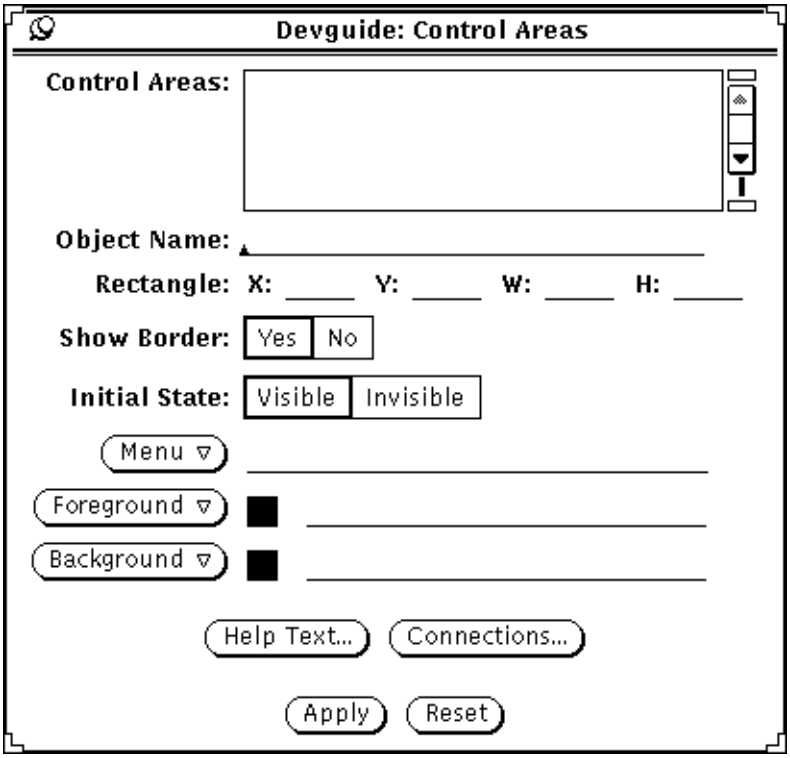


Figure 4-3 Control Areas Property Window

Control Areas Scrolling List

The scrolling list at the top of the property windows lists each of the control areas in the current interface. To set the properties of a control area, click SELECT on the control area's name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected control area.

Object Name

Object Name establishes the name that Devguide gives the control area in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the control area.

Rectangle

The Rectangle entry contains four text fields that are used to describe a control area.

- X – The distance in pixels of the upper-left corner of the pane or control area from the left edge of the parent (containing) window
- Y – The distance in pixels of the upper-left corner of the pane or control area from the top of the parent (containing) window
- W – The control area's width in pixels
- H – The control area's height in pixels

These values update automatically when you make any changes by dragging resize corners or handles.

Show Border

The Show Border setting controls the border that appears around a control area.

- Yes – Turns on the border around the control area
- No – Turns off the border

If you turn off the border, it won't disappear while you're in Build mode, but it will disappear when you enter Test mode.

Initial State

There are two possible initial states for a control area when its parent window first appears.

- Visible – The control area appears as an open window when its parent window first appears.
- Invisible – The control area is invisible when its parent window first appears.

Menu

Press MENU on the Menu button to display the New... and Names items. Names has an attached submenu. Choose New... to open the Menu Editor Window and create a new menu. To attach an existing menu to the control area, choose the menu's name from the Names submenu, or enter the name on the line to the right of the Menu button.

Foreground

Press MENU on the Foreground menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Foreground sets the foreground color of the control area.

Background

Press MENU on the Background menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Background sets the background color of the control area.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the control area.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the control area and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Aligning Controls in a Control Area

After you add controls to a control area, you can arrange the controls any of three methods:

- Choose any of Devguide's Arrange menu items: Group and Ungroup, the Align... pop-up, or the Grid and Grid Snap submenus.
- Create groups, and arrange the group elements using the Groups window.
- Use the Align and Distribute submenus of a control area's pop-up menu. (The pop-up menu also allows you to group and ungroup.)

Canvas Panes

A canvas pane is a region of a window where the program displays graphics and where users can draw their own graphics or edit existing graphics. Each canvas pane has a boundary surrounding it. Because the canvas pane often contains a larger image (called a canvas) than it can display at once, you can add a vertical scrollbar and a horizontal scrollbar so the user can scroll through the canvas.

Editing the Properties of a Canvas Pane

To edit the properties of a UI canvas pane, select it and open its property window, shown in Figure 4-4. To do this, double-click SELECT on the canvas pane. See “Opening UI Element Property Windows” on page 60 for other methods of opening property windows.

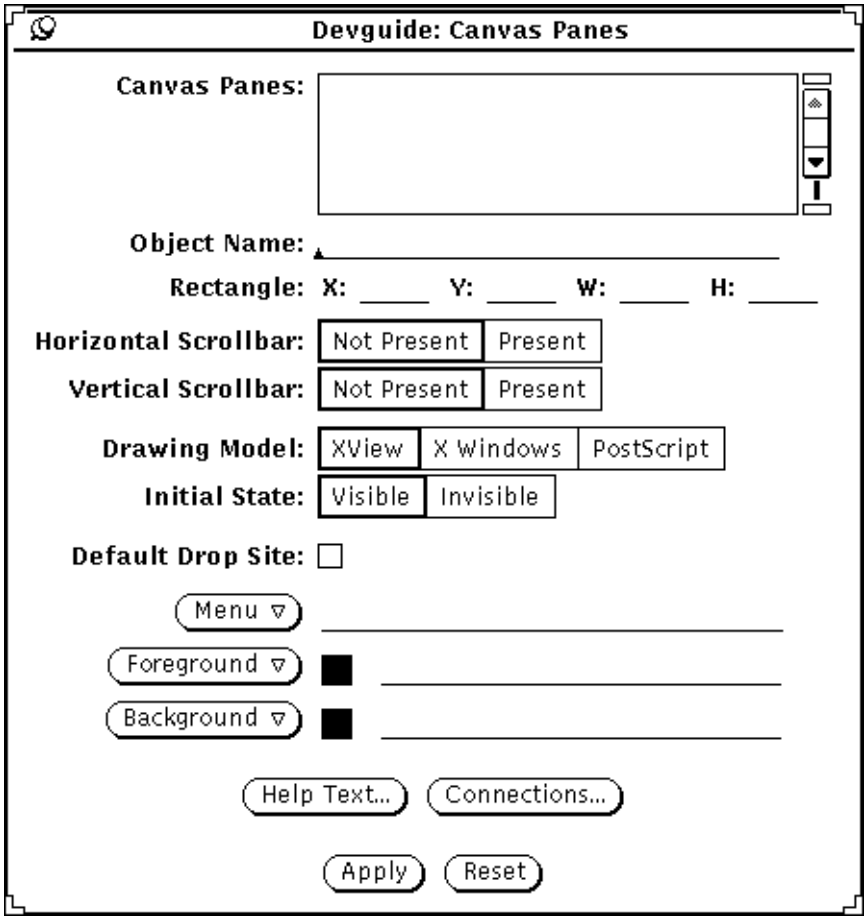


Figure 4-4 Canvas Panes Property Window

Canvas Panes Scrolling List

The scrolling list at the top of the property windows lists each of the canvas panes in the current interface. To set the properties of a canvas pane, click SELECT on the pane's name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected canvas pane.

Object Name

Object Name establishes the name that Devguide gives the canvas pane in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the canvas pane.

Rectangle

The Rectangle entry contains four text fields that are used to describe a canvas pane.

- X – The distance in pixels of the upper-left corner of the canvas pane from the left edge of the parent (containing) window
- Y – The distance in pixels of the upper-left corner of the canvas pane from the top of the parent (containing) window
- W – The canvas pane's width in pixels
- H – The canvas pane's height in pixels

These values update automatically when you make any changes by dragging resize corners or handles.

ScrollBars

The Horizontal Scrollbar setting controls the appearance of a horizontal scrollbar across the bottom of the canvas pane, and the Vertical Scrollbar setting controls the appearance of a vertical scrollbar along the right edge of the canvas pane.

- Not Present – Turns off the scrollbar
- Present – Turns on the scrollbar

Once a scrollbar is added to the canvas pane, it allows the canvas pane to display an image (or canvas) larger than the canvas pane.

Drawing Model

If you want to assign a repaint handler to the canvas pane using connections, you must select the drawing model here. The model you choose should be supported by the toolkit postprocessor that you intend to use:

- GXV – Choose any of the three available drawing models.
- GOLIT – Choose XWindows only.

Initial State

There are two possible initial states for a canvas pane when its parent window first appears.

- Visible – The canvas pane appears when its parent window first appears.
- Invisible – The canvas pane is invisible when its parent window first appears.

Default Drop Site

Default Drop Site is a check box setting. Check Default Drop Site if you want the canvas pane to be the default drop site if a file or data is dropped on your application's icon. The drop is then forwarded to this drop site. You can have only one default drop site for your application. It must reside within the application's main base window.

Default Drop Site is enabled only if you have designated the canvas as a legal drop site. To do this, you must define a connection with the canvas pane as both source and target. Choose DroppedUpon from the When menu and CallFunction from the Action menu. The function whose name you supply will be called when a file or data is dropped on the canvas pane. See "Handling Events Using Connections" on page 81 for details.

Menu

Press MENU on the Menu button to display the New... and Names items. Names has an attached submenu. Choose New... to open the Menu Editor Window and create a new menu. To attach an existing menu to the canvas pane, choose the menu's name from the Names submenu, or enter the name on the line to the right of the Menu button.

Foreground

Press MENU on the Foreground menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Foreground sets the foreground color of the canvas pane.

Background

Press MENU on the Background menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Background sets the background color of the canvas pane.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the canvas pane.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the canvas pane and other existing UI elements using the Connections Manager.

Apply

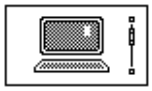
Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Term Panes



A term pane is a region of a window where the program presents a UNIX shell to the user. It displays the workstation's prompt where the user can enter UNIX commands and work directly with the operating system. As the text of the user's session with the shell grows in length, it scrolls up and off the top edge of the term pane.

Editing the Properties of a Term Pane

To edit the properties of a UI term pane, select it and open its property window, shown in Figure 4-5. To do this, double-click SELECT on the term pane. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

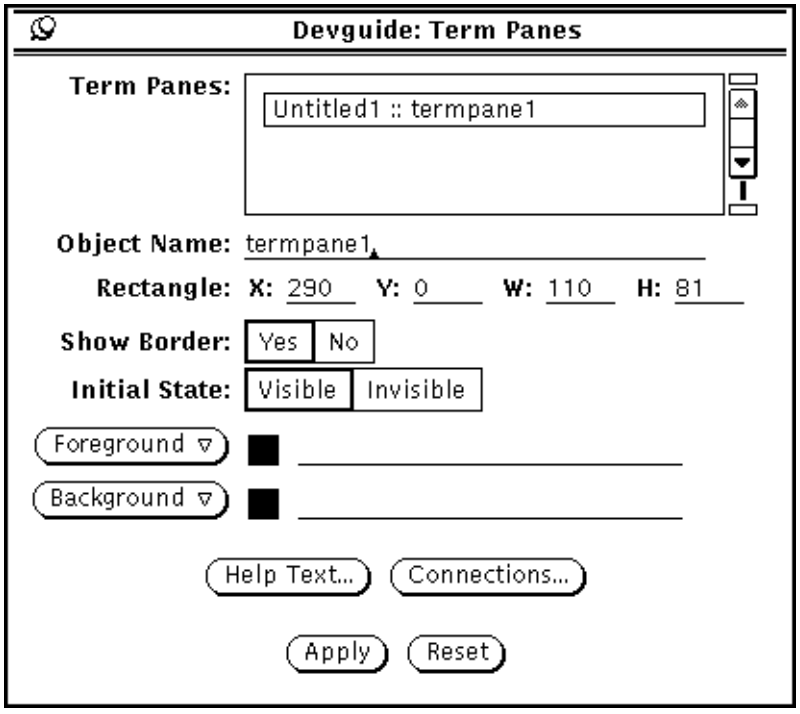


Figure 4-5 Term Panes Property Window

Term Panes Scrolling List

The scrolling list at the top of the property windows lists each of the term panes in the current interface. To set the properties of a term pane, click SELECT on the pane's name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected term pane.

