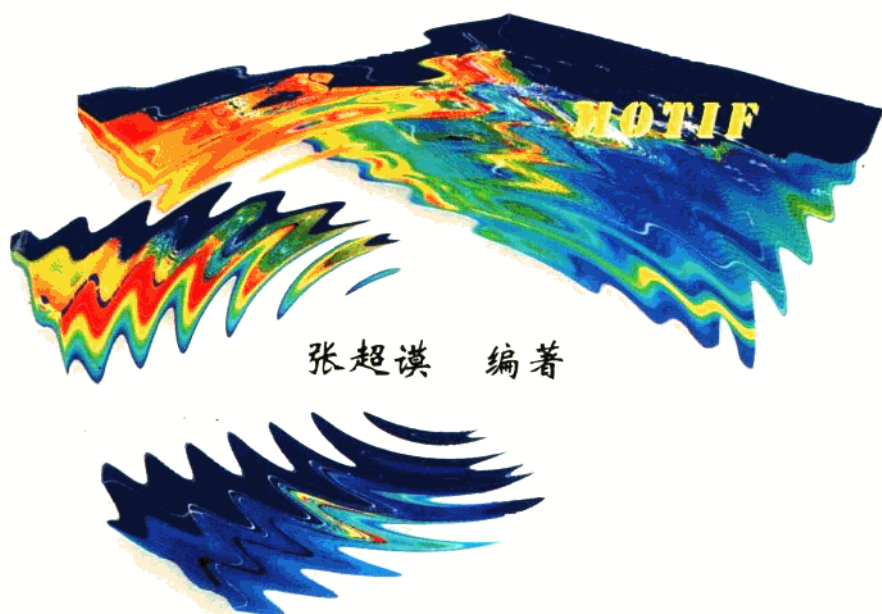




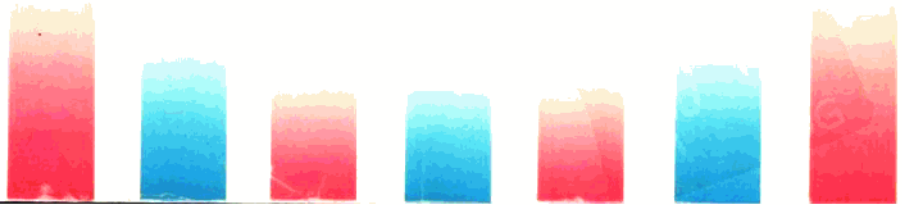
MOTIF

编程方法及实例



张超谟 编著

石油工业出版社



前 言

1991年,我们承担了中国石油天然气总公司八·五重点攻关项目“裂缝性碳酸盐岩测井解释工作站应用软件研究”,当时,采用的用户界面为 OpenLook,图形显示为 GKS。1993年,由于重新调整规划内容,要求采用 Motif 作用户界面,图形显示必须用 Motif 界面控制下的 Xlib 绘图语句。研究人员按手中所有的参考资料,花了一年的时间完成界面及图形的转换工作,但对其中的过程还是不清楚。为使新的研究人员在使用 Motif 编程方面少走弯路,我们首次应用地球物理专业硕士研究生班中开设“MOTIF 编制方法及实践”课程,在一个月之内,研究生可以从对 Motif 一无所知到熟练编程,最终提交综合作业,并能熟练使用 Motif 资源。

本书的特点是由简单具体到复杂全面,每个例子都有对应的结果,所有图件均是在 Sun-SparcPrinter 激光打印机上照相拷贝的,有时读者不用上机就能明白其执行过程。读者学完第一章后,就可以着手编制、调试程序。书中,对每一资源的说明都是在逐一作了调试之后按个人的理解来叙述的。第二章到第六章对各个功能作了完整全面的描述。第七章的绘图部分除叙述 Motif 的内容外,加了大部分 Xlib 的绘图功能。第八章的综合例子,尽管从使用上有更好的程序,但从综合理解及进一步的功能扩充上,使用了最简单且保持几乎全部功能的程序段。

