Introduction to Motif Programming  
*(lectures programs)*

---

**Example 1**  
`memo.c`:  
To display a string on the screen using `LabelWidget`  

_E.g._:  

```bash  
% memo "Call your wife"
```

This produces the following:  

![Call your wife](image)

**`memo.c outline:`**  

```c  
void main ( int argc, char **argv )  
{  
  Widget shell, msg;  
  XtAppContext app;  
  XmString xmstr;  
  
  shell = XtAppInitialize (&app, "Memo", NULL, 0, &argc, argv, NULL, NULL, 0 );  
  
  xmstr = XmStringCreateLtoR ( argv[1],  
```
XmFONTLIST_DEFAULT_TAG);
msg = XtVaCreateManagedWidget ( "message",
xmLabelWidgetClass, shell,
    XmNlabelString, xmstr, NULL);

// xmstr (derived from argv[1]) is the label displayed inside
the label widget
XmStringFree (xmstr);
XtRealizeWidget (shell);
XtAppMainLoop (app);

// That takes care of all events generated from the widget, e.g.,
resize event.

}  

Example 2  xecute.c:

To Execute a command after the user confirms
the action.
It is composed of:

A container widget:
BulletinBoardWidget, that contains:

  • A LabelWidget and
  • Two PushButtonWidget

Example:
% xecute "list all files?" "ls -lt"

This produces the following GUI:
void main ( int argc, char **argv )
{
    Widget   shell, msg, bb, yes, no;
    XtAppContext   app;
    XmStringxmstr;
    Dimension    height;

    shell = XtAppInitialize ( &app, "Xecute", NULL, 0, &argc, argv, NULL, NULL, 0 );
    bb = XtVaCreateManagedWidget ( "bboard", xmBulletinBoardWidgetClass, shell, NULL );

    xmstr = XmStringCreate ( argv[1],
          XmFONTLIST_DEFAULT_TAG );
    msg = XtVaCreateManagedWidget ( "message", 
          xmLabelWidgetClass, bb, 
          XmNlabelString, xmstr, XmNx, 0, 
          XmNy, 0, NULL );

    XtVaGetValues ( msg, XmNheight, &height, NULL );
// Retrieve the height of the label widget, so we know where to place the buttons

    yes = XtVaCreateManagedWidget ( "yes",
        xmPushButtonWidgetClass, bb,
        XmNx, 0, XmNy, height + 20, NULL );

    no = XtVaCreateManagedWidget ( "no",
        xmPushButtonWidgetClass, bb,
        XmNx, 200, XmNy, height + 20, NULL );

    XtAddCallback ( yes, XmNactivateCallback, YesCallback, ( XtPointer ) argv[2] );

    XtAddCallback ( no, XmNactivateCallback, NoCallback, NULL );

    XtRealizeWidget ( shell );
    XtAppMainLoop ( app );

}  

void YesCallback ( Widget w, XtPointer clientData, XtPointer callData )
{
    char * cmd = (char *) clientData;
    if ( cmd )
        system ( cmd );
    exit ( 0 );
}

void NoCallback ( Widget w, XtPointer clientData, XtPointer callData)  
{
    exit ( 0 );
}
Example 3: A drawing editor:

The code to create this interface is: editor.c

```c
void main ( int argc, char **argv )
{
    Widget shell, canvas, panel, commands, options;
    XtAppContext app;
}```
shell = XtAppInitialize ( &app, "Editor", NULL, 0, &argc, argv, NULL, NULL, 0 );

panel = XtCreateManagedWidget ( "panel",
xmFormWidgetClass, shell, NULL, 0 );

commands =
XtVaCreateManagedWidget ( "commands",
xmRowColumnWidgetClass, panel, XmNnumColumns, 3,
XmNorientation, XmHORIZONTAL,
XmNtopAttachment, XmATTACH_FORM,
XmNrightAttachment, XmATTACH_FORM,
XmNleftAttachment, XmATTACH_FORM,
XmNbottomAttachment, XmATTACH_NONE, NULL );

options =
XtVaCreateManagedWidget ( "options",
xmRowColumnWidgetClass, panel, XmNnumColumns, 1,
XmNorientation, XmVERTICAL,
XmNtopAttachment, XmATTACH_WIDGET,
XmNrightAttachment, XmATTACH_NONE,
XmNleftAttachment, XmATTACH_FORM,
XmNbottomAttachment,XmATTACH_FORM, NULL );

canvas =
XtVaCreateManagedWidget ( "canvas",
xmDrawingAreaWidgetClass, panel,
XmNtopAttachment, XmATTACH_WIDGET,
XmNtopWidget, commands,
XmNrightAttachment, XmATTACH_NONE,
XmNleftWidget, options,
XmNleftAttachment, XmATTACH_WIDGET,
XmNbottomAttachment,XmATTACH_FORM, NULL );

XtCreateManagedWidget ( "button1",
xmPushButtonWidgetClass, commands, NULL, 0 );
XtCreateManagedWidget ( "button2",
XmHORIZONTAL,
XmATTACH_WIDGET, NULL, NULL, 0 );
xmPushButtonWidgetClass, commands, NULL, 0 );
XtCreateManagedWidget ( "button3",
xmPushButtonWidgetClass, commands, NULL, 0 );

XtCreateManagedWidget ( "button1",
xmPushButtonWidgetClass, options, NULL, 0 );
XtCreateManagedWidget ( "button2",
xmPushButtonWidgetClass, options, NULL, 0 );
XtCreateManagedWidget ( "button3",
xmPushButtonWidgetClass, options, NULL, 0 );

XtRealizeWidget ( shell );
XtAppMainLoop ( app );

}

The resource file is: Editor

Editor outline:
*options*button1*labelString: Option 1
*options*button2*labelString: Option 2
*options*button3*labelString: Option 3

*commands*button1*labelString: Command One
*commands*button2*labelString: Command Two
*commands*button3*labelString: Command Three

We use:

% setenv XENVIRONMENT Editor
to associate the resource file with the program before execution.
Adding life to editor: Quit and Draw lines

**editor2.c outline:**