Object Name

Object Name establishes the name that Devguide gives the term pane in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the term pane.

Rectangle

The Rectangle entry contains four text fields that are used to describe a term pane.

- X – The distance in pixels of the upper-left corner of the term pane from the left edge of the parent (containing) window
- Y – The distance in pixels of the upper-left corner of the term pane from the top of the parent (containing) window
- W – The term pane's width in pixels
- H – The term pane's height in pixels

These values update automatically when you make any changes by dragging resize corners or handles.

Show Border

The Show Border setting controls the border that appears around a term pane.

- Yes – Turns on the border around the term pane
- No – Turns off the border

If you turn off the border, it won't disappear while you're in Build mode, but it will disappear when you enter Test mode.

Initial State

There are two possible initial states for a term pane when its parent window first appears.

- Visible – The term pane appears when its parent window first appears.
- Invisible – The term pane is invisible when its parent window first appears.

Foreground

Press MENU on the Foreground menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Foreground sets the foreground color of the term pane.

Background

Press MENU on the Background menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Background sets the background color of the term pane.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the term pane.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the term pane and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

4

Text Panes

A text pane is a region of a window where the program displays text and where the user can edit text and enter new text. Each text pane has a boundary surrounding it. Because the text pane often contains more text than it can display at once, it has a vertical scrollbar on its right side that the user can use to scroll up and down through the text.

Editing the Properties of a Text Pane

To edit the properties of a UI text pane, select it and open its property window, shown in Figure 4-6. To do this, double-click SELECT on the text pane. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

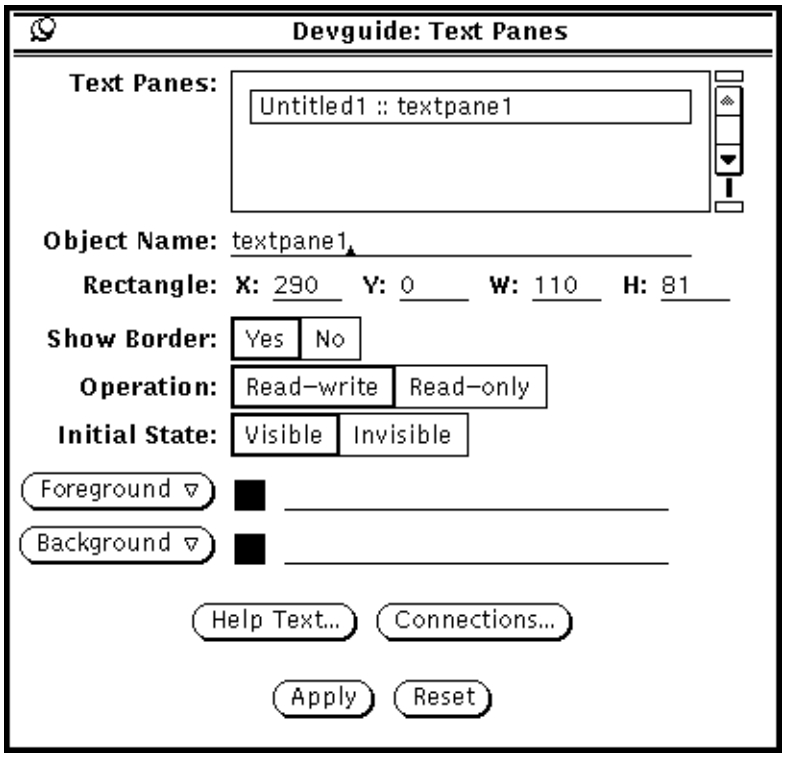


Figure 4-6 Text Panes Property Window

Text Panes Scrolling List

The scrolling list at the top of the property windows lists each of the text panes in the current interface. To set the properties of a text pane, click SELECT on the pane's name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected text pane.

Object Name

Object Name establishes the name that Devguide gives the text pane in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the text pane.

Rectangle

The Rectangle entry contains four text fields that are used to describe a text pane.

- X – The distance in pixels of the upper-left corner of the text pane from the left edge of the parent (containing) window
- Y – The distance in pixels of the upper-left corner of the text pane from the top of the parent (containing) window
- W – The text pane's width in pixels
- H – The text pane's height in pixels

These values update automatically when you make any changes by dragging resize corners or handles.

Show Border

The Show Border setting controls the border that appears around a text pane.

- Yes – Turns on the border around the text pane
- No – Turns off the border

If you turn off the border, it won't disappear while you're in Build mode, but it will disappear when you enter Test mode.

Operation

The Operation setting determines whether text panes can accept input from the user, in addition to displaying text and numeric values.

- Read-write – Allows the user to alter text presented in the text pane
- Read-only – Locks the text so the user can't change it

Initial State

There are two possible initial states for a text pane when its parent window first appears.

- Visible – The text pane appears when its parent window first appears.
- Invisible – The text pane is invisible when its parent window first appears.

Foreground

Press MENU on the Foreground menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Foreground sets the foreground color of the text pane.

Background

Press MENU on the Background menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Background sets the background color of the text pane.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the text pane.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the text pane and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

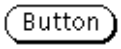
Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

4

Buttons



A button is a control within a control area that initiates an action or presents an attached menu when the user selects the button. When the button has an attached menu, it's called a menu button.

Editing the Properties of a Button

To edit the properties of a button, double-click SELECT on it to open its property window, shown in Figure 4-7. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

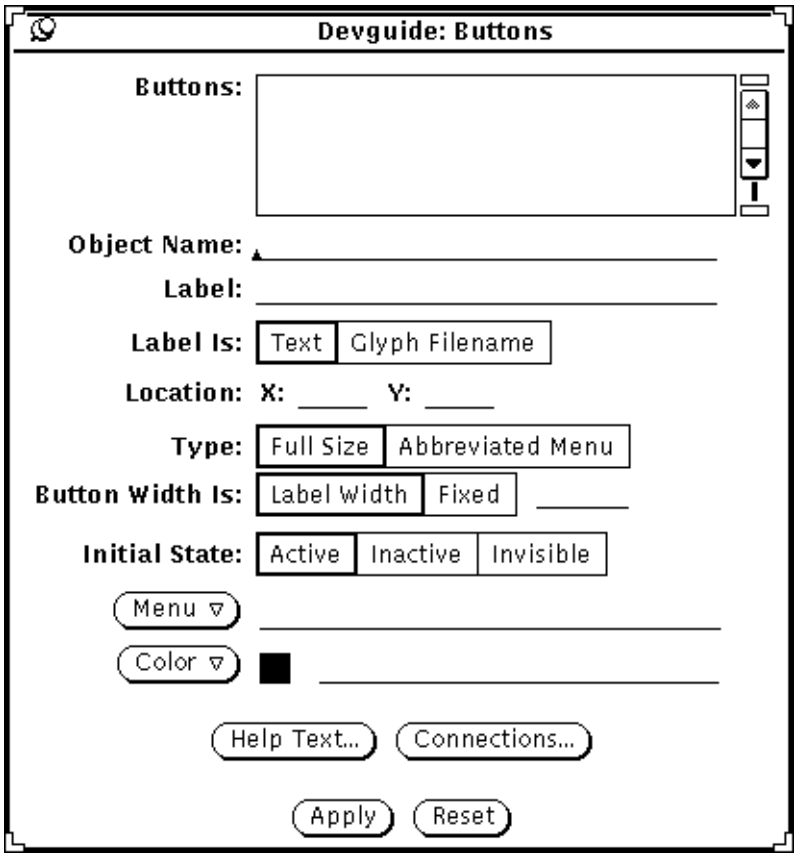


Figure 4-7 Buttons Property Window

Buttons Scrolling List

The scrolling list at the top of the property windows lists each of the buttons in the current interface. To set the properties of a button, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected button.

Object Name

Object Name establishes the name that Devguide gives the button in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the button.

Label

Contains the label (or glyph) that appears in the button. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the button is labeled with text or with a glyph.

- Text – Inserts the text entered in the Label text field as the button label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the button label

Location

The Location line contains two text fields:

- X – The distance in pixels of the upper-left corner of the button from the left edge of the control area
- Y – The distance in pixels of the upper-left corner of the button from the top of the control area

Type

The Type setting controls the appearance of the button when you have an attached menu.

- Full Sized – The button appears as a standard button with a button label within the button boundaries.
- Abbreviated Menu – The button appears as a small down-pointing arrow with the current menu choice appearing next to it.

Button Width Is

The Button Width Is setting sets the width of the button.

- Label Width – The button width changes to fit the length of the text in the button's label.
- Fixed – The button width remains fixed regardless of the button label.

The text field on the right side of the Button Width Is setting determines the length in pixels of the fixed width button.

Initial State

There are three possible initial states for a button when its parent control area first appears.

- Active – The button appears and is active when its parent control area first appears.
- Inactive – The button appears, but is inactive when its parent control area first appears.
- Invisible – The button is invisible when its parent control area first appears.

Menu

Press MENU on the Menu button to display the New... and Names items. Names has an attached submenu. Choose New... to open the Menu Editor Window and create a new menu. To attach an existing menu to the button, choose the menu's name from the Names submenu, or enter the name on the line to the right of the Menu button.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the button's color.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the button.

Note – Help text disappears if the foreground and background colors are set to any color close to the default color. This is a general color problem due to the fact that color is inherited from the base window. To avoid this problem, don't set the color.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the button and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Messages

"Message" A message is a control that presents a message to the user in read-only text or as a glyph that the user can't alter. You can use messages to add text and images throughout a control area.

Editing the Properties of a Message

To edit the properties of a UI message, you double-click SELECT on it to open its property window, shown in Figure 4-8. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

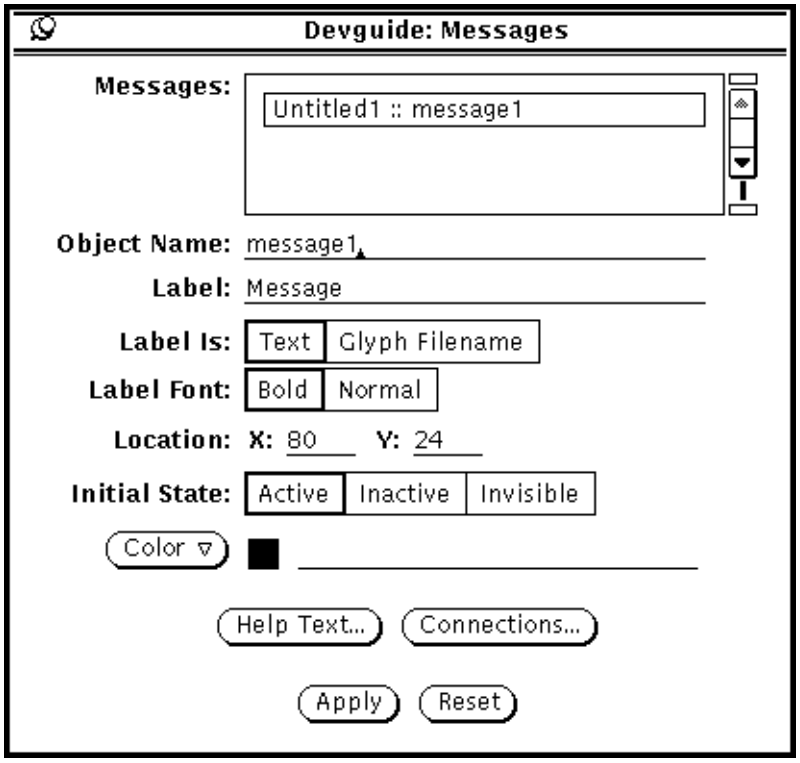


Figure 4-8 Messages Property Window

Messages Scrolling List

The scrolling list at the top of the property windows lists each of the messages in the current interface. To set the properties of a message, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected message.

