

Windows程序设计

——用 Microsoft C/C++
和 Visual C++

章生立 董三立等 编

北京航空航天大学出版社

(京)新登字 166 号

图书在版编目(CIP)数据

Windows 程序设计:用 Microsoft C/C++和 Visual C++/
章生立,董三立编著. -北京:北京航空航天大学出版社,
1995.5
ISBN 7-81012-569-9

I. W... I. ①章... ②董... III. 操作系统, Windows-程
序设计 IV. TP316

中国版本图书馆 CIP 数据核字 (95) 第 02555 号

内 容 简 介

本书介绍利用 Microsoft C/C++和 Visual C++进行 Windows 程序设计的技术和参考信息。首先介绍 Windows 的历史,Windows 的概念和术语,如何控制 Windows 环境,用 C 和 C++编写简单的应用程序;然后讨论如何控制 Windows 窗口,如何设计图标、光标和位图,如何开发菜单和键加速器,如何编写使用对话框和字体的代码,如何绘制图形包括科学图、数学图、饼形图、条形图和线图,如何设计多媒体的程序;最后阐述使用 Microsoft 基础类库(MFC)开发 Windows 应用程序的基本概念和技巧;书后列表了 Windows API,Microsoft 基础类库参考和 Visual Workbench 参考与 WINDIWS.H 头文件参考。

供青年学生、初级程序员和中级程序员阅读使用。

- 书 名:Windows 程序设计——用 Microsoft C/C++和 Visual C++
Windows CHENGXU SHEJI ——YONG Microsoft C/C++ HE Visual C++
- 编 著 者:章生立 董三立等 编
- 责任编辑:王小青
- 出 版 者:北京航空航天大学出版社
- 地 址:北京市海淀区学院路 37 号(100083)
- 印 刷 者:朝阳科普印刷厂印刷
- 发 行:新华书店总店科技发行所
- 经 售:全国各地书店
- 开 本:787×1092 1/16
- 印 张:43.5
- 字 数:1108 千字
- 印 数:5000 册
- 版 次:1995 年 5 月第 1 版
- 印 次:1995 年 5 月第 1 次印刷
- 书 号:ISBN 7-81012-569-9/TP·158
- 定 价:49.00 元



Microsoft C/C++ 是 Visual C++ 的前身,从 Microsoft C/C++ 7.0 版开始,Microsoft C/C++ 演化为 Visual C++ 1.0,后来又推出了 Visual C++ 1.5 和 2.0。Visual C++ 非常庞大,而 Microsoft C/C++ 相对较小,很实用。

从 Visual C++ 1.0 开始,Visual C++ 的集成环境 Visual Workbench 运行于 Windows,是一个很好用的 Windows 应用程序开发环境,要特别指出的是,在新的 IDE 中可在 Windows 中直接调试程序。

在 Visual C++ 1.0~2.0 中,配备了 CodeView for Windows 用来调试 Visual C++ 设计的 Windows 应用程序,增强了包括图标、光标、位图、对话框、菜单、热键和字体等的资源设计。特别是在 Visual C++ 2.0 中,资源设计已集成到 Visual C++ 的 Workbench 中,可用 Resource 菜单访问,用来设计 Windows 应用程序的界面。Visual C++ 在 MFC 中封装了 Windows API,包括函数、数据结构和消息等,2.0 版支持在 Windows 3.1、Windows NT 3.5 和 Windows 95 操作系统下开发应用程序,方便 C/C++ 读者开发 Windows 应用程序。