对 Motif 未接触的读者,可从第一章看起,然后,根据需要从任何一章看起,各章基本上可以独立。

由于作者水平有限,Motif 也在不断发展中,对其中的一些问题没有理解透彻甚至错误之处,请专家、读者批评指正。

作 者

1997年10月

目 录

第一章 Motif 概述	(1)
第一节 X 简介	(1)
一、X-Window 的基本概念	(1)
二、X 的工具箱	(2)
三、X 的层次结构	(2)
四、Motif 程序的编译	(2)
五、X 应用程序的运行	(3)
第二节 Motif 图形用户界面系统的开发工具	(3)
一、专门供开发人员使用的基本开发工具	(4)
二、用户界面语言 UIL	(4)
三、交互式开发工具 bx	(4)
第三节 Motif 程序的组成	(4)
一、一个简单的 Motif 程序	(4)
二、widget 与 gadget	(6)
三、Motif 程序的头文件	(9)
四、初始化 XtIntrinwics	(10)
五、创立 widget	(11)
六、管理 widget	(11)
七、为 widget 加回调、事件处理或动作映射函数	(12)
八、实现	(12)
九、事件循环	(12)
十、放弃或取消 widget	(12)
第四节 命名规则	(13)
第五节 Motif 资源	(13)
一、资源文件格式	(13)
二、资源命名	(14)
三、资源文件的查找	(15)
四、用程序设置资源值	(15)
五、在程序中获取资源值	(16)
第二章 管理器型 widget	(17)
第一节 通告栏 XmBulletinBoard	(17)
一、包含文件	(17)
二、建立 BulletinBoard	(17)
三、BulletinBoard 常用资源	(18)
四、使用 BulletinBoard 举例	(18)

第二节 标号 XmLabel (元 widget)	(20)
一、包含文件	(21)
二、建立 Label	(21)
三、Label 常用资源	(21)
四、XmString 数据类型	(22)
五、常用的 XmString 函数	(23)
六、Pixmap 数据类型	(23)
七、使用 Label 举例	(24)
第三节 行列 XmRowColumn	(26)
一、包含文件	(26)
二、建立 RowColumn	(26)
三、RowColumn 常用资源	(27)
四、使用 RowColumn 举例	(28)
第四节 活动窗格 XmPanedWindow	(29)
一、包含文件	(29)
二、建立 PanedWindow	(29)
三、PanedWindow 常用资源	(30)
四、使用 PanedWindow 举例	(30)
第五节 表格 XmForm	(31)
一、包含文件	(31)
二、建立 Form	(32)
三、Form 常用资源	(32)
四、使用 Form 举例	(33)
第六节 三维边框 XmFrame	(36)
一、包含文件	(36)
二、建立 Frame	(36)
三、Frame 常用资源	(36)
四、使用 Frame 举例	(36)
第七节 滚动窗 XmScrolledWindow 与滚动条 XmScrollBar	(38)
一、滚动窗 XmScrolledWindow	(38)
二、使用 ScrolledWindow 举例	(40)
三、滚动条 XmScrollBar	(42)
四、使用滚动条举例	(44)
第八节 标尺 XmScale	(46)
一、包含文件	(46)
二、建立 Scale	(46)
三、Scale 常用资源	(46)
四、Scale 回调	(47)
五、对 Scale 值的存取	(48)
六、使用 Scale 举例	(48)

第三章 元 widget	(56)
第一节 按钮	(56)
一、下压按钮 XmPushButton	(56)
二、箭头按钮 XmArrowButton	(57)
三、开关按钮 XmToggleButton	(59)
四、绘画按钮	(63)
第二节 键盘输入	(65)
一、输入焦点与遍历	(65)
二、XmText 与 XmTextField	(68)
三、滚动正文	(74)
第三节 列表 XmList 与滚动列表 XmScrolledList	(76)
一、列表 List	(76)
二、滚动列表 ScrolledList	(81)
第四章 事件与回调	(85)
第一节 事件	(85)
一、登记事件	(85)
二、XEvent 联合	(86)
三、事件屏蔽码	(87)
四、X 事件的种类	(89)
五、事件使用举例	(93)
六、为不可屏蔽事件、登记事件处理函数	(99)
第二节 回调	(100)
一、登记回调	(100)
二、回调函数形式	(100)
三、回调结构	(100)
四、删除回调	(101)
五、回调资源	(101)
六、常用 widget 回调速查表	(108)
第三节 动作	(118)
一、动作举例	(118)
二、动作表	(120)
三、安装动作表	(120)
四、动作函数形式	(120)
五、转换表	(121)
第四节 其它事件处理技术	(126)
一、定时调用	(126)
二、后台处理	(128)
三、文件输入与输出	(132)
第五章 菜单	(136)
第一节 标准 Motif 程序的外观	(136)

