

(共五册)

HOPE

Turbo Pascal for Windows

技 术 丛 书



程序设计指南

4

北京希望电脑公司

(共五册)

Turbo Pascal for Windows

技 术 丛 书

Windows 程序设计指南

博 山 编 译

4

北 京 希 望 电 脑 公 司

一 九 九 一 年 八 月

前 言

Microsoft Windows 3.0 是一个多任务环境,它覆盖了 DOS 的全部功能,提供了直观优美的公用图形用户界面、高级应用程序编程接口和丰富的软件开发工具箱 SDK,实现了动态数据交换、模动动态链接、自动内存管理等功能。Microsoft Windows 3.0 是一个完善的软件开发环境,它采用了面向对象的程序设计技术,可以对开发任务进行自动处理,充分发挥了微型计算机的潜力。

Microsoft Windows 是当今最受欢迎的软件开发环境之一,许多著名的软件公司都在开发支持 Microfot Windows 的软件产品。Borland 公司率先推出 Turbo Pascal for Windows,它提供了面向对象的 ObjectWindows 应用程序开发工具包。利用这一语言工具,广大 Turbo Pascal 程序员可以方便、快速地开发出 Windows 应用程序。Turbo Pascal for Windows 的文档集包括:

《Turbo Pascal for Windows 用户指南》

该书阐述了如何安装、学习和使用 Turbo Pascal 的集成环境和命令行编译器。它还包括 Turbo Pascal 程序设计的基本知识,以及许多高级专题,如面向对象的程序设计和大型开发项目的管理等。

《Turbo Pascal for Windows 程序员指南》

该书是 Turbo Pascal 的技术参考手册,它详细描述了语言的定义,标准库的内容及其 Turbo Pascal 实现,以及 Turbo Pascal 与汇编语言的混合编程技术。该书还包含了所有编译指令和错误信息,并以字母顺序列出和描述了运行库中的所有标准过程与函数。

《Turbo Pascal for Windows—Windows 程序设计指南》(本书)

本书详细介绍了 Windows 的基本概念,着重讲述了如何利用 Turbo Pascal for Windows 所提供的 ObjectWindows 来开发 Windows 应用程序。

《Turbo Pascal for Windows—Windows 程序员参考手册》

该书描述了 Windows 接口中的所有常量、窗口风格、消息和 API 函数,以及 ObjectWindows 库中的所有对象。

《Turbo Pascal for Windows—WhiteWater 资源工具包》

该书讲述了利用 WhiteWater 资源工具包来编辑、编译、创建资源。

《Turbo Pascal for Windows—Turbo Debugger for Windows 用户指南》

该书详细讲述了如何利用 TDW 调试器调试 Windows 应用程序。

《Turbo Pascal for Windows—联机 Help 系统设计指南》

此书讲述了如何利用 Windows 的 Help 系统设计工具,为 Windows 应用程序开发良好的联机 Help 系统。

本套丛书终于和读者见面了。虽然我们进行了认真细致的消化、理解、翻译和校对等工作,但由于我们水平有限,加之时间仓促,缺点错误一定不少,恳请读者批评指正。最后,希望本套丛书能为广大程序员提供较大的帮助。

编译者于北京
一九九一年八月

目 录

第0章 引言

0.1 进入 Turbo Pascal for Windows	1
0.1.1 Turbo Pascal 语言	1
0.1.2 Windows 环境中的程序设计	1
0.1.3 资源编辑工具	1
0.1.4 ObjectWindows 库	1
0.1.5 Turbo Debugger for Windows	2
0.2 需要知道什么	2
0.3 本书的编排	2

第一部分 学习 ObjectWindows

第一章 继承窗口

1.1 什么是 Windows 应用程序	4
1.1.1 Windows 的优点	5
1.1.2 需求	5
1.2 编程工具	5
1.2.1 事件驱动结构	5
1.2.2 与设备无关的图形	5
1.2.3 多任务	6
1.2.4 内存管理	6
1.2.5 资源	7
1.2.6 动态连接	7
1.2.7 剪裁板	7
1.2.8 动态数据交换	7
1.2.9 多文档接口	7
1.3 Windows 数据类型	7
1.4 面向对象的窗口编程	8
1.4.1 一个更好的 Windows 接口	8
1.4.2 自动消息响应	9
1.5 Windows 程序的结构	9
1.5.1 Windows 的结构	10
1.5.2 和 Windows 及 DOS 的交互	10
1.6 “Hello, Windows”程序	10
1.6.1 应用程序开发周期	12