本书适用于 Microsoft C/C++、Visual C++ 以及各种演化版本。

本书中的例子针对国内程序员的需要,全部采用中文的菜单、中文对话框和中文提示界面,每个程序都有很好的注释和解说,方便理解和学习。

目 录

编者前言

第一章 Windows 入门	1
1.1 什么是 Windows	1
1.2 历史展望	1
1.2.1 从 BIOS 10H 软件到 Windows 软件	2
1.3 Windows 是什么	3
1.3.1 标准用户界面	3
1.3.2 多任务	5
1.3.3 内存管理	5
1.3.4 队列输入	6
1.3.5 消息	7
1.3.6 设备的独立性	7
1.3.7 动态连接库	8
1.3.8 “新可执行”格式	9
1.3.9 MS-DOS 的应用程序	10
1.4 对窗口的增强	10
1.5 必备的条件	10
1.5.1 Windows 工具	11
第二章 Windows 的概念和术语	12
2.1 什么是窗口	12
2.2 可视界面	12
2.2.1 边界	12
2.2.2 标题条	13
2.2.3 控制框	13
2.2.4 系统菜单	13
2.2.5 最小化框	13
2.2.6 最大化框	14
2.2.7 垂直滚动条	14
2.2.8 水平滚动条	14
2.2.9 菜单条	14
2.2.10 客户区	14
2.3 Windows 类	14
2.4 面向对象式程序设计	15

2.4.1	图标	15
2.4.2	光标	16
2.4.3	插入记号	16
2.4.4	消息框	16
2.4.5	对话框	16
2.4.6	字体	16
2.4.7	位图	18
2.4.8	画笔	18
2.4.9	画刷	18
2.5	获取消息	18
2.5.1	消息格式	19
2.5.2	消息的来源	20
2.5.3	消息的处理	20
2.5.4	消息循环	21
2.6	什么是资源	21
2.7	访问 Windows 函数	22
2.7.1	PASCAL 调用规程	22
2.8	什么是 WINDOWS. H	22
2.9	WINDOWS 的标识	23
2.10	Windows 应用程序的组成	24
2.10.1	C 编译器	25
2.10.2	资源编辑器	25
2.10.3	资源编译器	25
2.10.4	连接器	25
2.10.5	NMAKE 工具	25
第三章	访问 Windows 环境	26
3.1	坐标系统	26
3.1.1	8 种映像模式	26
3.1.2	设备坐标	27
3.1.3	视口	27
3.1.4	MM_ISOTROPIC 和 MM_ANISOTROPIC 的说明	27
3.1.5	改变缺省坐标	28
3.2	选择初始窗口大小、位置、光标、图标和风格	28
3.3	ShowWindow 函数	37
3.4	SetClassWord 函数	38
3.5	虚拟键	39
3.6	控制和对话框	41
3.6.1	静态控制	41

3.6.2	按式按钮控制	41
3.6.3	单选按钮控制	41
3.6.4	复选框控制	41
3.6.5	编辑框	42
3.6.6	列表框	42
3.6.7	滚动条	42
3.7	系统计时器	43
3.7.1	使用定时器	43
3.8	内存	43
3.8.1	内存分配	44
3.8.2	内存管理	44
第四章	编写简单的 Windows 程序	47
4.1	入门	47
4.1.1	句柄的进一步说明	47
4.2	Windows 应用程序的基本组成	48
4.2.1	WinMain 函数	48
4.2.2	注册窗口类	48
4.2.3	创建窗口	52
4.2.4	显示及更新窗口	52
4.2.5	消息循环	53
4.3	窗口函数	53
4.3.1	WM_PAINT 消息	55
4.4	创建模块定义文件	56
4.5	创建 NMAKE 文件	57
4.6	联编	58
4.7	怎样使用 SWA 创建其它的 Windows 应用程序	59
4.8	创建 Include 文件	61
4.9	资源文件	62
4.9.1	资源编译器	64
4.10	MAKE 的回顾	65
第五章	控制 Windows 窗口	66
5.1	什么是滚动条	66
5.1.1	向下的含义	66
5.1.2	滚动条范围	66
5.1.3	滚动条位置	66
5.1.4	滚动条类型	67
5.2	怎样使用滚动条编写应用程序	67