Object Name

Object Name establishes the name that Devguide gives the message in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the message.

Label

Contains the label (or glyph) that appears in the message. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the message is labeled with text or with a glyph.

- Text – Inserts the text entered in the Label text field as the message label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the message label

Label Font

The Label Font setting controls the appearance of the text label of the message UI element.

- Bold – The text appears in boldface.
- Normal – The text appears unemphasized.

Location

The Location line contains two text fields:

- X – The distance in pixels of the upper-left corner of the message from the left edge of the control area
- Y – The distance in pixels of the upper-left corner of the message from the top of the control area

Initial State

There are three possible initial states for a message when its parent control area first appears.

- Active – The message appears and is active when its parent control area first appears.
- Inactive – The message appears, but is inactive when its parent control area first appears.

- Invisible – The message is invisible when its parent control area first appears.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the message's color.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the message.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the message and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first

selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Settings

A setting is a control that offers on or off choices for a collection of related options. There are four types of settings: exclusive, nonexclusive, check boxes, and stacks. Stacks, exclusive settings, and check boxes have glyphs in Devguide's palette. You can use any setting glyph to create any one of these types of settings.

Editing the Properties of a Setting

To edit the properties of a UI setting, select it and open its property window, shown in Figure 4-9. To do this, double-click SELECT on the setting. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

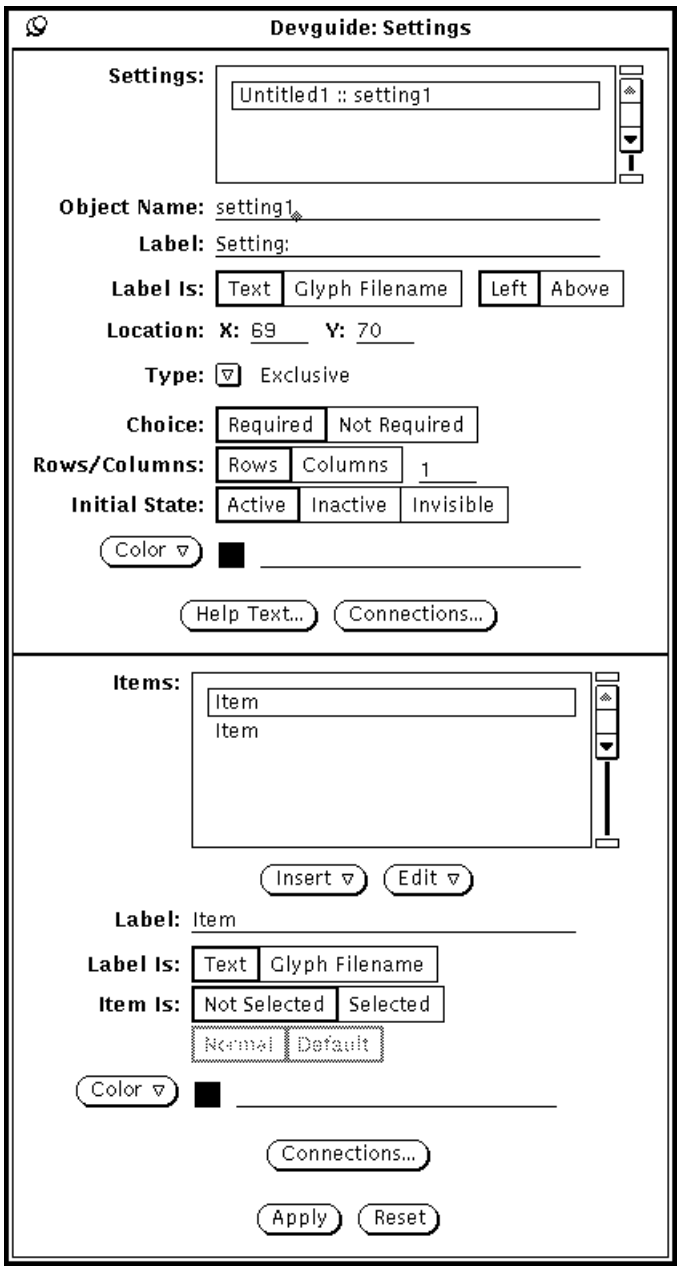


Figure 4-9 Settings Property Window

Settings Scrolling List

The scrolling list at the top of the property windows lists each of the settings in the current interface. To set the properties of a setting, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected setting.

Object Name

Object Name establishes the name that Devguide gives the setting in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the setting.

Label

Contains the label (or glyph) that appears in the setting. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the setting is labeled with text or with a glyph, and determines the label's location.

- Text – Inserts the text entered in the Label text field as the setting label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the setting label
- Left – Places the label to the left of the setting
- Above – Places the label above the setting

Location

The Location line contains two text fields:

- X – The distance in pixels of the upper-left corner of the setting from the left edge of the control area
- Y – The distance in pixels of the upper-left corner of the setting from the top of the control area

Type

Offers an abbreviated menu button that allows you to choose the type of a setting.

- Exclusive – Allows the user to turn on only one of its options at a time. When the user chooses an option, it turns on, and all other options in the setting automatically turn off.
- Nonexclusive – Allows the user to turn on each option within the setting or menu without affecting any of the other options. When the user chooses an option, it turns on if it is off, and off if it is on. (It toggles.) All other options in the setting are unaffected by the selection. Many options can be on at the same time.
- Check Box – Works the same way a nonexclusive setting works. It differs in appearance: each option label appears to the right of a check box. When the option is turned on, a check appears in the check box. When turned off, the check box is empty.
- Stack – Works much the same way an exclusive setting works. It differs in that it doesn't show all the options at one time, but only shows the label, an abbreviated button, and the option that is currently turned on displayed to the right of the button. To see all the options, the user clicks on the button with MENU to display the options in a vertical list and choose one of them. This is the same as an abbreviated menu button.

Choice

Exclusive settings allow you the flexibility of requiring or not requiring the user to make a choice.

- Required – The user must select one of the choices available.
- Not Required – The user needn't select any choice available.

Rows/Columns

Arranges the setting items in row or column format.

- Rows – Displays the items in a series of rows
- Columns – Displays the items in a series of columns

Enter the number of rows or columns you want in the text field to the right of the setting.

Initial State

There are three possible initial states for a setting when its parent control area first appears.

- Active – The setting appears and is active when its parent control area first appears.
- Inactive – The setting appears, but is inactive when its parent control area first appears.
- Invisible – The setting is invisible when its parent control area first appears.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the setting's color.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the setting.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the setting and other existing UI elements using the Connections Manager.

The Items Scrolling List

The scrolling list below the Connections... button is the Items scrolling list; it contains the items available in the setting. When you first create a setting, there are two items in the list labeled "Item." To change an item, select it. A box appears around it to show that it's selected, and its label appears in the Label

text field just below the scrolling list. To change the item's label, select the Label text field and then use the keyboard to change its contents. Click on the Apply button or any other item in the scrolling list to apply your label change.

Insert Button

The Insert button below the Items scrolling list allows you to insert a new item at any location in the scrolling list. Select one of the items to insert a new item where you want it:

- Before – Inserts a new item before the selected item
- After – Inserts a new item after the selected item
- Top – Inserts a new item at the top of the scrolling list
- Bottom – Inserts a new item at the bottom of the scrolling list

Edit Button

The Edit button below the Items scrolling list offers options that cut, copy, and paste items within the scrolling list. Use these functions to move items from one location to another:

- Cut – Removes the selected item from the scrolling list and places it in the item clipboard.
- Copy – Puts a copy of the selected item in the item clipboard.
- Paste – Puts the contents of the clipboard in the scrolling list. The four choices in the submenu are Before, After, Top, and Bottom, which insert the clipboard contents in the same locations as the corresponding items in the Insert button menu.

Label

Contains the label (or glyph) that appears as the setting item's label. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the setting item is labeled with text or with a glyph.

- Text – Inserts the text entered in the Label text field as the item's label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the item's label

Item Is

The Item Is setting determines whether or not an item is selected when the setting first appears. For stacks, Item Is also determines whether or not a setting item is the default item. The current selection (initially the default) appears to the right of the stack's abbreviated menu button.

- Not Selected – The item is initially not selected.
- Selected – The item is initially selected.
- Normal (stacks) – The item is a normal item (not the default).
- Default (stacks) – The item is the default.

Only one item in each stack can be a default item. When you set an item as the default, any other item which had been the default becomes a normal item.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the color of the setting item you create.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the setting item and other existing UI elements using the Connections Manager.

Apply

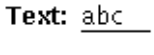
Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Text Fields



A text field is a control that requests a string of text from the user or presents a string of text for the user to edit. Text fields are typically used to request values or file names. A numeric text field is a special type of text field that accepts only numeric values. A multiline text field allows the user to display multiple lines of text in a control area. Alphanumeric and numeric text fields have glyphs on Devguide's palette. You can use any text field glyph to create any one of the three types of text fields.

Editing the Properties of a Text Field

To edit the properties of a UI text field, select it and open its property window, shown in Figure 4-10. To do this, double-click SELECT on the text field. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

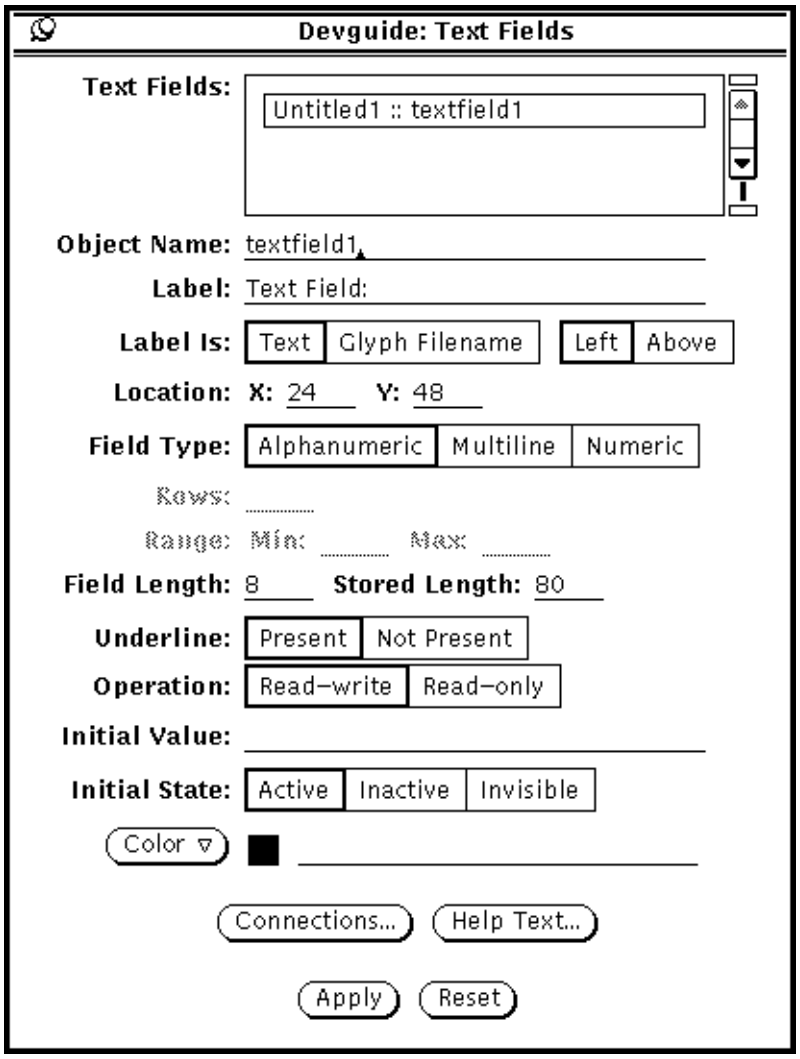


Figure 4-10 Text Fields Property Window

Text Fields Scrolling List

The scrolling list at the top of the property windows lists each of the text fields in the current interface. To set the properties of a text field, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected text field.