一、包含文件.....	(136)
二、创建 MainWindow	(136)
三、MainWindow 常用资源	(137)
第二节 菜单.....	(140)
一、菜单条.....	(141)
二、下拉菜单.....	(145)
三、弹出菜单.....	(148)
四、选择菜单.....	(152)
第六章 对话框	(154)
第一节 消息对话框 MessageBox	(154)
一、包含文件.....	(154)
二、创建消息对话框.....	(154)
三、消息对话框常用资源.....	(155)
四、对话框的态性.....	(157)
五、对话框的管理.....	(158)
六、获取 XmMessageBox 的内部子成分.....	(158)
第二节 选择对话框 XmSelectionBox	(162)
一、包含文件.....	(162)
二、创建选择对话框.....	(162)
三、选择对话框常用资源.....	(164)
四、获取选择对话框的孩子.....	(165)
五、提示对话框.....	(170)
六、文件选择对话框.....	(171)
七、命令对话框.....	(172)
第三节 定制对话框.....	(174)
第七章 绘图	(183)
第一节 标准 Xlib 参数	(183)
一、说明显示器.....	(183)
二、目标窗口.....	(183)
三、Pixmap 数据类型	(184)
四、图形属性集.....	(184)
第二节 颜色.....	(209)
一、颜色映射表.....	(209)
二、颜色的使用方法.....	(215)
第三节 常用 Xlib 绘图函数	(217)
一、画点画线函数.....	(217)
二、填充函数.....	(220)
三、绘字符串函数.....	(221)
第四节 绘画管理器 XmDrawingArea	(221)
一、包含文件	(221)

二、创建 DrawingArea	(221)
三、DrawingArea 常用资源	(222)
四、使用 DrawingArea 举例	(223)
第八章 Motif 综合编程方法	(248)
第一节 方法简述与操作过程	(248)
一、方法简述	(248)
二、操作过程	(250)
第二节 程序及详细注解	(255)
参考文献	(287)

第一章 Motif 概述

用户界面是使用计算机的人与计算机系统进行对话的接口,是软件的组成部分之一。图形用户界面(Graphics User Interface,简称 GUI)则是一种以图形方式进行人机交互的用户界面。

80 年代末出现了两个工业标准制定组织:Unix International(UI)和 Open Software Foundation(OSF)。它们的任务都是制定标准化的操作系统,UI 制定了 Unix System V, OSF 制定了 OSF/1,由于双方的合作,每个产品都是基于 X 的界面,UI 的界面是 OpenLook, OSF 的界面是 Motif。HP、Digital、IBM 等是 OSF 的成员, Sun 公司是 UI 的成员。在 Sun 的机器上的界面就是 OpenLook,但同时也支持 Motif,如今 Motif 似乎已经成为一种事实上的图形用户界面标准。

Motif 是一套准则,规定了 GUI 的外观(应用程序在屏幕上的显示方式和布局)及感觉(规定用户如何与应用程序打交道),这些准则与运行程序的机器无关。

由于 OpenLook 及 Motif 都选 X 窗口系统作为基础(平台),所以为了使用 Motif 或 OpenLook,必须对 X 有一定的了解,同时要能够熟练地使用 C 语言编程。

第一节 X 简介

Motif 是建立在 X - Window 之上的一个图形用户界面产品,它以 X - Window 作为显示模型和窗口模型,所以必须对 X 有一定了解才能灵活运用。

X 窗口系统采用了客户/服务器式的结构模型。客户也称客户程序,是指在本地或网上运行的利用 X 进行显示输出的应用程序。X 服务器是一个在图形工作站上运行的服务进程,负责对显示器的输出、键盘和鼠标的输入进行管理。

客户程序可以根据需要调用 Xlib 库函数,这些库函数将被转换为 X 的协议请求并通过网络发送给服务器,服务器按协议的要求完成指定的任务。

一、X - Window 的基本概念

(1) 屏幕:指用来显示图形和正文的硬件输出设备。

(2) 显示器:指 X 服务器及它控制的键盘、鼠标和屏幕。X 允许一个服务器控制多个屏幕,但在只有一个 CPU、一个屏幕、运行一个 X 服务器的情况下,屏幕与显示器常交替使用。