5.2.1	Make 文件(SCROL)	72
5.2.2	定义文件(SCROL.DEF)	72
5.2.3	文档文件(SCROL.DOC)	72
5.2.4	应用程序文件(SCROL.CPP)	72
5.3	如何使用系统计时器	77
5.3.1	Make 文件(TICK)	81
5.3.2	定义文件(TICK.DEF)	81
5.3.3	消息文件(TICK.DOC)	81
5.3.4	应用程序文件(TICK.CPP)	81
5.4	如何创建一个抵押偿还表	84
5.4.1	LOAN Make 文件	90
5.4.2	LOAN.DEF 模块定义文件	90
5.4.3	LOAN.CPP 应用程序文件	90
第六章	设计图标、光标和位图	93
6.1	使用 Windows 工具创建图标、光标和位图	93
6.1.1	Microsoft Image Editor 的启动	93
6.2	使用图标	99
6.2.1	Make 文件(ICON)	103
6.2.2	定义文件(ICON.DEF)	103
6.2.3	资源文件(ICON.RC)	103
6.2.4	应用程序文件(ICON.CPP)	103
6.3	使用光标	103
6.3.1	Make 文件及定义文件(CUR 及 CUR.DEF)	106
6.3.2	资源文件(CUR.RC)	106
6.3.3	应用程序文件(CUR.CPP)	106
6.4	使用 Microsoft NMAKE	106
6.4.1	NMAKE Make 文件的创建	106
6.4.2	NMAKE 选项	107
6.4.3	NMAKE 的调用	108
6.5	使用资源编译器	108
6.5.1	资源编译器语句	108
6.5.2	使用资源编译器编译	110
第七章	开发菜单和键加速器	112
7.1	菜单机制	112
7.1.1	什么是菜单	112
7.1.2	菜单关键字和选项	113
7.2	随时创建菜单	117

7.2.1	创建菜单来改变图形的尺寸	117
7.2.2	用菜单改变背景颜色	122
7.2.3	用菜单决定系统信息	130
7.2.4	用菜单查看目录表列	138
7.3	总结	146
第八章	数据输入对话框	147
8.1	对话框简介	150
8.2	对话框编辑器	151
8.2.1	为何要使用对话框编辑器	151
8.2.2	使用 Microsoft 对话框编辑器	151
8.2.3	使用编辑器进行编辑	154
8.2.4	创建一个对话框	155
8.2.5	检查.DLG 文件	156
8.3	为各种需要创建对话框	157
8.3.1	创建一个简单的 About 对话框	157
8.3.2	用一个对话框改变图形形状	164
8.3.3	通过对话框输入文本	174
8.3.4	用对话框输入整型数	183
8.3.5	用对话框输入实型的数字	191
8.4	创建消息框	200
第九章	字体的使用	207
9.1	字体的结构和定义	207
9.1.1	逻辑字体常数	207
9.1.2	TEXTMETRIC 结构	209
9.1.3	LOGFONT 结构	210
9.1.4	字符单元	210
9.2	字体附加的特性	212
9.2.1	字体宽度	212
9.2.2	自动引导和压缩字距磅	212
9.2.3	OEM 与 ANSI 字符集相比较	212
9.2.4	逻辑字体与物理字体	214
9.2.5	向量字体与光栅字体	214
9.2.6	生成字体	214
9.3	字体的类型	214
9.3.1	缺省字体	214
9.3.2	打印机字体和显示字体	214
9.3.3	Custom 字体	215

9.4 字体映射方法	215
9.5 字体编辑器的使用	215
9.5.1 如何装入字体编辑器	216
9.5.2 基本的字体编辑器窗口	216
9.5.3 如何改变字体头	218
9.5.4 如何定制字体	219
9.5.5 如何存储 Custom 字体设计	220
9.5.6 如何生成一个字体源文件	220
9.6 字体程序	221
9.6.1 CreateFont 函数	221
9.6.2 CreateFontIndirect 函数	222
9.6.3 FONT1 程序	222
9.6.4 FONT1 文件	225
9.6.5 FONT1.DEF 文件	225
9.6.6 FONT1.CPP 文件	225
9.6.7 FONT2 程序	226
9.6.8 FONT2 和 FONT2.DEF 文件	230
9.6.9 FONT2.CPP 文件	231
9.6.10 FONT3 程序	231
9.6.11 FONT3 及 FONT3.DEF 文件	234
9.6.12 FONT3.CPP 文件	234
9.6.13 FONT4 程序	235
9.6.14 FONT4 及 FONT4.DEF 文件	238
9.6.15 FONT4.CPP 文件	238
9.7 更好的字体	239