第二章 循序渐进地学习

2.1 第一步：一个简单的 Windows 应用程序	14
----------------------------------	----

2.1.1 应用程序需求.....	14
2.1.2 定义应用程序类型.....	14
2.2 主窗口对象类型.....	16
2.2.1 什么是窗口对象.....	16
2.2.2 创建一个新窗口类型.....	17
2.2.3 对消息的响应.....	17
2.2.4 终止应用程序.....	19
第三章 在窗口中填写内容	
3.1 什么是显示环境.....	22
3.2 在窗口中显示正文.....	22
3.2.1 消息记录.....	23
3.2.2 消除屏幕.....	23
3.3 第四步:在窗口中画线	24
3.3.1 拖动模型.....	24
3.3.2 对拖动消息的响应.....	25
3.4 第五步:改变线宽	27
3.4.1 跟踪线宽.....	27
3.4.2 运行输入对话.....	29
3.5 自动刷新图形.....	30
3.5.1 绘图模型.....	30
3.5.2 将图形作对象存储.....	31
3.5.3 重画储存的图形.....	33
第四章 加入一个菜单	
4.1 菜单资源.....	34
4.2 主窗口菜单.....	34
4.3 截获菜单消息.....	35
4.4 响应菜单消息.....	36
4.5 给可执行文件添加资源.....	37
第五章 保存对话	
5.1 第八步:加一个弹式窗口	38
5.1.1 创建和显示弹出式窗口.....	39
5.1.2 MakeWindow 函数	39
5.2 加一个对话框.....	40
5.2.1 加一个对象域.....	41
5.2.2 修改 Init 构子	41
5.2.3 运行对话框.....	42
5.3 第九步:将图形存入一个文件中	42
5.3.1 监视状态.....	42
5.3.2 保存和装载文件.....	44

第六章 弹出窗口

6.1 第十步:弹出一个帮助窗口	47
6.2 在 ObjectWindows 中使用单元	47
6.2.1 修改主程序.....	48
6.2.2 创建单元.....	48
6.3 给窗口加上控制.....	50
6.3.1 什么是控制.....	50
6.3.2 创建窗口控制.....	50
6.3.3 作为域的控制对象.....	51
6.3.4 管理控制.....	51
6.3.5 对控制事件的响应.....	52

第二部分 使用 ObjectWindows

第七章 ObjectWindows 概述

7.1 ObjectWindows 约定	55
7.2 ObjectWindows 的层次	55
7.2.1 基对象.....	55
7.2.2 TApplication	55
7.2.3 接口对象.....	55
7.2.4 窗口对象.....	56
7.2.5 对话对象.....	57
7.2.6 控制对象.....	57
7.2.7 MDI 对象	58
7.3 ObjectWindows 单元	58
7.3.1 WObjects	58
7.3.2 WinProcs	58
7.3.3 WinTypes	58
7.3.4 StdDlgs	58
7.4 Windows 资源	58
7.5 Windows 函数	59
7.5.1 ObjectWindows 调用 Windows 函数	59
7.5.2 对函数的存取.....	59
7.5.3 组合风格常量.....	60
7.5.4 Windows 函数的类型	60
7.5.5 回调函数.....	60
7.6 接收 Windows 消息	61
7.6.1 怎样将事件变成方法调用.....	61
7.6.2 Windows 消息的参数	62
7.6.3 Windows 消息类型	62

7.6.4	缺省消息处理	63
7.6.5	发送消息	63
7.6.6	消息范围	64
7.6.7	用户定义消息	64
第八章 应用程序对象		
8.1	控制应用程序流程	66
8.2	初始化应用程序	67
8.2.1	初始化主窗口	67
8.2.2	初始化每个应用程序实例	68
8.2.3	初始化第一个应用程序实例	69
8.3	运行应用程序	71
8.4	关闭应用程序	71
第九章 接口对象		
9.1	TWindowsObject 类型	73
9.2	为什么要有接口对象	73
9.3	窗口父子关系	74
9.3.1	子窗口链表	74
9.3.2	子窗口循环程序	75
9.4	消息处理	75
9.4.1	响应消息	76
9.4.2	命令和子窗口消息	77
第十章 窗口对象		
10.1	TWindow 类型	80
10.2	初始化和创建窗口对象	80
10.2.1	初始化对象	80
10.2.2	创建窗口元素	83
10.2.3	初始化和创建小结	83
10.3	窗口类的登录	83
10.3.1	登录属性	84
10.4	编辑窗口和文件窗口	86
10.4.1	编辑窗口	86
10.4.2	文件窗口	88
10.5	滚动窗口	89
10.5.1	流动器属性	89
10.5.2	给窗口加一个滚动器	90
10.5.3	滚动标例	90
10.5.4	自动滚动与跟踪	92
10.5.5	修改滚动单元和范围	92
10.5.6	修改滚动的位置	92