(3) 资源:X 应用程序所使用的窗口、字体、颜色表及其它数据结构的总称。资源存储在 X 服务器中,由服务器控制和管理。

(4) 请求:应用程序通过向服务器发送请求来使用服务器所提供的各种服务,通俗地讲就是程序要干什么。例如请求创建窗口、在窗口中显示文本等。

应用程序发出的请求按发送的先后次序放在请求队列中,应用程序发出请求后继续运行,不等待服务器的响应,而服务器则按队列中的次序依次处理各个请求。

(5) 事件:X 服务器以发送事件的方式向应用程序传送信息。事件是一种数据结构,包括事件类型、事件发生的窗口及其它相关信息,它常是用户操作的结果,服务器将这些结果报告给应用程序。例如用户按下鼠标,就会产生相应的 X 事件,服务器将这一事件发送给应用程序,应用程序根据事先安排好的代码进行相应地处理。

(6) 窗口:指屏幕上的一块矩形区域,具有颜色、大小等各种属性。窗口是由程序请求服务器建立的,服务器将为每一窗口维护相关的信息。

二、X 的工具箱

Xlib 提供了大量的函数供用户调用,但根据这些函数调用编写程序界面工作量很大,为此,在 X 窗口系统之上引入了高层的软件开发工具箱。X Toolkit Intrinsics(简称 Xt Intrinsics)是其中的一种工具箱,它提供了一组高级的实用函数和一些定义好的界面元件,用户用它可以节省许多工作量。

三、X 的层次结构

参见图 1—1, X 用户程序由三层组成:顶层、中间层(工具层)、底层。

(1) 最顶层为程序代码,用于对数据进行操作及定义用户界面,可用(Motif) Xm、Xt 或 Xlib。所以一般我们所说的 Motif 应用程序就是指使用 Xlib、Xt Intrinsics 和 Motif Toolkit 编写的用户程序。一般而言,尽可能使用高层次的库,即对于完成同一功能而言,尽量使用 Motif 的 widget 和函数,而不使用 Xt 和 Xlib。

(2) Motif 提供各界面元件及控制所需的功能,且 Motif 既使用了 Xt,也使用 Xlib。

(3) 最底层为 X 协议,Xlib 提供了使用 X 协议的接口,用户不必关心这一部分的实现方法。

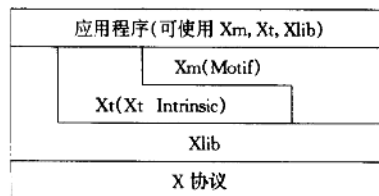


图 1—1 X 的层次结构

四、Motif 程序的编译

```
cc sourcename.c -D_NO_PROTO -lXm -lXt -lX11 -o program
```

其中:

sourcename.c 为 c 源程序名;

program 为编译之后形成的执行文件名;

D_NO_PROTO 是对 c 源程序编译时所必须的选项(原因见后面第三节 Motif 程序的头文件说明);

后面的 Motif 库(Xm), 工具箱库(Xt)以及 Xlib 库(X11)的次序很重要, 因为 C 编译器将按指定的顺序寻找库, 由图 1—1 可看出, Motif 依赖于 X 工具箱, 而 X 工具箱又依赖于 Xlib, 如果说明的次序不对将导致编译错误。

五、X 应用程序的运行

X 的应用程序是一种事件驱动的非模块化程序, 它在结构上可分为三个部分。

1. 初始化部分

- (1)建立显示连接;
- (2)建立窗口;
- (3)建立资源;
- (4)转下面的“事件等待”。

2. 事件处理循环部分

- (1)等待事件;
- (2)读取事件;
- (3)检查事件类型;
- (4)处理事件;
- (5)判断是否退出事件处理循环, 是则转下面的“关闭窗口”, 否则转(1)再次等待事件。

3. 结束部分

- (1)关闭窗口;
- (2)释放资源;
- (3)断开显示连接。

第二节 Motif 图形用户界面系统的开发工具

开发工具可以给编程人员提供很多方便, 但使用开发工具均要求对 Motif 有充分的了解, 否则使用开发工具是有很大的困难的。对 Motif 十分熟悉的程序员, 往往偏向于自己动手写程序。