第十章 图形概念和绘图元语..... 240

10.1 图形设备接口.....	240
10.1.1 GDI 的用途.....	240
10.1.2 像素操作.....	240
10.1.3 设备信息.....	240
10.1.4 设备场境句柄.....	244
10.1.5 映射方式.....	244
10.2 GDI 绘图元语.....	245
10.2.1 图形元语.....	245
10.2.2 GDI 绘图元语的使用.....	251
10.2.3 简单的条形图.....	255
10.3 GDI 工具.....	259
10.3.1 画笔.....	259

10.3.2	画刷	260
10.3.3	简单的条形图再述	261
10.3.4	颜色	265
10.3.5	位图	274
第十一章	科学图和数学图的绘制	279
11.1	正弦波	279
11.2	衰减正弦波	283
11.3	一个傅立叶级数	289
第十二章	饼形图、条形图和线图的设计	299
12.1	调色板管理程序	299
12.1.1	逻辑调色板的开销	299
12.2	饼形图	301
12.2.1	PIE 文件和 PIE.DEF 文件	309
12.2.2	PIE.H 头文件	309
12.2.3	PIE.RC 资源文件	310
12.2.4	PIE.CPP 程序	310
12.3	条形图	313
12.3.1	BAR 文件和 BAR.DEF 文件	325
12.3.2	BAR.H 头文件	325
12.3.3	BAR.RC 资源文件	325
12.3.4	BAR.CPP 程序	326
12.4	线图	328
12.4.1	LINE 文件和 LINE.DEF 文件	341
12.4.2	LINE.H 头文件	342
12.4.3	LINE.RC 资源文件	342
12.4.4	LINE.CPP 程序	342
12.5	关于三大程序的变化	346
第十三章	专用应用程序:带多媒体声音的草图、动画和屏幕保存程序	347
13.1	草图绘制:带多媒体声音的鼠标速写	347
13.1.1	SKCH 文件	356
13.1.2	SKCH.CPP 代码	356
13.2	动画:火车 1	357
13.3	动画:带多媒体声音的火车 2	363
13.4	屏幕保存程序:一个带多媒体声音的屏幕保存程序	369
13.4.1	SAVER 文件	370
13.4.2	SAVER 应用程序代码	376

13.5 重要的是实践.....	377
第十四章 为面向对象窗口程序开发,MICROSOFT 基础类库简述	378
14.1 类库合理性.....	378
14.2 考虑 MICROSOFT 基础类库的设计	379
14.3 重要的 Microsoft 基础类库特性	379
14.4 OBJECT;全部类库应用程序的基础	380
14.5 重要的 Microsoft 基础类库类	382
14.6 简单的基础类例子.....	382
14.6.1 使用 EASY.CPP 建立一个窗口	383
14.6.2 AFXWIN.H 头文件	384
14.6.3 CWinApp;派生的 CTheApp 类	384
14.6.4 CFrameWnd;应用程序窗口	386
14.6.5 InitInstance;使用一个成员函数	387
14.6.6 构造函数.....	387
14.6.7 测试 EASY 应用程序	387
14.7 一个简单的 Microsoft 基础类库应用程序	387
14.7.1 MFCA Make 文件说明	390
14.7.2 MFCA.DEF 模块定义文件说明	391
14.7.3 MFCA.H 头文件说明	391
14.7.4 MFCA.CPP 说明	392
14.7.5 MFCA 应用程序说明	393
14.8 图形元语.....	393
14.8.1 GDI Make 文件、GDI.DEF 模块定义文件和 GDI.H 头文件	398
14.8.2 GDI.CPP 说明	398
14.8.3 GDI 应用程序说明.....	399
第十五章 用资源开发 Microsoft C++ 基础类库应用程序	401
15.1 给科学图表添加资源:菜单、对话框和多媒体声音.....	401
15.1.1 FOUR Make 文件和 FOUR.DEF 模块定义文件	409
15.1.2 FOUR.H 头文件	409
15.1.3 FOURR.H 资源头文件,FOUR.RC 资源脚本文件和 FOUR.DLG 对话脚本文件.....	410
15.1.4 FOUR.CPP 应用程序文件	412
15.1.5 创建定制的 CMainWnd 类	412
15.1.6 获得 Windows 的当前大小	413
15.1.7 绘制傅立叶波形.....	413
15.1.8 创建“关于”对话框.....	415
15.1.9 数据输入对话框.....	415