Object Name

Object Name establishes the name that Devguide gives the text field in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the text field.

Label

Contains the label (or glyph) that appears in the text field. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the text field is labeled with text or with a glyph, and determines the label's location.

- Text – Inserts the text entered in the Label field as the text field label
- Glyph Filename – Inserts the glyph described in the file specified in the Label field as the text field label
- Left – Places the label to the left of the text field
- Above – Places the label above the text field

Location

The Location line contains two fields:

- X – The distance in pixels of the upper-left corner of the text field from the left edge of the control area
- Y – The distance in pixels of the upper-left corner of the text field from the top of the control area

Field Type

A text field can present and accept three different types of information:

- Alphanumeric – The standard text field
- Multiline – Multiple lines of text
- Numeric – Numbers only

Rows

Set Rows to the number of rows you want to be visible in the multiline text field.

Range

If you use the Field Type setting to create a numeric text field, the Range line sets the upper and lower limits for the numeric values the text field handles.

- Min – Sets the minimum numeric value available
- Max – Sets the maximum numeric value available

Field Length

The Field Length field sets the displayed length in characters of the selected UI text field.

Stored Length

The Stored Length field sets the maximum length in characters of the UI text field storage capacity. A scrollbar appears in a multiline text field whenever the text you enter exceeds what can be displayed within the text field's borders.

Underline

Underline determines whether or not the text appearing in the text field is underlined.

- Present – The text is underlined.
- Not Present – The text is not underlined.

Operation

The Operation setting determines whether text fields can accept new input from the user in addition to displaying text and numeric values.

- Read-write – Accept input from the user.
- Read-only – Refuse input from the user.

Initial Value

The Initial Value field allows you to assign text strings, numeric values, or multiple lines of text to your text field. This value appears in the text field when it is initially displayed. A scrolling button allows you to enter as much text as you want for a multiline text field.

Initial State

There are three possible initial states for a text field when its parent control area first appears.

- Active – The text field appears and is active when its parent control area first appears.
- Inactive – The text field appears, but is inactive when its parent control area first appears.
- Invisible – The text field is invisible when its parent control area first appears.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the text field’s color.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the text field and other existing UI elements using the Connections Manager.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the text field.

Apply

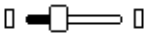
Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Sliders



A slider is a control within a control area that sets a single value from a continuous range of values. To set the value, the user drags the slider back and forth.

Editing the Properties of a Slider

To edit the properties of a slider, select it and open its property window, shown in Figure 4-11. To do this, double-click SELECT on the slider. See “Opening UI Element Property Windows” on page 60 for other methods of opening property windows.

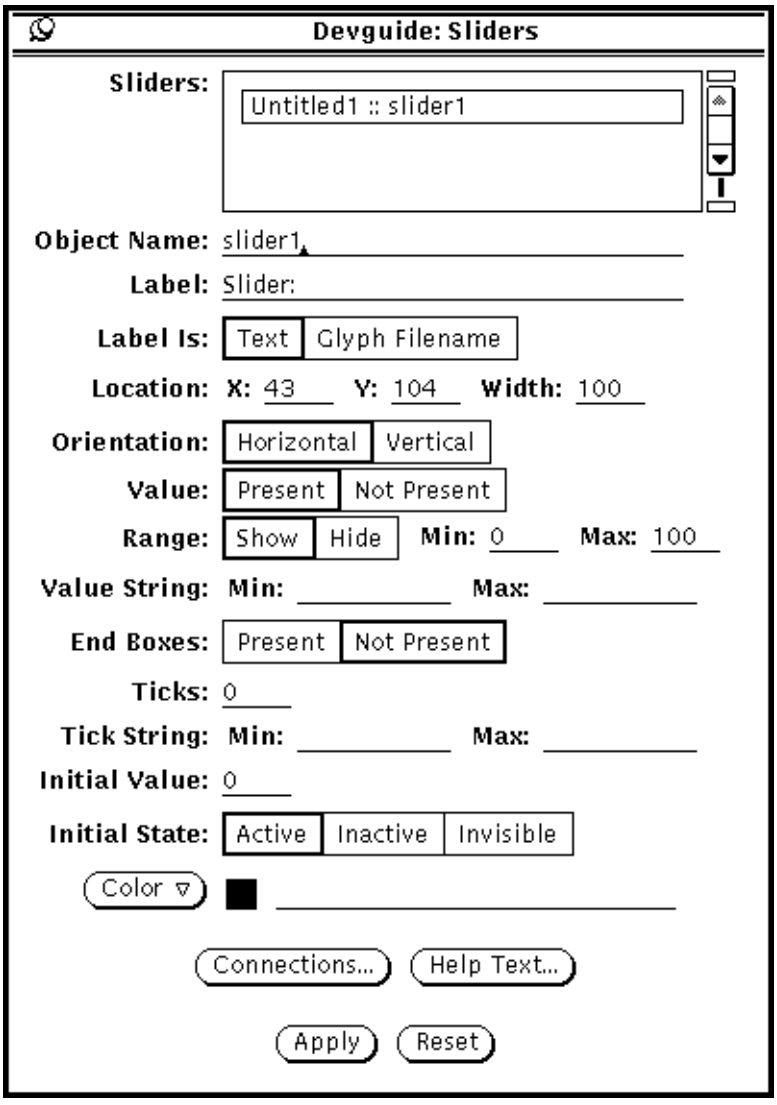


Figure 4-11 Sliders Property Window

Sliders Scrolling List

The scrolling list at the top of the property windows lists each of the sliders in the current interface. To set the properties of a slider, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected slider.

Object Name

Object Name establishes the name that Devguide gives the slider in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the slider.

Label

Contains the label (or glyph) that appears in the slider. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the slider is labeled with text or with a glyph, and determines the label's location.

- Text – Inserts the text entered in the Label text field as the slider label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the slider label
- Left – Places the label to the left of the slider
- Above – Places the label above the slider

Location

The Location line contains three fields:

- X – The distance in pixels of the upper-left corner of the slider from the left edge of the control area
- Y – The distance in pixels of the upper-left corner of the slider from the top of the control area

- Width – The width of the slider in pixels

Orientation

The Orientation setting controls the orientation of the slider.

- Horizontal – The slider runs left and right.
- Vertical – The slider runs up and down.

Value

The Value setting controls the presence of the slider's value at the left or top end of the slider.

- Present – The current value appears.
- Not Present – The current value doesn't appear.

Range

The Range line controls the range of the slider.

- Show – The range appears, with the minimum value available displayed on the left or low end and the maximum value available displayed on the right or high end.
- Hide – The minimum and maximum values don't appear.
- Min – You set the minimum numeric value available here.
- Max – You set the maximum numeric value available here.

Value String

You can supply text strings to appear at either end of the slider, instead of the minimum and maximum values you have set.

- Min – Text string to represent the minimum value you have set (in Range)
- Max – Text string to represent the maximum value you have set (in Range)

End Boxes

The End Boxes setting controls the presence of end boxes (small boxes at each end of the slider).

- Present – Adds end boxes to the slider
- Not Present – Removes end boxes from the slider

Ticks

The Ticks field controls both the presence and number of tick marks along the slider. If you enter a value of 0, no tick marks are displayed. If you set a value of 2 or higher, that many tick marks appear along the slider.

Tick String

You can supply text strings that appear under the first and last tick marks if ticks are displayed.

- Min – Text string that appears under the first tick mark
- Max – Text string that appears under the last tick mark

Initial Value

The Initial Value text field allows you to set the slider to a number in the Min/Max range you specify in Range. The slider is set and the number appears to the left of the slider when the slider initially appears.

Initial State

There are three possible initial states for a slider when its parent control area first appears.

- Active – The slider appears and is active when its parent control area first appears.
- Inactive – The slider appears, but is inactive when its parent control area first appears.
- Invisible – The slider is invisible when its parent control area first appears.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the slider's color.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the slider and other existing UI elements using the Connections Manager.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the slider.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Gauges

A gauge is a read-only slider that presents a value to the user that the user can't change.

Editing the Properties of a Gauge

To edit the properties of a gauge, select it and open its property window, shown in Figure 4-12. To do this, double-click SELECT on the gauge. See "Opening UI Element Property Windows" on page 60 for other methods of opening property windows.

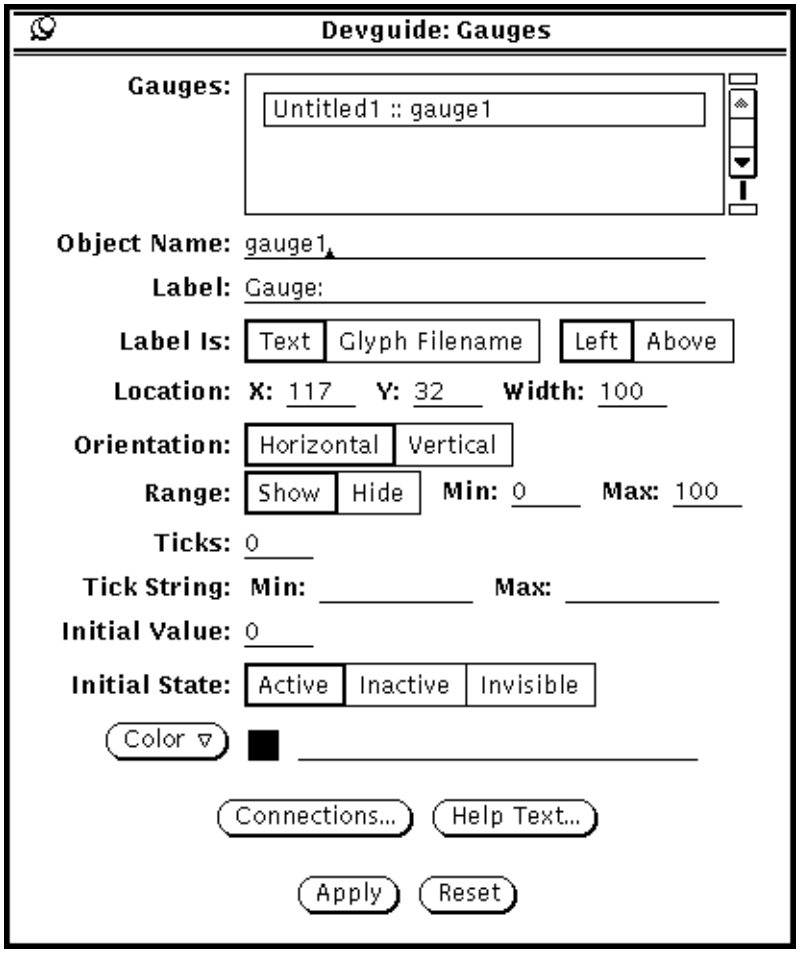


Figure 4-12 Gauges Property Window

Gauges Scrolling List

The scrolling list at the top of the property windows lists each of the gauges in the current interface. To set the properties of a gauge, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected gauge.

Object Name

Object Name establishes the name that Devguide gives the gauge in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the gauge.

Label

Contains the label (or glyph) that appears in the gauge. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the gauge is labeled with text or with a glyph, and determines the label's location.