一、专门供开发人员使用的基本开发工具

这类工具有 Xlib、Xt Intrinsics、Motif Toolkit 等。它们均以库函数的形式提供,供用户编程调用,这类工具要求开发人员具备 Motif 及相关知识,一般由熟练的程序员使用。

二、用户界面语言 UIL

UIL 只要用少量的描述性语句即可编制出应用程序的图形用户界面。使用 UIL 语言的缺点是程序的后期维护工作较困难。

三、交互式开发工具 bx

bx 是 Computer Solution 公司的 Builder Xcessary 产品,它将界面上的各种元件以直观的方式显示在一块面板上,用户用鼠标器对之进行操作,捡起所需元件进行组装,然后根据需要可形成 UIL、C++、C 等语言程序,只要使用者懂得 Motif 的概念及各元件的意义与关系就可以使用 bx。

第三节 Motif 程序的组成

一、一个简单的 Motif 程序

widget 的数据类型:widget 是一个在 Motif 程序中提供用户界面功能的对象,一个具有 widget 类型的变量可以存放一个 WidgetID,WidgetID 是唯一标识 widget 的整数值,在 widget 内部,这个 ID 码被视为指向 widget 事例数据的指针。

一个完整的 Motif 程序包括六个部分:

- (1)包含有关头文件;
- (2)初始化;
- (3)widget 的创建与管理;
- (4)为所建立的 widget 建立回调函数和/或事件处理函数和/或动作映射表;
- (5)实现所有的 widget;
- (6)事件循环。

下面以例子进行说明。例 1—1 与例 1—2 都完成同样的功能,体现了两种编程风格。

例 1—1

```
#include<Xm/Xm.h> /*公共定义头文件 7
#include<Xm/PushB.h> /*下压按钮头文件 7
main(argc,argv)
```

```

char * argv[];
{
    Widget toplevel, button; /* 定义所用到的 widget */
    XtAppContext app; /* 应用程序描述表的地址 */
    void i_was_pushed(); /* 定义回调函数 */
    XmString label; /* 定义 label 为复合串型 */
}
/*

```

初始化: 返回一个 shell 类 widget, 它负责处理应用程序与窗口管理器之间的关系, 并作为应用程序的顶层窗口, 程序中的所有其它 widget 都是 shell 的子 widget

app: 应用程序描述表的地址, 描述表是一个结构, Xt 将在其中处理一些与应用程序相关的关系, 同时它又是 Xt 内部的数据, 其它参数意义见后面说明

```

}
toplevel = XtVaAppInitialize(&app, "Simple1", NULL, 0,
    &argc, argv, NULL, NULL); /* 建顶层外壳 */
label = XmStringCreateSimple("Push here to say hello"); /* 产生一个复合串 */
button = XtVaCreateManagedWidget("Pushme", /* 建立一个下压按钮并进行管理 */
    xmPushButtonWidgetClass, toplevel, /* 下压按钮的父为 toplevel */
    XmNlabelString, label, /* 下压按钮的标号为 Push here to say hello */
    XmNwidth, 200, /* 下压按钮的宽 200 */
    XmNheight, 100, NULL); /* 下压按钮的高为 100 */
XmStringFree(label); /* 释放 label */
XtAddCallback(button, XmNactivateCallback, /* 为下压按钮加激活回调 */
    i_was_pushed, NULL); /* 回调函数为 i_was_pushed */
XtRealizeWidget(toplevel); /* 实现 toplevel */
XtAppMainLoop(app); /* 进入事件主循环 */
{
    void i_was_pushed(w, client_data, cbs) /* 回调函数 */
    Widget w; /* 变量说明 */
    XtPointer client_data;
    XmPushButtonCallbackStruct * cbs;
}
printf("Hello yourself! \n"); /* 本回调函数很简单, 在用户命令窗口显示 Hello Yourself */
}

```

例 1—2

```

#include <Xm/Xm.h>
#include <Xm/PushB.h>
main(argc, argv)
char * argv[];
{
    Widget toplevel, button;
    XtAppContext app;
    void i_was_pushed();
    XmString label;
    toplevel = XtVaAppInitialize(&app, "Simple2", NULL, 0, &argc, argv, NULL, NULL);
}