15.1.10	响应 OnExit	416
15.1.11	消息映射处理	416
15.1.12	运行 FOUR 应用程序	416
15.2	给条形图添加资源：一个菜单和一个对话框	418
15.2.1	BAR 命令行 Make 文件, BAR. MAK 工程 Make 文件, BAR. DEF 模块定义文件	432
15.2.2	BAR. H 头文件	432
15.2.3	BARR. H 资源头文件, BAR. RC 资源脚本文件, BAR. DLG 对话脚本文件	432
15.2.4	BAR. CPP 应用程序文件	433
15.2.5	测试 BAR 应用程序	438
附录 A	Windows API 函数和 Microsoft 基础类库项目	440
附录 B	Visual C++ 集成环境用法	517
B.1	进入 Visual Workbench 环境	517
B.1.1	主菜单分类	517
B.1.2	书写新的应用程序	529
B.1.3	编辑应用程序	531
B.1.4	拷贝、剪切和粘贴(Copying ,Cutting 和 Pasting)	532
B.2	编译和连接	534
B.2.1	在 Visual Workbench 内工作	534
B.2.2	在 DOS 命令行下工作	535
B.3	使用 Microsoft 帮助系统	536
B.3.1	程序语句帮助	536
B.3.2	其他帮助特性	536
B.3.3	打印帮助信息	538
B.4	设置重要的 Options 选项	538
B.4.1	Project... (工程选项)	538
B.4.2	调试选项	543
B.4.3	目录选项	543
B.4.4	编辑器选项	545
B.4.5	Workspace 选项	545
B.4.6	工具选项	547
B.4.7	颜色选项	548
B.4.8	字体选项	550
B.4.9	TOOLS. INI 文件	550
附录 C	WINDOWS. H 头文件	552

第一章 Windows 入门

Microsoft Windows 是 Microsoft 基于图形的优秀操作环境。它将单击式控制、弹出式菜单、专为运行 Windows 编写的应用程序的能力及在 DOS 环境下运行标准应用程序的能力结合在一起。

1.1 什么是 Windows

Microsoft Windows 是在 MS-DOS 下运行的一个操作环境。它是一个以图形为基础、具有多任务的窗口环境。所有专门为 Windows 编写的程序都有一致的形式和命令结构,以使新的 Windows 应用程序较易掌握。

对于应用程序的开发,Windows 提供了大量的内建子程序,使你很容易实现弹出式菜单、滚动条、对话框、图标以及许多其它对用户友好的图形界面。刚开始使用 Windows 3.0 时,用户可使用新的对话控制、菜单类型及“拥有者绘图控制”。通过使用 Windows 提供的扩展图形编程接口,能容易地将应用程序格式化,并以各种字体和间距输出文本。

Windows 也允许应用程序的设计人员以一种与设备无关的方式对待视频显示器、键盘、鼠标、打印机、串行口以及系统计时器。这可以使一个应用程序在不同硬件配置的情况下正常运行。