- Text – Inserts the text entered in the Label text field as the gauge label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the gauge label
- Left – Places the label to the left of the gauge
- Above – Places the label above the gauge

Location

The Location line contains three fields:

- X – The distance in pixels of the upper-left corner of the gauge from the left edge of the control area
- Y – The distance in pixels of the upper-left corner of the gauge from the top of the control area
- Width – The width of the gauge in pixels

Orientation

The Orientation setting controls the orientation of the gauge.

- Horizontal – The gauge runs left and right.
- Vertical – The gauge runs up and down.

Range

The Range line controls the range of the gauge.

- Show – The range appears, with the minimum value available displayed on the left or low end and the maximum value available displayed on the right or high end.
- Hide – The minimum and maximum values don't appear.
- Min – You set the minimum numeric value available here.
- Max – You set the maximum numeric value available here.

Ticks

The Ticks field controls both the presence and number of tick marks along the gauge. If you enter a value of 0, no tick marks are displayed. If you set a value of 2 or higher, that many tick marks appear along the gauge.

Tick String

You can supply text strings that appear under the first and last tick marks.

- Min – Text string that appears under the first tick mark
- Max – Text string that appears under the last tick mark

Initial Value

The Initial Value text field allows you to set the gauge to a number in the Min / Max range you specify in Range. The gauge is set when it initially appears.

Initial State

There are three possible initial states for a gauge when its parent control area first appears.

- Active – The gauge appears and is active when its parent control area first appears.

- Inactive – The gauge appears, but is inactive when its parent control area first appears.
- Invisible – The gauge is invisible when its parent control area first appears.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the gauge's color.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the gauge.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the gauge and other existing UI elements using the Connections Manager.

Apply

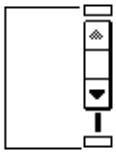
Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Scrolling Lists



A scrolling list presents a vertical list of text items and glyphs through which the user can scroll and select items using the pointer. A scrolling list is typically used whenever a set of choices varies in number from one instance to the next or when there are too many choices to present comfortably in a limited space.

Editing the Properties of a Scrolling List

To edit the properties of a UI scrolling list, select it and open its property window, shown in Figure 4-13. To do this, double-click SELECT on the scrolling list. See “Opening UI Element Property Windows” on page 60 for other methods of opening property windows.

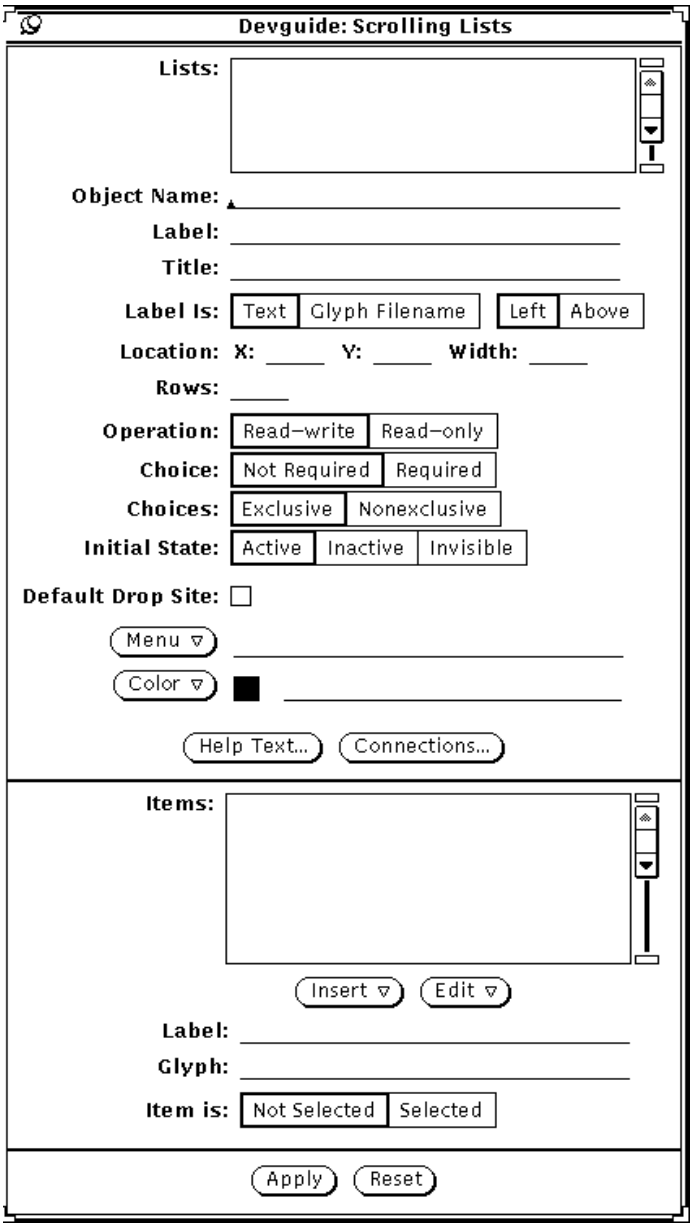


Figure 4-13 Scrolling Lists Property Window

Lists Scrolling List

The scrolling list at the top of the property windows lists each of the UI scrolling lists in the current interface. To set the properties of a UI scrolling lists, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected UI scrolling list.

Object Name

Object Name establishes the name that Devguide gives the scrolling list in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the scrolling list.

Label

Contains the label (or glyph) that appears in the scrolling list. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the scrolling list is labeled with text or with a glyph, and determines the label's location.

- Text – Inserts the text entered in the Label text field as the scrolling list label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the scrolling list label
- Left – Places the label to the left of the scrolling list
- Above – Places the label above the scrolling list

Location

The Location line contains three fields:

- X – The distance in pixels of the upper-left corner of the scrolling list from the left edge of the control area

- Y – The distance in pixels of the upper-left corner of the scrolling list from the top of the control area
- Width – The width of the scrolling list in pixels

Rows

Set Rows to the number of rows you want visible in the scrolling list.

Operation

The Operation setting determines whether the scrolling list can accept new input from the user in addition to displaying text and numeric values.

- Read-write – Accept input from the user.
- Read-only – Refuse input from the user.

Choice

A scrolling list can require that one of its items is always chosen, or it can permit the user to make no selection.

- Not Required – Allows the user the option of choosing no item
- Required – Requires one item to be chosen

Choices

A scrolling list can allow the user to choose only one of its items at a time (exclusive) or as many of its items as the user cares to choose (nonexclusive).

- Exclusive – Permits a single choice from the scrolling list
- Nonexclusive – Permits multiple choices from the scrolling list

Initial State

There are three possible initial states for a scrolling list when its parent control area first appears.

- Active – The scrolling list appears and is active when its parent control area first appears.

- Inactive – The scrolling list appears, but is inactive when its parent control area first appears.
- Invisible – The scrolling list is invisible when its parent control area first appears.

Default Drop Site

Default Drop Site is a check box setting. Check Default Drop Site if you want the scrolling list to be the default drop site if a file or data is dropped on your application's icon. The drop is then forwarded to this drop site. You can have only one default drop site for your application. It must reside within the application's main base window.

Default Drop Site is enabled only if you have designated the scrolling list as a legal drop site. To do this, you must define a connection with the scrolling list as both source and target. Choose DroppedUpon from the When menu and CallFunction from the Action menu. The function whose name you supply will be called when a file or data is dropped on the scrolling list. See "Handling Events Using Connections" on page 81 for details.

Menu

Press MENU on the Menu button to display the New... and Names items. Names has an attached submenu. Choose New... to open the Menu Editor Window and create a new menu. To attach an existing menu to the scrolling list, choose the menu's name from the Names submenu, or enter the name on the line to the right of the Menu button.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Colors sets the scrolling list's color.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the scrolling list.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the scrolling list and other existing UI elements using the Connections Manager.

Items Scrolling List

The scrolling list below the Connections... button contains the items available in the scrolling list you create. When you first create a scrolling list, the Items scrolling list is empty. You add items to the scrolling list as explained below. To change an item, select it. A box appears around it to show that it's selected, and its label appears in the Label text field just below the scrolling list. To change the item's label, select the Label text field and then use the keyboard to change its contents. Click on the Apply button or any other item in the scrolling list to apply your label change.

Insert Button

The Insert button below the Items scrolling list allows you to insert a new item at different locations in the scrolling list. The options to insert a new item where you want it are:

- Before – Inserts a new item before the selected item
- After – Inserts a new item after the selected item
- Top – Inserts a new item at the top of the scrolling list
- Bottom – Inserts a new item at the bottom of the scrolling list

Edit Button

The Edit button below the Items scrolling list offers cut, copy, and paste choices within the scrolling list. Use these to move items from one location to another:

- Cut – Removes the selected item from the scrolling list and places it in the clipboard.
- Copy – Puts a copy of the selected item in the clipboard.

- Paste – Puts the contents of the clipboard in the scrolling list. The four items in its submenu offer Before, After, Top, and Bottom, which insert the clipboard contents in the same locations as the corresponding items in the Insert button menu.

Label

Contains the text label that appears as the item's label.

Glyph

An item can have a glyph displayed to the left of its label in the scrolling list. Enter the glyph file name on this line if you want the item to have a glyph.

Item Is

The Item Is setting determines whether or not an item is selected when the scrolling list first appears.

- Not Selected – The item is initially not selected.
- Selected – The item is initially selected.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the scrolling list invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

Drop Targets

A drag and drop target is a region in a control area where drag and drop operations can be performed.

Editing the Properties of a Drop Target

To edit the properties of a UI drag and drop target, select it and open its property window, shown in Figure 4-14. To do this, double-click SELECT on the drag and drop target. See “Opening UI Element Property Windows” on page 60 for other methods of opening property windows.

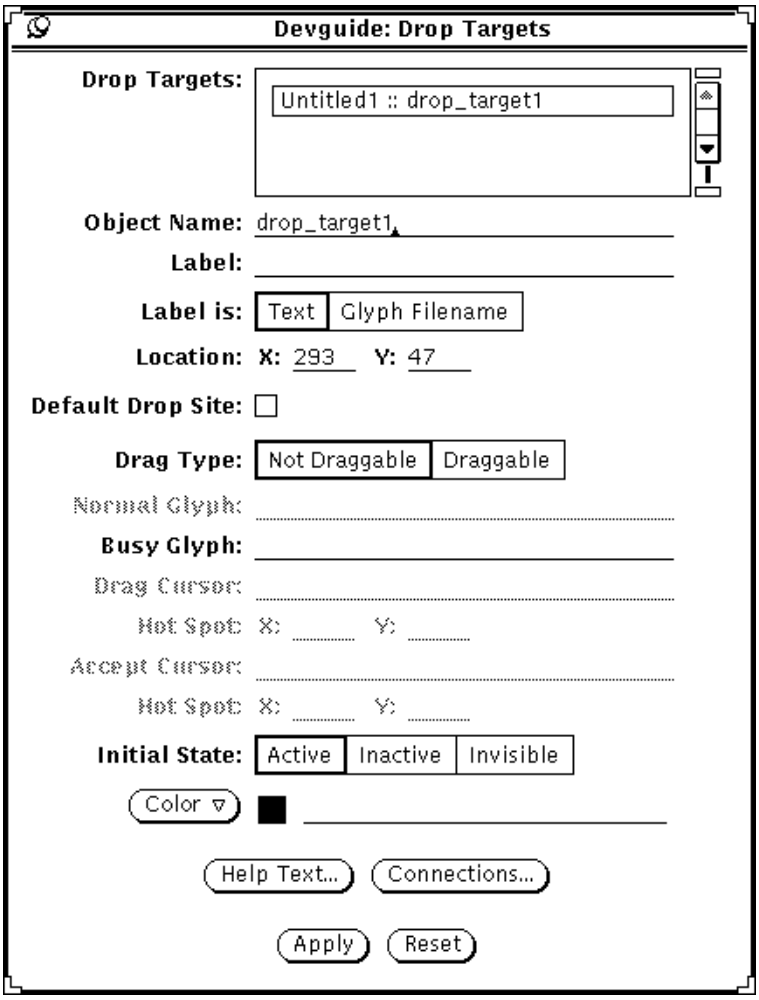


Figure 4-14 Drop Targets Property Window

Drop Targets Scrolling List

The scrolling list at the top of the property windows lists each of the UI drag and drop targets in the current interface. To set the properties of a UI drag and drop targets, click SELECT on its name. The name is boxed to show that it's selected, and the controls in the rest of the property window change to show the current settings of the selected UI drag and drop target.