```

/* 用 Motif 提供的函数建按钮, 按钮的参数用资源(放在资源文件中)说明, 其功能同例 1—1, 但比例 1—1 更灵活 */

```
button = XmCreatePushButton(toplevel, "Pushme", NULL, 0);
XtManageChild(button); /* 对按钮进行管理 */
XtAddCallback(button, XmNactivateCallback, i_was_pushed, NULL);
XtRealizeWidget(toplevel);
XtAppMainLoop(app);

void i_was_pushed(w, client_data, cbs)
Widget w;
XtPointer client_data;
XmPushButtonCallbackStruct * cbs;
{
    printf("Hello yourself! \n");
}
```

资源文件(说明下压按钮的资源, 本程序的类名为 Simple2)

Simple2.Pushme.labelString: Push here to say hello (下压按钮标号)

Simple2.Pushme.width: 200 (下压按钮的宽度)

Simple2.Pushme.height: 100 (下压按钮的高度)

Simple2.Pushme.background: White (下压按钮的背景色)

Simple2.Pushme.foreground: Black (下压按钮的前景色)

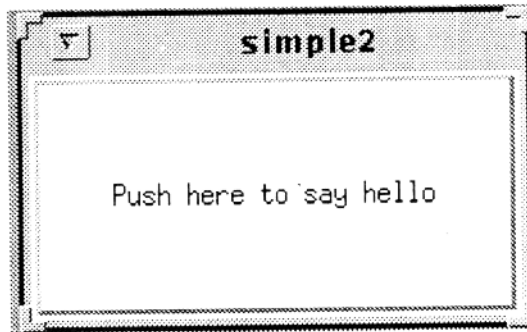
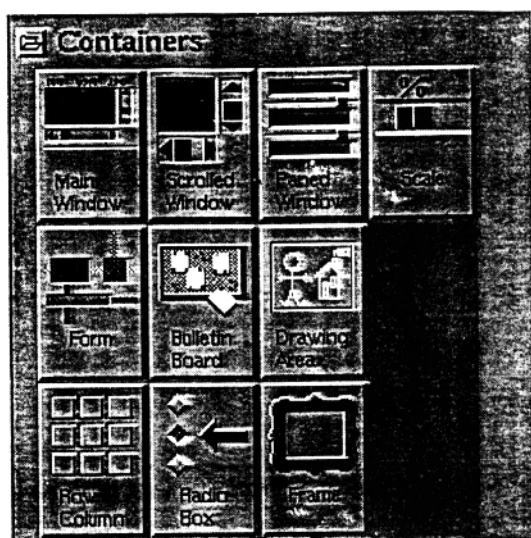


图 1—2 例 1—2 的执行结果

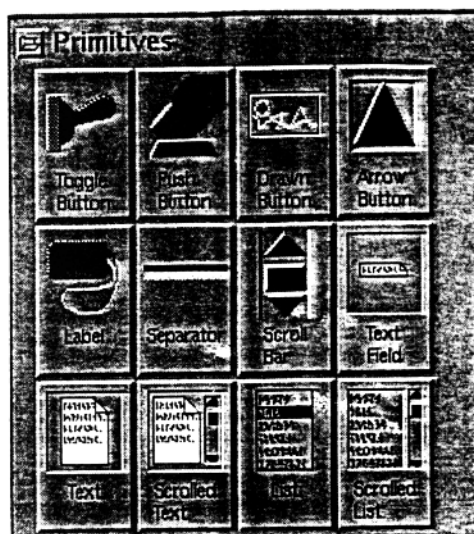
二、widget 与 gadget

Motif 的工具库中包含许多基本部件, 程序员可以把它们安装在自己的图形用户接口中。

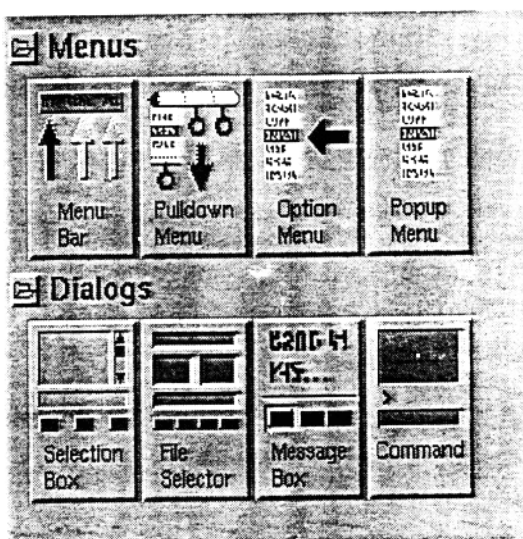
通常, widget 是指一件物品, 在 X - Window 系统中, 它是一个控制应用程序与 X - Window 交互的对象。实现这种 widget 有一套规则与原理, 但应用程序员不必关心这些原理到底是什么。对用户来说, 我们按其形式, 从使用的角度将其分为: 元 widget、管理器型 widget、菜单型 widget 及对话型 widget。见图 1—3。