随着 Windows NT(New Technology)32 位操作系统的发行,Windows NT 的用户将能利用完全占先的、基于线程的多任务的优势。这个新操作系统设计得可移植到优于 80386 和 80486 的微处理器,比如 RISC 结构。

Windows NT 是高级计算机环境(ACE)的一个关键部件,ACE 的设计可为微处理系统提供一个开放的、标准的计算机环境。Windows NT 能在两个方面提供帮助:386/486 的 PC 以及 MIPS RISC 为基础的系统。

1.2 历史展望

在十九世纪四十年代,计算机使用的是硬件拷贝设备(按现在的标准它是过时的),要求用户对大量的输出结果进行检查。为了调查飞机的稳定性和控制性,于 1950 年在 MIT(麻省理工学院)建立的一套系统,计算机最早使用了阴极射线管(CRT),缩短了用户输入和计算机输出的时间,这就促使现代公共显示设备与计算机相连接。

十九世纪五十年代飞机的发展也促进了 SAGE 防空系统的发展,它将雷达的可视信号转换为计算机产生的图像。这个系统也是第一次利用光笔在显示屏上选择符号。

十九世纪六十年代早期,一个 MIT Ph.D 的候选人开发了草图直线绘制系统。这个系统可使用户通过对屏幕上光笔的选择来绘制草图。该绘图系统能画出直线或结构封闭的图形,因此简化了表达简单物体的复杂图形设计。早期的 CRT 能在显示屏上任两点间画出一条直线。然而,因为图像容易很快消失,所以它不得不每秒重复绘画几次。在十九世纪六十

年代,这需要一个用存储直线末端点的内存及快速重画直线的硬件——这两样都非常昂贵。例如,在1965年,IBM为这种图形显示引进了第一批大量生产的CRT,单买显示器就用了将近\$10万,价格昂贵是这段时间内没有更多的人安装该设备的原因。三年以后,Tektronix开发了第一批CRT存储管,这种类型的CRT能保存草图直至用户不再需要为止。由于设计了该显示管,昂贵的内存和重画图硬件就被淘汰了,从而将显示器花费降到了\$1.5万。凭这个价钱,Tektronix显示器立即成功了。

十九世纪七十年代,由于在内存和硬件逻辑单元花费上的急剧降低,促进了图形开发环境进一步发展。这种开发增强了扩展内存光栅扫描显示器,可产生逼真的阴影和色彩图案。

到了十九世纪八十年代末,显示监视器不再是数字量。IBM视频图形数组(VGA)的输出为模拟量,它保持了与先前应用程序环境的兼容性,以前的所有视频方式都受到支持。单色或有色图形适配器(CGA)和增强型图形适配器(EGA)的模式都可通过VGA适配器复制。例如,一个模拟的单色显示器与VGA相连,那么彩色就会转变为灰色。此外,用VGA还可得到下面几个新的模式:

- 640×480 图形(2种色彩和16种色彩)
- 720×400 文本(单色和彩色)
- 360×400 文本(16种色彩)
- 320×200 图形(256种色彩)

以后还产生了XGA和SVGA监视器——具有更高的分辨率和更大的彩色调色板的监视器;并且,利用这些产品,用户不需要再改变已编写的Windows程序的源代码。

1.2.1 从 BIOS 10H 软件到 Windows 软件

为了解图形软件的发展历史,我们先介绍一下BIOS(基本输入/输出)子程序。每个IBM PC,PS/2和兼容机上都固化有一套BIOS子程序,它们存储在ROM(只读存储器)上。这些存储的子程序给具有标准特征的硬件提供了一个界面,包括时钟、键盘、软盘和硬盘及Windows的视频系统相关的硬件。

视频BIOS程序由一套可执行基本视频程序工作的简单工具组成,象往荧幕写入字符串、清除字符和改换颜色等等。

1.2.1.1 中断 10H

过去为实现实时图形,程序员不得不编写一个汇编语言程序来访问同样也用汇编语言编写的BIOS程序。为访问BIOS的视频部分,使用8088处理器的程序员不得不发出中断10H命令。

