Advanced Topics 5

This chapter provides advanced examples of callbacks that you may find useful in writing your own applications.

Specifying an Icon for an Application

Golit does not allow you to specify an icon for your application in Devguide. However, you can write a Create Proc that displays a specified icon when the application is in an iconified state. To do this, you set up a Create Proc in Devguide for the Base Window. Assuming you call the icon `my.xbm`, you would modify the generated callback to appear like the one shown in Figure 5-1. You can create an icon with bitmap or IconEdit (make sure that the format is set to X Bitmap).

```
. . .
/* Put this line at top of  _stubs.c file  * /
#include "my.xbm"

. . .
Widget
create_window1(app_name, app_class, wdp, dpy, args, num_args,
                closure)
      String app_name;
      String app_class;
      register GolitWidgetDescPtr wdp;
      Display *dpy;
      ArgList args;
      Cardinal num_args;
      XtPointer closure;
{
      Widget toplevel;
      Pixmap toplevel_icon;
      Screen *screen;
      unsigned long fg, bg;
      unsigned int depth = 1;
      Arg arg;

      toplevel = GolitFetchShellHier(app_name, app_class, wdp,
                              dpy, args, num_args, closure);
      screen = XtScreen(toplevel);
      toplevel_icon = XCreateBitmapFromData(
                    XtDisplay(toplevel),  /* display */
                    RootWindowOfScreen(screen), /* drawable */
                    myicon_default_bits, /* bitmap data */
                    myicon_default_width,  /* width */
                    myicon_default_height /* height */);
      XtVaSetValues(toplevel,
           XtNiconPixmap, (XtArgVal)toplevel_icon,
           XtNiconName, (XtArgVal)"toplevel",
           NULL);

      return toplevel;
}
```

Figure 5-1 Base Window Create Proc Modified to Display Icon

Making a Text Field Read-Only

In OLIT there is no direct resource to make a Text Field read-write or read only. The only way to change this aspect of the Text Field's behavior is to access the textEdit widget that is associated with the Text Field and then set the resource XtNeditType with the appropriate value. Specify a create proc for the text field object and modify it to look like the code in Figure 5-2.

```
Widget
textfield1CP(name, wdp, parent, args, num_args, closure)
 String name;
 register GolitWidgetDescPtr wdp;
 Widget parent;
 ArgList args;
 Cardinal num_args;
 XtPointer closure;
{
Widget textfield;
Widget textedit;
Arg arg;

textfield = GolitFetchWidget(name, wdp, parent, args, num_args,
                          closure);
      XtSetArg(arg, XtNtextEditWidget, &textedit);
      XtGetValues(textfield, &arg, 1);
      XtSetArg(arg, XtNeditType, OL_TEXT_READ);
      XtSetValues(textedit, &arg, 1);
      return textfield;
}
```

Figure 5-2 Text Field Create Proc Modified to Make the textEdit Widget Read-Only

Specifying a Label Image for a Button

You can also use Create Procs to specify label images for a button. To do this, set up a Create Proc for the button. Assuming you call the glyph `mylabel.xbm`, you would modify the generated callback to appear like the one shown in Figure 5-3. You can create an image with bitmap or IconEdit (make sure that the format is set to X Bitmap).

```
. . .
/* Put this line at top of  */
#include "mylabel.xbm"
/* _stubs.c file  * /
. . .
Widget
create_button(name, wdp, parent, args, num_args, closure)
      String name;
      register GolitWidgetDescPtr wdp;
      Widget parent;
      ArgList args;
      Cardinal num_args;
      XtPointer closure;
{
Widget button;
XImage *label_image;
Display *dpy;

      button = GolitFetchWidget(name, wdp,
                     parent, args, num_args,closure);
      dpy = XtDisplay(button);
      visual = OlVisualOfObject(button);
      label_image = XCreateImage( dpy, /* display */
                           visual, /* visual*/
                           1,   /* depth*/
                           XYBitmap, /* format*/
                           0,   /* offset*/
                           mylabel_bits, /* bitmap data */
                           mylabel_width, /* width*/
                           mylabel_height, /* height */
                           8,   /* bitmap_pad*/
                           0    /* bytes_per_line */
                     );
      XtVaSetValues(button,
           XtNlabelType, (XtArgVal)OL_IMAGE,
           XtNlabelImage, (XtArgVal)label_image,
           NULL);
      return button;
}
```

Figure 5-3 Button Create Proc Modified to Display Icon Label