(a)



(b)



(c)

图 1—3 各种 widget 示意图

1. 元 widget

- Label 显示文本或图像形式的标号, 无回调。
- PushButton Label 的子类, 同时还提供了回调资源, 外观上具有阴影边界。
- DrawnButton 与 PushButton 类似, 但主要用于画出文本或图像, 这里考虑到图像是变化的。

·ToggleButton 开关按钮,但按钮的标号显示在右边,按钮下凹指示开,按钮凸起指示关。

·CascadeButton 只能作为 RowColumn 的子 widget,用于挂接菜单。

·ArrowButton 箭头按钮,它不是 Label 的子类,所以不显示标号,它的箭头分四个方向,即上下左右。

·List 列表项,该列表中的项可由用户选择。

·ScrollBar 滚动条,用于当屏幕显示不下时,可以上下左右移动显示。

·Separator 分隔符,提供视觉上的分隔效果;

建立方法为:

```
Widget sp = XmCreateSeparator(parent, name, args, n)
```

```
XtManageChild(sp)
```

·Text 文本编辑器,它提供了不同的资源,程序员可用这些资源设置 widget 的编辑方式。

·TextField 与 Text 相似,它只是单行的文本编辑器而已。

2. 管理器型 widget

元 widget 单独使用时并不能生成一个有效的接口,例如随意地将很多按钮放在一起可能使程序变得很复杂,因此要对界面进行合理地组织和布局,这是通过管理器型的 widget 实现的,管理器型 widget 基本上没有与用户交互的回调例程,它的主要功能就是管理其中的子 widget 的大小与位置。

·BulletinBoard 通告栏,可以放置任何子 widget 的管理器型 widget,主要用于 Motif 预定义好的对话 widget。

·Form BulletinBoard 的子类,用于控制子 widget 的位置和大小。

·RowColumn 行列 widget,其子 widget 均按行列方式布局。

·Frame 为一些不具边界的 widget 提供一个三维的边界。

·PaneWindow:后代按一列多行放置的管理器型 widget,其中的子 widget 的函数高度可由用户调节,所有子 widget 的宽度取各子 widget 中最宽的一个。

·DrawingArea 绘画区,提供显示图形的区域。

·ScrolledWindow 滚动窗,当被观察的对象超出了 ScrolledWindow 的大小时,将自动加上滚动条,以便让用户操纵滚动条观看。

·MainWindow 主窗口,该窗口提供了标准布局,在其上可安装菜单条、命令窗、工作区、水平滚动条、垂直滚动条。

·Scale 标尺,与 ScrollBar 类似(两端无步进箭头)的对象。

3. 类与事例

类是一些具有相似特性的对象的抽象,赋与类的属性以特定的值则称为该类的事例,即类的具体化。以例子进行说明:如下压按钮为一个类,有标号、宽、高等,而具体的下压按钮则是这个类的事例。如例 1—2 下压按钮标号为 Push here to say hello、下压按钮的宽度为 200、下压按钮的高度为 100;下压按钮的背景色为 White、下压按钮的前景色为 Black。

继承是指从一已经存在的类(超类)中派生出一新类(子类)。子类继承了超类中的类属性

和类方法。

4. gadget

gadget 是 DEC-window 对 Motif 的贡献之一, 一个 gadget 是一个无窗口的 widget, 大部分 widget 都有对应的 gadget。所谓无窗口指的是 gadget 的外观完全由它的父决定, 例如设有一绿色前景红色背景的对话框, 则对话框中的内容可以是 widget 或 gadget, 如果为 gadget, 那么它使用与对话框一致的颜色, 如果为 widget 则可以为别的颜色。

gadget 相对于 widget 是它的效率高, 一个 widget 是一个窗口, 必须由服务器维护, 而 gadget 只是父窗口中的区域, 所需开销较少。

三、Motif 程序的头文件

Motif 的头文件放在 /usr/include/Xm 目录下, Xlib 与 Xt 的头文件放在 /usr/include/X11 目录下。

以 P 结尾的头文件是 widget 类使用的私有头文件, 它们包含了 widget 类内部数据定义, 用户程序中不能直接包含这些头文件, 以 G 结尾的头文件则是 gadget 的专用头文件。