10.5.7 设置页大小	93
10.5.8 优化 Paint 方法	93
第十一章 对话对象	
11.1 使用对话资源	95
11.2 使用对话对象	95
11.2.1 构造和初始化对话	95
11.2.2 执行对话	95
11.2.3 结束对话	96
11.2.4 管理对话对象	96
11.2.5 控制操作和消息处理	97
11.2.6 连接控制对象	98
11.2.7 调用 DefWndProc	99
11.2.8 使用对话的扩展例子	100
11.2.9 对话窗口	100
11.3 输入对话	101
11.4 文件对话	102
11.4.1 初始化文件对话	102
11.4.2 运行文件对话	103
第十二章 控制对象	
12.1 TControl 对象类型	104
12.1.1 构造和创建控制对象	105
12.1.2 取消和释放控制	106
12.1.3 控制和消息处理	106
12.2 象对话的窗口	108
12.3 列表框控制	108
12.3.1 构造和创建列表框	108
12.3.2 修改列表框	109
12.3.3 询问列表框	110
12.3.4 从列表框中获取选择	110
12.3.5 示例程序 LBoxTest	111
12.4 静态控制	111
12.4.1 构造静态控制	112
12.4.2 询问静态控制	113
12.4.3 修改静态控制	113
12.4.4 StatTest 应用程序示例	113
12.5 按钮控制	114
12.5.1 响应按钮消息	114
12.6 检查框和无线按钮	115
12.6.1 构造检查框和无线按钮	115

12.6.2	询问选择框的状态	115
12.6.3	修改选择框的状态	116
12.7	组框	116
12.7.1	响应组框消息	116
12.7.2	BtnTest 示例程序	117
12.8	滚动条	117
12.8.1	构造滚动条	117
12.8.2	询问滚动条	119
12.8.3	修改滚动条	119
12.8.4	响应滚动条事件	119
12.8.5	SBarTest 示例程序	121
12.9	编辑控制	121
12.9.1	构造编辑控制	121
12.9.2	剪裁板和编辑操作	123
12.9.3	询问编辑控制	123
12.9.4	修改编辑控制	125
12.9.5	EditTest 示例程序	126
12.10	组合框	126
12.10.1	三种组合框	126
12.10.2	构造组合框	128
12.10.3	修改组合框	128
12.10.4	CBoxTest 示例程序	128
12.11	设置控制值	129
12.11.1	定义转换缓冲区	129
12.11.2	定义相应的对话或窗口	130
12.11.3	转换数据	131
12.11.4	支持用户控制的数据转换	132
12.11.5	转换示例	132
第十三章 MDI 对象		
13.1	MDI 应用程序的组成部分	133
13.1.1	每一个 MDI 窗口是一个对象	133
13.2	构造 MDI 窗口	134
13.2.1	构造 MDI 框架窗口	134
13.2.2	构造 MDI 子窗口	134
13.3	MDI 应用程序中的消息处理	135
13.4	管理 MDI 子窗口	135
13.4.1	子窗口的激活	136
13.4.2	子窗口菜单	136
13.5	MDI 应用程序示例	136

第三部分 ObjectWindows 高级专题

第十四章 内存管理

14.1 内存管理程序的使用	137
14.2 Windows 内存管理方法	137
14.2.1 句柄和地址	137
14.3 局部和全局内存	138
14.4 局部堆的使用	138
14.4.1 局部内存的分配和存取	139
14.4.2 局部内存块的释放和弃掉	140
14.4.3 局部内存块的再分配和修改	140
14.4.4 局部内存块的查询	141
14.4.5 程序设计的考虑	141
14.5 全局堆的使用	141
14.5.1 全局堆的分配和存取	142
14.5.2 锁存全局内存块的其它方法	143
14.5.3 全局内存块的释放和弃掉	143
14.5.4 全局内存块的再分配和修改	143
14.5.5 全局内存块的查询	144
14.5.6 全局可弃内存块的修改	145
14.5.7 内存较少时的警告消息	145
14.5.8 程序设计的考虑	145