Object Name

Object Name establishes the name that Devguide gives the drag and drop target in the source code that the postprocessor generates. It is used within Devguide as a unique identifier for the drag and drop target.

Label

Contains the label (or glyph) that appears to the left of the drag and drop target. The interpretation of the Label text field depends on the value of the Label Is setting.

Label Is

Label Is determines whether the drag and drop target is labeled with text or with a glyph.

- Text – Inserts the text entered in the Label text field as the drag and drop target label
- Glyph Filename – Inserts the glyph described in the file specified in the Label text field as the drag and drop target label

Location

The Location entry contains two text fields that are used to describe a drag and drop target.

- X – The distance in pixels of the upper-left corner of the drag and drop target from the left edge of the parent (containing) window
- Y – The distance in pixels of the upper-left corner of the drag and drop target from the top of the parent (containing) window

Default Drop Site

Default Drop Site is a check box setting. Check Default Drop Site if you want the drag and drop target to be the default drop site if a file or data is dropped on your application's icon. The drop is then forwarded to the drag and drop target. You can have only one default drop site for your application. It must reside within the application's main base window.

Drag Type

The Drag Type setting determines whether data can be dragged from the drag and drop target.

- Not Draggable – You can only drop onto the drag and drop target.
- Draggable – You can drag from and drop onto the drag and drop target.

Normal Glyph

You can specify a glyph file name in the Normal Glyph text field. The glyph described in the file is inserted as the drag and drop target's default glyph, and appears when the application is started. If you do not specify a file name, the drag and drop target's normal glyph looks like the UI element glyph on Devguide's palette. Normal glyph is disabled if you select Not Draggable for the drag and drop target's Drag Type.

Busy Glyph

You can specify a glyph file name in the Busy Glyph text field. The glyph described in the file is visible whenever your application is processing a drop operation. If you do not specify a filename, Devguide provides a default busy glyph.

The busy glyph is displayed when you drag data from a drag and drop target, if you select Draggable for the Drag Type. It is also displayed during a preview, if the drag and drop target is a valid drop site.

Drag Cursor

You can specify a glyph file name in the Drag Cursor text field. This file describes the cursor that is displayed when you drag data from the drag and drop target. If you do not specify a filename, a default drag cursor glyph is provided. Drag Cursor is disabled if you select Not Draggable for the drag and drop target's Drag Type.

Hot Spot

The Hot Spot entry can contains two text fields that are used to describe the drag cursor's hot spot. Hot Spot is disabled if you select Not Draggable for the drag and drop target's Drag Type, or if you have not supplied a file name in the Drag Cursor text field.

- X – The distance in pixels of the upper-left corner of the hot spot from the left edge of the box containing the drag cursor
- Y – The distance in pixels of the upper-left corner of the hot spot from the top of the box containing the drag cursor

Accept Cursor

You can specify a glyph file name in the Accept Cursor text field. The glyph described in the file is used as the accept cursor. This cursor is displayed to identify a legal drop site. The drag and drop target's drag cursor changes into the accept cursor when its hot spot is on top of a legal drop site. Accept Cursor is disabled if you select Not Draggable for the drag and drop target's Drag Type.

Hot Spot

The Hot Spot entry can contains two text fields that are used to describe the accept cursor's hot spot. Hot Spot is disabled if you select Not Draggable for the drag and drop target's Drag Type, or if you have not supplied a file name in the Accept Cursor text field.

- X – The distance in pixels of the upper-left corner of the hot spot from the left edge of the box containing the accept cursor
- Y – The distance in pixels of the upper-left corner of the hot spot from the top of the box containing the accept cursor

Initial State

There are three possible initial states for a drag and drop target when its parent control area first appears.

- Active – The drag and drop target appears and is active when its parent control area first appears.
- Inactive – The drag and drop target appears, but is inactive when its parent control area first appears.
- Invisible – The drag and drop target is invisible when its parent control area first appears.

Color

Press MENU on the Color menu button to display the Chooser... and Palette menu items. Palette has an attached submenu of colors. Press Chooser... to open the Color Chooser. Color sets the drag and drop target's color.

Help Text...

The Help Text... button opens the Help Editor window, where you can add or edit help text for the drag and drop target.

Connections...

When you click SELECT on Connections..., the Connections Manager window appears. You can establish connections between the drag and drop target and other existing UI elements using the Connections Manager.

Apply

Clicking SELECT on Apply makes your changes permanent. You cannot undo these changes with an undo command. If you want to change an element's properties after you SELECT Apply, enter new property field values and SELECT Apply again. Dismissing the property window without choosing Apply *does not* apply changes.

Note – When you have been editing one UI element, clicking on another element in the drag and drop target invokes an implicit Apply on the first element, saving all your changes. If you wish to go to another element without saving your changes, you must first click on Reset.

Reset

Clicking SELECT on Reset returns all the controls in the property window to their original settings or to the settings as they were when you last chose the Apply button. (The original settings are the ones that appeared when you first selected a UI element from the scrolling list, or the settings that appeared when you first opened the property window.) Reset only resets the properties; it does not undo them after an apply.

4

Files Provided With Devguide A

The Devguide software includes a set of files useful to Devguide users. These files are installed, along with Devguide and the code generators, on your workstation or server. You'll find the files in separate subdirectories in Devguide's home directory.

`bin` *Subdirectory*

The `$GUIDEHOME/bin` subdirectory contains the executable files for Devguide, `gmomerge`, GXV, GXV++, and GOLIT.

`demo` *Subdirectory*

The `$GUIDEHOME/demo` subdirectory contains an `xview` subdirectory, with three files:

- `rasview` – A sample program to display images from raster files
- `tree` – A fractal tree drawing program
- `psdraw` – A PostScript drawing program that uses `-lxvps`

The `$GUIDEHOME/demo/gnt` subdirectory contains the following subdirectories:

- `doc` – Documentation in PostScript format
- `man` – `gnt man` page

A

`include` *Subdirectory*

The `$GUIDEHOME/include` subdirectory contains header files that can be included in the source code for a window-based program using one of the three toolkits. These files offer different useful components of Devguide's own interface for programmers to use in their own programs.

`lib` *Subdirectory*

The `$GUIDEHOME/lib` subdirectory contains libraries of C modules that you can link into other C programs. The `libguide.a` library contains routines that are independent of the window system and the `libguidexv.a` library contains routines that depend on XView. The modules in these libraries correspond to the components offered in the `include` subdirectory.

`libgolit.a` is the runtime library for GOLIT that provides facilities for you to create user interfaces described by GOLIT-generated C code.

In addition, `lib` contains the three toolkit `.config` files. These files are used to configure the Connections Manager window. Each file lists the events, actions, and targets that are permitted by a particular toolkit for each possible source object.

`lib` also contains a `help` subdirectory that includes help files for Devguide, a `locale` subdirectory to be used for localization, and a `templates` subdirectory described below.

`lib/templates` *Subdirectory*

The `templates` subdirectory contains sample GIL files. An OPEN LOOK property window and OPEN LOOK application window created in `properties.G` and `sample.G` can be used as templates in your application to get you started. The sample application window contains File, View, and Edit menu buttons. `controls.G` contains sample UI elements you can create using Devguide.

man *Subdirectory*

The $GUIDEHOME/share/man subdirectory contains man pages for GXV, GXV++, GOLIT, and gmomerge. To see them when you use the man command, you can append $GUIDEHOME/share/man to the MANPATH variable to look in this directory when you use the man command. You can also use man -M path to view man pages that are not in the standard path. The -M option overrides MANPATH. For more information, see the man(1) man page.

src *Subdirectory*

The $GUIDEHOME/src subdirectory contains source code for the libraries included in the lib subdirectory. This source code is supplied “as is” and is not supported. You can port the source to other platforms.

doc *Subdirectory*

The $GUIDEHOME/doc subdirectory contains miscellaneous documentation about Devguide. The GIL syntax file gil_syntax.doc and project syntax file project_syntax.doc can be found in this subdirectory.

A

Porting Devguide 3.0 Applications to Devguide 3.0.1

B

To port your applications from Devguide 3.0 to Devguide 3.0.1 you need to understand the basic differences between the operating systems underlying the two different Devguide versions. Devguide 3.0 runs under the Solaris 1.0 environment, but Devguide 3.0.1 runs under Solaris 2.x.

This appendix gives you porting hints specific to Devguide applications. For more detailed information about porting applications from the SunOS 4.x environment to Solaris 2.x, see the *Solaris 1.x to Solaris 2.x Transition Guide (*prior to Solaris 2.2, there were three separate transition guides: *Solaris 2.x System Transition Guide for Users, System Transition Guide for Application Developers - SunOS 5.x,* and *SunOS 5.x System Transition Guide for Administrators).*

Note – Solaris 2.x stands for Solaris 2.0 or later; SunOS 5.x stands for SunOS 5.0 or later.

B

Background Information

By default, Devguide 3.0 produced K&R C code such as this:

```
void
my_callback(item, event)
    Panel_item            item;
    Event                 *event;
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);

    fputs("foo: my_callback\n", stderr);

    /* gxv_start_connections DO NOT EDIT THIS SECTION */

    xv_set(Foo_window1->button2, XV_SHOW, TRUE, NULL);

    /* gxv_end_connections */

}
```

Code Example B-1 K&R C code example

Devguide now produces ANSI C as the default version of C. This means that procedure names and prototypes have their parameters listed in the declaration. For example:

```
void
my_callback(Panel_item item, Event *event)
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);

    fputs("foo: my_callback\n", stderr);

    /* gxv_start_connections DO NOT EDIT THIS SECTION */

    xv_set(Foo_window1->button2, XV_SHOW, TRUE, NULL);

    /* gxv_end_connections */
}
```

Code Example B-2 ANSI C code example

Clearly, these forms are different. As a result, if your application is built with Devguide 3.0, you might experience problems merging the `_stubs.c` file(s). Also, if you use GXV and you did not use the `-a` switch parameter when you generated code, the 3.0.1 compiler will produce unexpected results when you compile your 3.0 application.

Porting Hints

The first time you use Devguide 3.0.1, generate the code and compile as you would normally. The compiler will generate errors for the callbacks that have been redefined (declared more than once).

For example, if your `old _stubs.c` file contains the following:

```
/*
 * THIS IS THE OLD STYLE FROM DEVGUIDE 3.0
 */
void
my_callback(item, event)
    Panel_itemitem;
    Event*event;
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);

    /** Some of my code here. */
    fprintf(stderr, " foo 1\n");
    fprintf(stderr, " foo 2\n");
    fprintf(stderr, " foo 3\n");

    fputs("foo: my_callback\n", stderr);

    /* gxv_start_connections DO NOT EDIT THIS SECTION */
    /* gxv_end_connections */

}
```

Code Example B-3 Old-Style `_stubs.c` file

Running Devguide 3.0.1 on the same interface produces the following callback with the same name:

```
/*
 * THIS IS THE NEWLY GENERATED CALLBACK
 */
void
my_callback(Panel_item item, Event *event)
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);

    fputs("foo: my_callback\n", stderr);

    /* gxv_start_connections DO NOT EDIT THIS SECTION */

    xv_set(Foo_window1->button2, XV_SHOW, TRUE, NULL);

    /* gxv_end_connections */

}
```

Code Example B-4 New generated callback

Notice that you now have a callback with the same name, but with an argument list in the ANSI C style. Also, the code in the body of `my_callback()` does not contain any of the changes you may have added to the 3.0 version; that is, the code has not been merged. Because the original argument list is different from the ANSI standard, Devguide treats the body of the function as a new callback.