表 1-1

widget 类	头文件
公共定义	< Xm/Xm. h>
XmArrowButton 箭头按钮	< Xm/ArrowB. h>
XmBulletinBoard 通告栏	< Xm/BulletinB. h>
XmCascadeButton 级联按钮	< Xm/CascadeB. h>
XmCommand 命令框	< Xm/Command. h>
剪裁板	< Xm/CutPaste. h>
XmDialogShell 对话框	< Xm/DialogS. h>
XmDrawingArea 绘画区	< Xm/DrawingA. h>
XmDrawnButton 绘图按钮	< Xm/DrawnB. h>
XmFileSelectionBox 文件选择框	< Xm/FileSB. h>
XmForm 依联管理器	< Xm/Form. h>
XmFrame 三维视觉框	< Xm/Frame. h>
XmLabel 标号	< Xm/Label. h>
XmList 列表	< Xm/List. h>
XmMainWindow 主框	< Xm/MainW. h>
XmMenuShell 菜单	< Xm/MenuShell. h>
XmMessageBox 信息对话框	< Xm/MessageB. h>
XmPanedWindow 窗格框	< Xm/PanedW. h>
XmPushButton 下压按钮	< Xm/PushB. h>
XmRowColumn 行列管理器	< Xm/RowColumn. h>
XmScale 标尺	< Xm/Scale. h>

widget 类	头文件
XmScrollBar 滚动条	<Xm/ScrollBar.h>
XmScrolledWindow 滚动条	<Xm/ScrolledW.h>
XmSelectionBox 选择框	<Xm/SelectionB.h>
XmSeparator 分隔条	<Xm/Separator.h>
XmText 文本编辑	<Xm/Text.h>
XmToggleButton 开关选择	<Xm/ToggleB.h>

Motif 的应用程序中一般应包括以下的头文件:

C 语言的一般头文件, 如 <stdio.h>、<ctype.h>、<math.h> 等。

Xm/Xm.h 定义了 Motif Toolkit 所使用的变量和数据类型。

所用到的各个 widget 所对应的头文件。例如, 前面的例子中用到了 PushButtonWidget, 因此必须包含头文件 <Xm/PushB.h>, 一般, 某一 widget 的头文件均已包含了 <Xm/Xm.h>。

由于 C 和 C++ 的函数说明语句的语法规则不同, 为了能使 Motif 的应用程序既可用 C 语言编写, 又可用 C++ 编写, 所以在 <X11/Xm.h> 之中大量使用了条件编译, 格式如下:

```
#ifdef _NO_PROTO
extern void XmListAddItem( );
#else
#if defined( __cplusplus ) || defined( cplusplus )
extern void XmListAddItem(Widget w, XmString item, int pos);
#endif
#endif
```

因 Sun C 编译程序要求函数说明语句中不能带参数, 所以凡用 C 语言编写的 Motif 应用程序在编译时都要用 -def 选项对常量 _NO_PROTO 加以定义, 或者在程序中用 #define 语句在 #include 之前进行定义:

```
#define _NO_PROTO
```

四、初始化 XtIntrinwics

```
Widget XtAppInitialize(app _context _return,
    class, options, num _opts, argc, argv, fallback _resources, args, num _args)
```

初始化函数将建立应用程序描述表; 建立应用程序与窗口管理器之间的联系; 将各资源文件和窗口特性中的资源描述装入到资源数据库中; 建立一个顶层 widget。其中:

app _context _return 为应用程序描述表的地址, 有关函数将要用到这个变量。应用程序描述表将一个应用程序划分成多个逻辑上的应用程序, 同属于一个应用程序描述表的部分为一个逻辑应用程序;

class 为应用程序类名。资源管理程序用此名字来指定有关资源, 这样, 不同的应用程序就可以根据自己的类名来分配资源的值;

options、num _opts 分别为资源管理器设定附加命令的选项及选项个数, 若不用则分别用