第十五章 动态连接库

15.1 DLL 子程序的存取	146
15.2 DLL 例示	146

第十六章 动态数据交换

16.1 Windows 的写字板	152
16.1.1 写字板的格式	152
16.1.2 数据在写字板中的存放	152
16.1.3 从写字板中检索数据	154
16.1.4 数据处理的延迟	155
16.2 程序间的消息传送	155
16.3 动态数据交换	155
16.3.1 术语	156
16.3.2 会话的创建	156
16.3.3 会话的终止	158
16.3.4 交换数据的方法	158
16.3.5 单数据项的请求	158
16.3.6 数据传送	161

16.3.7	服务程序修改数据值	162
16.3.8	服务程序执行宏命令	164
16.3.9	系统主题	165
第十七章 图形设备接口 GDI		
17.1	显示环境	166
17.1.1	显示环境的管理	166
17.1.2	显示环境的内容	166
17.2	绘图工具	167
17.2.1	库存工具	167
17.2.2	逻辑工具	168
17.3	在窗口中显示图形	174
17.3.1	在窗口中绘图	174
17.3.2	画窗口	174
17.3.3	绘图方法	175
17.3.4	绘图工具的使用	175
17.4	GDI 绘图函数	176
17.4.1	画正文的函数	177
17.4.2	画线函数	177
17.4.3	画形状的函数	179
17.4.4	调色板的使用	182
第十八章 资源使用		
18.1	资源创建	184
18.2	将资源加到可执行文件中	184
18.3	资源编译器的运行	185
18.3.1	资源编译器的说明文件	186
18.3.2	资源编译器指南	189
18.4	将资源加载到应用程序中	189
18.4.1	加载菜单	189
18.4.2	加载加速键表	190
18.4.3	加载对话框	191
18.4.4	加载光标和图标	191
18.4.5	加载字符串资源	192
18.4.6	加载位图	193
18.4.7	用位图创建画刷	193
18.4.8	在菜单中显示位图	194
第十九章 标准应用程序指南		
19.1	设计原则	196
19.1.1	给用户响应信息	196
19.1.2	让用户控制程序流	196

19.2 标准外观与行为.....	196
19.2.1 一般视面.....	197
19.2.2 鼠标与键盘的交互.....	197
19.2.3 菜单.....	197
19.2.4 对话框.....	199
19.2.5 设计考虑.....	199
19.3 编写安全的程序.....	199
19.3.1 主要消费对象	202

第四部分 集和流

第二十章 集

20.1 集.....	204
20.1.1 可动态地设置集的大小.....	204
20.1.2 有多形的集.....	204
20.1.3 类型检查和集.....	204
20.2 集的创建.....	205
20.3 循环程序.....	205
20.3.1 ForEach 循环	206
20.3.2 FirstThat 和 LastThat 循环	207
20.4 有序集.....	208
20.5 字符串集.....	209
20.5.1 再研究循环过程.....	210
20.6 多形集.....	210
20.7 集和内存管理.....	212

第二十一 流

21.1 对象的输入/输出问题	213
21.2 流.....	213
21.2.1 流是不规则的.....	213
21.2.2 流对对象的处理.....	213
21.3 流的核心用途.....	214
21.3.1. 建立流.....	214
21.3.2 流的读、写	214
21.3.3 流的关闭.....	215
21.4 对象与流.....	215
21.4.1 加载和存贮方法.....	215
21.4.2 流的登记.....	216
21.4.3 登记方法.....	217
21.4.4 标准对象的登记.....	217
21.5 流的机制.....	217

21.5.1	Put 对象的过程	217
21.5.2	Get 对象的过程	217
21.5.3	空对象指针的处理	218
21.6	流集的例子	218
21.7	进行存贮操作的主体	221
21.7.1	流中的域段	221
21.7.2	同类窗口实例	222
21.8	流的拷贝	222
21.9	随机存取的流	223
21.10	流中的非对象	223
21.11	设计自己的流	223
21.11.1	流错误的处理	223

第0章 引言

Turbo Pascal for Windows 为开发 Microsoft Windows 应用程序提供了一种令人激动的新方法。迄今为止,大多数 Windows 的程序设计使用 C 语言,而且使用了许多在 DOS 下运行的开发工具。其结果是使开发 Windows 程序的过程变得冗长、复杂和混乱。