To fix the callback, copy the missing code from the old callback (see Code Example B-3 on page 190) into the new callback. Then, delete the old callback.

To fix the callback, copy the missing code from the old callback (see Code Example B-3 on page 190) into the new callback. Then, delete the old callback.:

```
/*
 * Notify callback function for ‘button1’.
 */
void
my_callback(Panel_item item, Event *event)
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);

    /** Some of my code here. */
    fprintf(stderr, “ foo 1\n”);
    fprintf(stderr, “ foo 2\n”);
    fprintf(stderr, “ foo 3\n”);

    fputs(“foo: my_callback\n”, stderr);

    /* gxv_start_connections DO NOT EDIT THIS SECTION */

    xv_set(Foo_window1->button2, XV_SHOW, TRUE, NULL);

    /* gxv_end_connections */

}
```

Code Example B-5 Fixing the callback

System Calls

System calls differ in Solaris 1.x and Solaris 2.x. Consult the Solaris 2.x documentation to find out the changes that affect your application. For more information, see the *Solaris 1.x to Solaris 2.x Transition Guide* or one of the following:

- *Solaris 2.x System Transition Guide For Users*
- *System Transition Guide for Application Developers – SunOS 5.x*
- *SunOS 5.x System Transition Guide for Administrators*

If You Make a Mistake or Change Your Mind

If you make a mistake in generating your code, remember that the code generator backs up the previous `_stubs.c` file automatically and adds a `.BAK` extension to those files. For example, `foo_stubs.c` becomes `foo_stubs.c.BAK`. As a result, if you're not satisfied with your newly generated files, you can remove the new `_stubs.c` file and copy the backup to its original name. For example, if you're working on `foo.G`, you've generated code, and you don't like the results, you can remove your `foo_stubs.c` file and revert to the previous version as follows:

```
$ ls
foo.G               foo_stubs.c.BAK   foo_ui.h
foo.info            foo_ui.o          foo_stubs.c
foo_ui.c
$ rm foo_stubs.c
$ cp foo_stubs.c.BAK foo_stubs.c
```

B

Adding Devguide to the SPARCworks or ProWorks Environment

C

If you are using SPARCworks or ProWorks, and you want to access Devguide through the SPARCworks or ProWorks Manager, you can add Devguide to the manager as you would any tool. After you do this, the Devguide icon appears on the SPARCworks or ProWorks palette whenever you start up SPARCworks or ProWorks. You do not have to leave the SPARCworks or ProWorks context to use Devguide.

Follow these steps to add Devguide to SPARCworks or ProWorks (see *Managing Sparcworks Tools* or *Managing Proworks Tools* for further details):

1. **Select an existing icon.**
2. **Duplicate an existing SPARCworks or ProWorks tool by choosing Duplicate Tool from the SPARCworks or ProWorks Edit menu.**
   A duplicate of the icon you selected appears on the SPARCworks or ProWorks palette, with a different Icon Name.
3. **Select the new tool icon that appears on the palette.**
4. **Choose Selected Tool from the SPARCworks or ProWorks Properties menu.**
   The SPARCworks or ProWorks Manager: Tool Icon and Startup Properties popup appears.
5. **Edit the properties of the selected tool.**
   a. **Change the Tool Name field to** Devguide**.**

b. Change the Command field to

```
devguide $SUNPRO_SW_TT_ARGS $SUNPRO_SWM_GUI_ARGS
```

c. Change the Icon File field to

```
<$GUIDEHOME>/include/images/guide_32x32.icon
```

You must replace <$GUIDEHOME> with the value of $GUIDEHOME *for your system.*

d. Change the Icon Label field to Devguide**.**

e. Click SELECT on Apply to apply these changes.

Note – Selecting Apply activates your changes only during the current SPARCworks or ProWorks session. See below.

The SPARCworks or ProWorks Manager: Tool Icon and Startup Properties popup should look something like this now:

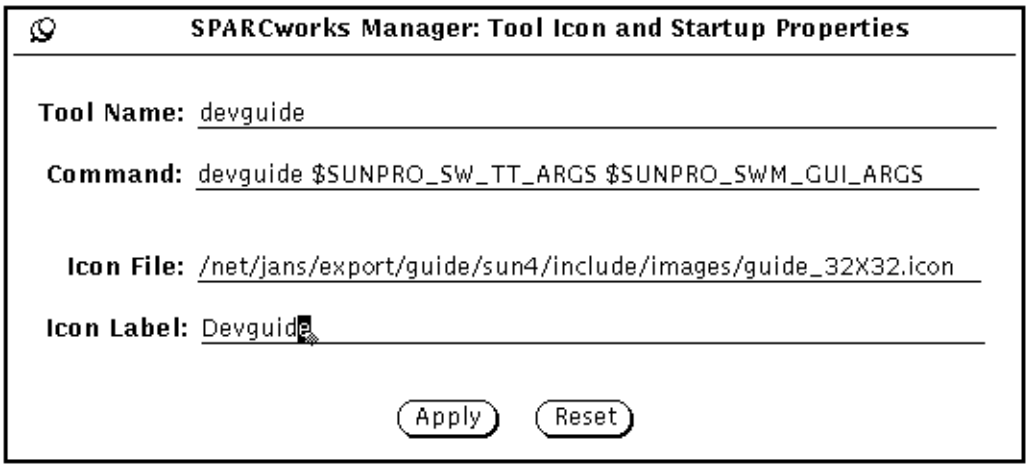


Figure C-1 Filling-in the Tool Icon and Startup Properties Pop-Up

with your system value of $GUIDEHOME replacing `/net/jans/export/guide/sun4`.

f. Select Save As... from the File menu if you want to save the changes that you applied in the previous stop. Choose a new file name so that you don't overwrite the original SPARCworks or ProWorks configuration file.
Saving the file ensures that your changes carry over into your next SPARCworks or ProWorks session. To bring-up the customized file for a new session, type:

```
% sparcworks (or proworks) yourfilename &
```

You can now open, close, hide, and show Devguide using the Session menu on the SPARCworks or ProWorks palette.

C

Devguide 3.0.1 Release Notes D

This appendix is a supplement to the following four OpenWindows Developer's Guide documents:

- *OpenWindows Developer's Guide: User's Guide* (this document)
- *OpenWindows Developer's Guide: XView Code Generator Programmer's Guide*
- *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide*
- *OpenWindows Developer's Guide: Motif Conversion Utilities Guide*

Devguide 3.0.1 is the successor to Devguide 3.0, which is the immediate successor to Devguide 1.1. There was no Devguide 2.x release. The OPEN LOOK toolkit, XView, and the OPEN LOOK Intrinsics Toolkit (OLIT), provide different levels of OPEN LOOK compliance.The following subjects are discussed in this appendix:

- Compatibility issues regarding Devguide 3.0.1
- New features of Devguide 3.0.1
- Differences between Devguide versions 3.0 and 3.0.1
- Known problems with the Devguide software
- Features not supported by the code generators

Compatibility

Devguide 3.0.1 and its associated code generators (GXV, GXV++, and GOLIT) produce code compatible with the Solaris 2.x operating environment and the OpenWindows Version 3 environment.

D

Devguide 3.0.1 can read GIL files produced by Devguide 1.1 and 3.0 and convert them into Devguide 3.0.1 format. The XView code generators GXV and GXV++ can also read Devguide 1.1 GIL files and generate OpenWindows Version 3 code.

New Devguide Features

Code Generator Tool: Devguide's code generator tool generates and compiles code, and runs applications for the GXV and GOLIT code generators, without requiring you to type commands at the command line. The code generator tool's main window is resizeable.The inner pane is a command tool**.**

Default Generated Code: Devguide now produces ANSI C as the default version of C. This means that procedure names and prototypes have their parameters listed in the declaration. For example:

```
void
my_callback(Panel_item item, Event *event)
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);
    fputs("foo: my_callback\n", stderr);
    /* gxv_start_connections DO NOT EDIT THIS SECTION */
    xv_set(Foo_window1->button2, XV_SHOW, TRUE, NULL);
    /* gxv_end_connections */
}
```

Devguide 3.0, by default, produced K&R C code such as this:

```
void
my_callback(item, event)
    Panel_item     item;
    Event          *event;
{
    foo_window1_objects *ip = (foo_window1_objects *)
xv_get(item, XV_KEY_DATA, INSTANCE);

    fputs("foo: my_callback\n", stderr);

    /* gxv_start_connections DO NOT EDIT THIS SECTION */

    xv_set(Foo_window1->button2, XV_SHOW, TRUE, NULL);

    /* gxv_end_connections */

}
```

Clearly, these forms are different. As a result, if your application is built with Devguide 3.0, you might experience problems merging the `_stubs.c` file(s). Also, if you use GXV and you did not use the -a switch parameter when you generated code, the 3.0.1 compiler will produce unexpected results when you compile your 3.0 application.

See Appendix B, “Porting Devguide 3.0 Applications to Devguide 3.0.1," in this manual for details.

Access From Within SPARCworks or ProWorks: If you are using SPARCworks or ProWorks, and you want to access Devguide through the SPARCworks or ProWorks manager, you can add Devguide to the manager as you would any tool. After you do this, the Devguide icon appears on the SPARCworks or ProWorks palette whenever you start up SPARCworks or ProWorks. You do not have to leave the SPARCworks or ProWorks context to use Devguide.

See Appendix C, “Adding Devguide to the SPARCworks or ProWorks Environment,” in this manual, for more information. The method described in Appendix C for adding Devguide to the SPARCworks or ProWorks manager is effective only for the current session of the manager. If you quit and restart the manager, the Devguide icon no longer appears on the SPARCworks or ProWorks palette. To permanently add Devguide to the SPARCworks or ProWorks manager, do the following:

1. **Copy the `sparcworks.defaults` or proworks.defaults file from the `lib` subdirectory where SPARCworks or ProWorks is installed. For example, if SPARCworks is installed in `/opt/SUNWste/SW2.0`, you would type the following:**

```
% cd
% cp /opt/SUNWste/SW2.0/lib/sparcworks.defaults .
% chmod +w sparcworks.defaults
```

2. **Edit the `sparcworks.defaults` or proworks.defaults file and make the following modifications:**
   - Add "devguide" (no quotes) to the end of the `swToolBox.tools` line.
   - Add the following to the bottom of the file:

```
swToolBox.applications.name: devguide
swToolBox.applications.glyph.label: Devguide
swToolBox.applications.glyph.file:
/opt/SUNWguide/include/images/guide_32X32.icon
swToolBox.applications.command: devguide $SUNPRO_SWM_TT_ARGS
$SUNPRO_SWM_GUI_ARGS
```

   - Modify the path to the icon file so it matches the path to where you have Devguide installed.

3. **Make sure that you have the Devguide `bin` directory in your path.**

4. **Start SPARCworks or ProWorks.**
   The Devguide icon should appear on the SPARCworks or ProWorks pallette.

Note – If SPARCworks or ProWorks generates an error message with reference to the `alert_qmark32.pr` icon, edit the `sparcworks.defaults` or proworks.defaults file to read:
`/usr/openwin/include/images/alert_qmark32.pr`

New GXV Features

GXV provides two new command line options (or flags) to enhance or customize code generation:

- `-k` (`-kandr`) —Specifies that K&R C code be generated (ANSI C is now the default).
- `-sb` —Adds the `-xsb` switch to the compiler options in the Makefile if you are using ANSI C. Adds the `-sb` switch if you are using K&R C. The resultant compiled code can then be used with Source Browser in SPARCworks.