```c
Widget quit;
Display *display;
int screen;
long fgcolor, bgcolor;
XGCValues gcval;
GC draw;
Window window;
int pointx, pointy;
int FirstPt = TRUE;

void main ( int argc, char **argv )
{

    ......
    code here is the same as editor.c
    ......

    change: XtCreateManagedWidget ( "button1",
        xmPushButtonWidgetClass, commands, NULL, 0 );
        XtCreateManagedWidget ( "button2",
        xmPushButtonWidgetClass, commands, NULL, 0 );
        XtCreateManagedWidget ( "button3",
        xmPushButtonWidgetClass, commands, NULL, 0 );

    to: quit = XtCreateManagedWidget ( "Quit",
        xmPushButtonWidgetClass, commands, NULL, 0 );
        no = XtCreateManagedWidget ( "no",
        xmPushButtonWidgetClass, commands, NULL, 0 );
        yes = XtCreateManagedWidget ( "yes",
        xmPushButtonWidgetClass, commands, NULL, 0 );
```
Add the call back:

```c
XtAddCallback ( quit, XmNactivateCallback, quitCallback, NULL );
XtAddCallback ( no, XmNactivateCallback, noCallback, NULL );
XtAddCallback ( yes, XmNactivateCallback, yesCallback, NULL );
```

Add the following code in order to draw lines on the canvas:

```c
display = XtDisplay(shell);
screen = DefaultScreen(display);

XtVaGetValues ( canvas, XmNforeground, &gcval.foreground, XmNbackground, &gcval.background, NULL );

gcval.foreground = gcval.foreground ^ gcval.background;
draw = XtGetGC ( canvas, GCForeground | GCBackground, &gcval );

XtAddEventHandler(canvas, ButtonPressMask, FALSE, HandleBoardEvents, NULL);

.....
```

void quitCallback ( Widget w, XtPointer clientData, XtPointer callData) {
{

  Colormap cmap;
  XColor color, ignore;
  char *colorname = "red";

  XtSetArg(wargs[0], XmNlabelString, XmStringCreateLocalized("Are you sure?")));
XtSetValues(quit, wargs, 1);
XtMapWidget(no);
XtMapWidget(yes);

cmap = DefaultColormap(display, screen);
XAllocNamedColor(display, cmap, colorname, &color, &ignore);
XtSetArg(wargs[0], XmNbackground, color.pixel);
XtSetValues(quit, wargs, 1);

}

void yesCallback ( Widget w, XtPointer clientData, XtPointer callData) {
    exit(0);
}

void noCallback ( Widget w, XtPointer clientData, XtPointer callData) {
    XtSetArg(wargs[0], XmNlabelString, XmStringCreateLocalized("QUIT"));
    XtSetValues(quit, wargs, 1);
    XtMapWidget(quit);
    XtUnmapWidget(no);
    XtUnmapWidget(yes);
}

void HandleBoardEvents ( Widget w, XtPointer clientData, XEvent *event, Boolean *flag ) {
    if (FirstPt) {
        FirstPt=FALSE;
        pointx = event->xbutton.x;
        pointy = event->xbutton.y;
        XDrawPoint (XtDisplay(w), XtWindow(w),
                    draw, pointx, pointy);
    }
}
else {
    FirstPt=TRUE;
    XDrawLine(XtDisplay(w), XtWindow(w), draw, pointx, pointy, event->xbutton.x, event->xbutton.y);
}