0.1 进入 Turbo Pascal for Windows

现在,为了祛除开发 Windows 应用程序的痛苦,Turbo Pascal for Windows 作了几项新的重大改革。

- ☐ Pascal 语言的扩展
- ☐ 一个 Windows 环境
- ☐ Windows 交互资源编辑工具
- ☐ 方便 Windows 编程的 ObjectWindows 库
- ☐ Windows 的 Turbo Debugger

以下是它们特点。

0.1.1 Turbo Pascal 语言

如果您已经开发过 Windows 应用程序,那么你一定使用过 C 语言,但有时并不希望用 C 语言编程,或许已需要使用 Windows 的 Pascal 代码。Turbo Pascal 现在具备开发 Windows 应用程序的所有能力。

除了 Turbo Pascal 所有特性(如快速编译、可重复使用单元和面向对象的程序设计)之外,Turbo Pascal for Windows 还支持使用和创建动态连接库(DLLs)、直接包含 Windows 资源文件、若干种新的数据类型及修改后 Windows 标准 DOS Turbo Pascal 单元的版本。

0.1.2 Windows 环境中的 Windows 程序设计

也许在编写 Turbo Pascal Windows 程序时首先注意到的是集成开发环境(IDE)作为一个 Windows 程序在 Windows 中运行。这说明在编码、编译、和测试 Turbo Pascal 应用程序时可以充分使用 Windows 的多任务能力。

0.1.3 资源编辑工具

Turbo Pascal for Windows 包括 Microsoft 资源编译器和一组完整的资源编辑器。资源编辑器可以直观地编辑菜单、对话、位象、图标和光标这样的资源,不用写程序编辑。资源编译器可编译别的 Windows 程序员编写的资源程序。

0.1.4 ObjectWindows 库

Windows 引入了许多以前难以想象的好方法,如在一个大小可变窗口中处理正文和图形、在一个多任务环境中和别的程序交互,以及在 Windows API 中处理近 600 个函数。也许最难的部分是列举出应用程序在 Windows 中运行时要做哪些基本工作,然后确信做完了这些工作。

使应用程序成为 Windows 程序需要做大量的工作。例如,不能直接写屏幕,也不能直接修改内存。另外,Windows 应用程序必须知道怎样响应报告消息的发送,这些送到应用程序中的消息来源于 Windows 对用户事件(如选择某菜单项)的响应。

但您不必做这么多的工作,我们给您一个开头: ObjectWindows。

ObjectWindows 是一个面向对象的库,它封装了每一个窗口必须知道的大多数行为,您可以继承所有东西,而不是在每开始一个新程序时重新开发。通过提供一个稳定和坚实的框架,可以使您能把精力集中在需要编写的程序部分,而不是集中在所有 Windows 应用程序共同的部分。

0.1.5 Turbo Debugger for Windows

Windows 的 Turbo Debugger 是一个供 Turbo Pascal for Windows 使用的强有力的源代码级调试器。Windows 的 Turbo Debugger 完全支持对 Turbo pascal 和汇编表达式的评估,并具有调试 Windows 应用程序和面向对象程序的特殊能力。

使用该调试器的所有细节见《Turbo Debugger for Windows 用户指南》。

0.2 需要知道什么

在开始编写 Windows 应用程序之前要知道以下几件事情。首先要知道怎样使用 Turbo Pascal 和 Windows。用 Turbo Pascal 编程的知识可以在 Turbo Pascal 的用户手册和编程指南中找到,使用 Windows 的细节可见 Windows 软件的资料。

另外,为了使用 ObjectWindows,必须熟悉面向对象的程序设计。用 ObjectWindows 编写的应用程序广泛地使用了面向对象的技术,包括继承和多态性。这些主题在该用户指南中的第四章“面向对象的程序设计”中论述。

除面向对象的技术外,还要熟悉指针和动态变量的使用,因为几乎所有 ObjectWindows 的对象实例是在堆上动态地分配的。

0.3 本书的编排

因为 Turbo Pascal for Windows 是 Windows 程序设计的一种新方法,并且 ObjectWindows 使用了一些也许您并不熟悉的技术,所以本手册包含了大量的解释说明材料。ObjectWindows 和 Windows 应用程序设计接口的完整参考材料见《Windows 参考手册》

本手册分为四个部分:

□ 第一部分,“学习 ObjectWindows”,向您介绍用 Turbo Pascal 编写 Windows 应用程序的原理,其中包括走过编写和扩展 ObjectWindows 应用程序过程的指导材料。

□ 第二部分,“使用 ObjectWindows”,给出了 ObjectWindows 本身元素的细节,其中包括对象层次的便览、它怎样和 Windows 环境交互及层次和怎样使用层次的详细说明。

□ 第三部分,“高级 ObjectWindows”,讨论更高级 Windows 程序设计的重要主题,尤其是必须和 Windows 环境直接交互的部分,其中包括图形、资源的使用、内存管理、共享代码和数据,以及 Windows 编程的准则。

□ 第四部分,“收集和流”,描述对于 Turbo Pascal 对象怎样使用收集和流机制,尤其着重收集和流在 ObjectWindows 中的使用。

第一部分 学习 ObjectWindows

第一章 继承窗口

本章是用 Turbo Pascal 编写 Microsoft Windows 程序的概述,重点在于面向对象的程序设计。本章中的例子使用了和 Turbo Pascal 一起提供的 ObjectWindows。您也将会学到什么进入 Windows 应用程序及 Windows 应用程序得出什么结果。您将看到 Windows 应用程序的所需行为、ObjectWindows 面向对象程序设计怎样使许多任务自动地完成及怎样简化某些任务。

为了从本章中学到最多的东西,需要理解面向对象程序设计的概念。如果面向对象的程序设计对于您来说是新的东西,请学习本用户指南中的第四章“面向对象的程序设计”,同时您还必须知道怎样使用 Windows。

1.1 什么是 Windows 应用程序

图 1.1 表明了 Windows 应用程序的主要组成部分。本章内容依赖于对这些组成部分的

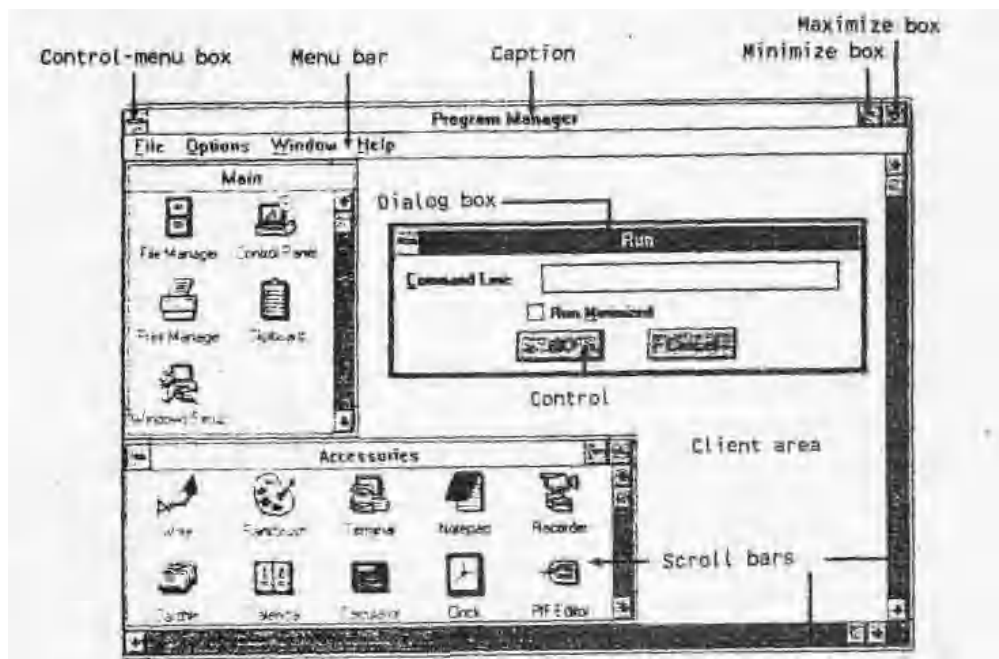


图 1.1 Windows 应用程序的屏幕组成部分

了解。

Windows 应用程序是一种特殊类型的 PC 程序,它

- ☐ 必须按一个特殊的可执行文件格式存在。
- ☐ 只和 Windows 一起运行。
- ☐ 一般在屏幕上的一个矩形窗口中运行。
- ☐ 遵照用户接口准则以标准的方式出现和执行。
- ☐ 可以和别的窗口和非窗口应用程序同时运行,其中包括它自己的实例。
- ☐ 可和别的 Windows 应用程序通讯及共享数据。