See the *OpenWindows Developer's Guide: XView Code Generator Programmer's Guide* for complete information on these and other GXV command line options.

New GOLIT Features

GOLIT provides the following new command line options (or flags) for code generation:

- `-i` (`-international`) —Writes all Level 3 internationalization-specific resources into a resource file.
- `-k` (`-kandr`) —Writes K&R C code that doesn't contain function prototypes. Golit produces ANSI C code by default.
- `-r` (`-resources`) —Writes all resources into a resource file.
- `-sb` —Adds the `-xsb` switch to the compiler options in the Makefile if you are using ANSI C. Adds the `-sb` switch if you are using K&R C. The resultant compiled code can then be used with Source Browser in SPARCworks.

See the *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide* for complete information on these and other Golit command line options.

Devguide Changes From the Previous Release

Drag and Drop: In Devguide 3.0, a NOTIFY_SAFE interposer was placed on control areas and canvases that contained items or could themselves be the destination of a drop operation. Devguide 3.0.1 places a NOTIFY_IMMEDIATE as well as a NOTIFY_SAFE interposer on these objects.

D

GNT Changes From the Previous Release

The GNT (Guide to TNT) code generator, along with all pertinent GNT manuals, is now located in the demo directory. This means that GNT is not a fully supported code generator, and is instead provided as a demonstration code generator for the TNT (NeWS) toolkit.

GNT will not be enhanced in future releases of Devguide, and will be removed from the demo directory in the next release of Devguide. Those developers who have used GNT in the past should plan to migrate their work to other toolkits using the GXV (Guide to XView) or the GOLIT (Guide to OLIT) code generators.

Known Problems with the Devguide Software

Drag and Drop Targets: If you create a drag and drop target in your application, create a CallFunction connection with DroppedUpon as the When item. Select the drag and drop target as both Source and Target for the connection. If you do not create this connection, the drag and drop target will appear to accept drops from File Manager, even though no file transfer occurs. In this case, the drag and drop target will send a delete request to File Manager. This causes File Manager to delete the file, even though the drag and drop operation was unsuccessful. This behavior results from an XView bug.

In addition, you cannot drag and drop within the same application. That is, the source and destination can't be within the same application. This is not so much a drag and drop limitation as it is a Devguide gdd library problem.

Groups: When saving, Devguide may cause a core dump on a GIL file with a large number of groups. This occurs only on rare occasions. If it does happen, however, restart Devguide, ungroup one of the larger files (or the largest), and then save again.

Groups and Anchors: If you create a GIL file containing a group and anchor object, and the group is positioned above the object that you want to anchor it to, Devguide ignores the anchor object. That is, the group is not anchored to the object in this case.

Sharing Menus: If you create a menu that is shared by objects in more than one .G file, put the menu in a .G file separate from those containing the object descriptions. Load the .G file containing the menu object description whenever you load any .G file that references the shared menu. Menu references in any file that does not contain the menu description are lost.

Known Problems with the GXV and GXV++ Software

GXV Merging: The GXV merging facility will not catch all programmer errors. In particular, if you forget to close off a quoted string (that is, you only put in the first double quote) in your `_stubs.c` file, the code generated will not contain any of your changes. The `_stubs.c.BAK` file, however, retains your old version. If you have any doubts about a `_stubs.c` file, check your `.BAK` file, or copy your `_stubs.c.BAK` to `_stubs.c` and fix your code.

GXV and GOLIT Stubs File Merging: Merging the `*_stubs.c` files on successive .G file processing does not work for all valid C or C++ syntax. For example, `char * ptr = dgettext("Text1","Text2");` where `ptr` is global, produces the following error:

```
<codegen> cannot parse <file> beyond lib <line>
```

This construct does not cause an error, for example, within `main`() {}. To allow the file to be merged, comment out the offending lines before running GXV or GOLIT. Then uncomment the lines before compiling.

XVPS Library: GXV and GXV++ use the XVPS library to implement the PostScript drawing option for canvas panes. The XVPS library is an XView extension that is included with the OpenWindows Version 2 and Version 3 environments only as a demo. XVPS and the PostScript drawing model option are not supported by SunSoft.

Note – Applications should *not* rely on XVPS's future existence *nor* should they rely on future versions of XVPS being binary compatible with present versions.

Multiline Text Field Display: Due to an XView bug, multiline text fields do not display correctly when their parent control areas are displayed. This impacts applications in which control areas are made visible and invisible—for example, when using layered panes.

To remedy the problem, whenever your application uses XV_SHOW to display (hide) a control area you must insert code using XV_SHOW to display (hide) the multiline text field.

Help -? Option: The -? option does not work properly if run in the C shell. The ? is interpreted by the C shell, and is not passed on to gxv or gxv++. If you use the C shell, and you want to view a list of all GXV (GXV++) command line options, use the -help option instead.

Generating Help Files for Projects: You cannot generate help files for a project by using the -h option in combination with the -p option. The commands gxv -h -p my_project and gxv -p my_project produce the same results. They generate files associated with the .P file my_project. No help files are generated in either case. If you want to generate help files for a set of GIL files in a project, you mustgenerate the help files individually. Run gxv on each GIL file using the -h option.

Known Problems with GNT Software

Initial Values and Initial States: You must include application code to set initial states of menu buttons, such as Active/Inactive or Selected.

Groups: Groups create excess overhead and their use in GNT is not recommended.

Internationalization: Internationalization is not fully supported by TNT in OpenWindows Version 3. Window menus are not localizable in this release.

Localization: During localization, after the interface is frozen, do not group and ungroup buttons. Grouping and ungrouping buttons changes the number of tokens created. If you group or ungroup buttons during localization, the executable will no longer run because the number of tokens will no longer match the number of buttons.

Menus: Create a menu before you attach it to another object. Otherwise, the menu is referred to in the GIL file before it is defined, and you will get a PostScript undefined error. Alternatively, you can edit the GIL file so that the menus are defined before they are referred to.

Unresolved Installation Bugs

Problem: there are no links for doc and man after installation.

Workaround: Do the following to create the links:

```
% cd /opt/SUNWguide [or your $GUIDEHOME, if different]
% su
Password: [enter root password]
# ln -s share/man man
# ln -s share/doc doc
```

Unresolved Devguide Bugs

The currently outstanding bugs for Devguide (listed according to bug ID number) are as follows:

1071386: Borders for text panes are not shown while building an interface within Devguide. Generated applications layout will be somewhat different.
1086058: There is no way to specify the spacing between choices in checkbox settings.
1060374: An anchor object is not displayed on the Groups Properties window when loading a GIL file.
1091778: Can't use something like `foo::myfunc` for a call function.
1081031: Choosing Save As... with the Project Managers Project button doesn't save .G files.
1097702: Devguide doesn't preserve symbolic links to GIL files correctly.
1080931: Devguide mishandles newline in `PANEL_LABEL_STRING`.
1079760: Stubsmerge doesn't handle unpaired quotes well.
1077355: Can't remove unwanted connections from the .P file with Devguide.
1072360: Devguide doesn't support all create-only attributes of XVIEW.
1082534: `Printfs` are being placed inside the `gxv_start_connections` area.
1085133: Panel size in GXV generated code is different from Devguide prototype.

Unresolved GXV Bugs

1093669: GXV generates bad event handler for `PANEL_TEXT`.
1097049: GXV cannot parse function prototypes.
1096076: `gxv -h` does not work in conjunction with projects.
1095825: GXV generates code that does not compile when using ExecuteCode.
1078451: Code generation for setting item fails if "when" is deselected.

1082698: Menu handles aren't accessible to stubs file.
1087522: GXV cannot handle multi-level indirection of functions.
1081030: GXV++/stubsmerge doesn't deal with argument defaults well.
1099391: Can't use functions declared in `gfm_ui.h` from C++ in SVR4.
1088170: Dragging from a drop target to a scrolling list on the same panel doesn't work.
1082374: Colors are incorrect when used on MacX.

Unresolved GOLIT Bugs

1099852: Scrolling list items placed inside a scrolling list in Devguide don't show up in a GOLIT generated application.
1079909: There is no callback when a drag and drop from source to target is used.
1091684: GOLIT doesn't create multi-line text widget
1091376 OLIT canvas created via GOLIT doesn't resize properly
1113998 Null pointer passed to callback function; occasionally, for connections across interface files or across windows, code is generated that uses a null pointer within a callback.

Unresolved GUIL Bugs

1154709: GUIL truncates two characters of file name if you do not include .G or .P suffix when you execute command. WORKAROUND: include suffix.

GOLIT Features not Implemented in the Current Release

Bold Labels: The OPEN LOOK specification calls for the labels of all control objects to be bold. When you prototype an interface in Devguide, the label font will appear bold. However, the default font weight for labels in GOLIT-generated interfaces is not bold. When you run your GOLIT program, the labels will not be bold. You can set the caption font resource by setting the fallback resources in the `_stubs.c` (or <`project`>`.c`) file, or application defaults file. You can achieve the same effect by setting the caption font resource on the command line in the following manner:

```
% my_application -xrm '*Caption.font: lucidasans-bold'
```

where *my_application* is the name of the application for which you want the labels to appear bold.

Help: GOLIT does not support Help as implemented in Devguide. To add help for a widget, you must use a Create Proc and `OlRegisterHelp`(). For more information, see Chapter 4 of the *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide*.

Miscellaneous Unsupported UI Elements: The following UI elements are available in Devguide but are not currently supported by GOLIT:

- Glyph types in labels
  Chapter 4 of the *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide* provides an example of how you can use Create Procs to display any bitmap image in a label.
- Icons for Base Windows
  Chapter 4 of the *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide* provides an example of how you can use Create Procs to assign an icon to your application.
- Term Panes
- Numeric Text Fields
- Drag and Drop from a Canvas or a Scrolling List
- List items
  The scrolling list itself is supported. However, GOLIT omits any list items that you inserted in Devguide. You must write your own code to insert list items.
- Setting stack
  GOLIT provides limited support for this object. The current setting is not displayed next to the abbreviated menu button.
- Layered panes in pop-up windows
  If your interface has layered panes in a pop-up, the panes will be tiled in your compiled application. Layered panes do work in base windows.
- Slider end values
  Although you can include slider end values in your interface prototype in Devguide, they will not appear in your compiled application.

GOLIT simply ignores most of these unsupported elements when it reads the GIL file. However, if you attempt to use GOLIT to generate code for an interface that includes term panes or numeric Text Fields, GOLIT will abort code generation and display an error message.

D

GNT Features that are not Implemented in the Current Release

Connections: Some actions (connections) have not been fully implemented. An error message appears:

```
Undefined action for <object type> <name> What code <number> Action
<Action Name>
```

Miscellaneous Unsupported UI Elements: The following user interface elements are available in Devguide but are currently not supported by GNT.

- Term panes
- Multiline text fields
  Multiline text fields are converted to single line text fields.

Internationalization: GNT supports string translation, but does not support the following:

- Window menu does not reflect LOCALE
- Decimal delimiter is not changed for LOCALE

Layered Panes: You can layer only control areas in GNT.

Do not use a canvas object as a member of a layered pane; you will not be able to display the canvas because the Show action does not support canvas objects. When you use a canvas with a Show action, GNT issues the following warning and then exits.

```
{31} gnt test
Running Postscript Code Generator
Undefined action for Canvas gtst_window1_canvas1 What code 4
Action Show
```

GNT does not support layering text panes. When you use a text pane in a layer of panes, GNT issues the following warning and then exits.

```
{31} gnt test
Running Postscript Code Generator
Stacked text panes are not supported by gnt
```

Index

Symbols

A

D

E

F

G

H

I

K

L

M

Q

R

S

T

U

V

W

X