现在ROM BIOS支持几种视频I/O函数,每种函数都通过执行一个中断10H来访问。这些函数被编了码;在执行一个中断10H前,程序员必须在相应的寄存器如ah中设置需要的子功能代码。

当执行中断时,微处理机上的其它寄存器可能保存有要送到BIOS程序去的其他参数。如果中断10H函数要调用返回程序的数据,那么它通过将数据留在一个或多个微处理器寄存器中就可以实现。不过,基于寄存器的参数传递主要用在汇编语言程序中。

1.2.1.2 高级语言

说到软件,读者可能对计算机的初步体验是使用BASIC命令:LINE,CIRCLE,COLOR

等;然后读者可能还学过其他语言如 Pascal,它可以提高程序结构性,但不能提高图形能力。随着 Borland Turbo Pascal 的推出,程序员有了一个相当丰富的绘图环境。从 Turbo Pascal 4.0 开始,象视频器、剪裁、用户定义填充模式和立体条等许多复杂功能都能实现。

另一种受人欢迎的开发语言是 C/C++,它在两个领域中实现得很好——汇编语言和高级语言的结合。C/C++提供给程序员汇编语言的硬件存取能力以及高级语言的库逻辑结构特征。C/C++程序员在开发专用于图形的应用程序时,直接利用高级图形库子程序。

前面的讨论忽略了对并行图形应用程序的讨论。到目前为止所写的每个程序都使用了整个视频系统,包括各种寄存器、内存及显示装置等。

随着用户对软、硬件期望的提高,对能同时运行的程序的需求量也增大。用户不再满足于他们编辑字符前等待数据库完成排序——他们希望同时做好这两项工作。这种需求在产品上帮助用户。象 Quarterdeck 公司的 DESQ View 和 Microsoft Windows——这些产品使用户能完成多任务。更为重要的是 Windows 提供了图形设备界面(GDI)。

利用 Windows 的 GDI 函数,程序员能编出用户熟悉的应用程序(用菜单、滚动条和信息框等)。此外,当其他程序正在运行的同时,能对应用程序重设尺寸、移动、图标化、转进后台等。最为重要的是,GDI 函数允许一种应用程序与另一种应用程序进行通讯,尽管大多数应用程序还不能充分利用这种特性。

对软、硬件发展的短暂历史回顾解释了两者变化迅速的程度。用户和应用程序开发者今天所需要的是一个与面向图形的用户界面、多任务能力、以及硬件独立性。在这些方面,Windows 已经相当成熟了。

1.3 Windows 是什么

和传统的 MS-DOS 环境相比,Windows 操作环境给用户和程序员带来了相当多的好处。三个主要性能(面向图形的用户界面、多任务以及硬件独立性)已明显不新颖了,需要的是将这三种能力结合到单一的微机操作环境中。

1.3.1 标准用户界面

在 Windows 提供的三种主要特性中,标准化的与面向图形的用户界面是最重要的,当然对用户也是最重要的。一般的用户界面使用图形或图标来表示驱动器、文件、子目录和多个操作系统命令及功能。图 1.1 表示一个典型的 Windows“文件管理器”的窗口。程序由标题条来识别,许多基本的文件操作功能通过用鼠标单击菜单来访问。大多数 Windows 程序既有键盘界面又有鼠标界面。尽管大多数 Windows 程序的功能能通过键来控制,但使用鼠标在很多方面更为方便些。

因为所有程序都有相似的“外表”,所以用户不需要再花费很多时间去学习使用一个新的应用程序。图 1.2(Windows“画笔”的屏幕)和图 1.3(Windows“书写器”的屏幕)说明了这个相似性,注意公共的“文件”和“编辑”选项及“最大化”/“最小化”按钮。对程序员来说,一致的用户界面通过使用直接固化在 Windows 中的子程序来实现菜单和对话框。所有菜单有相同的键盘和鼠标界面,由 Windows,而不是应用程序来处理这项工作。

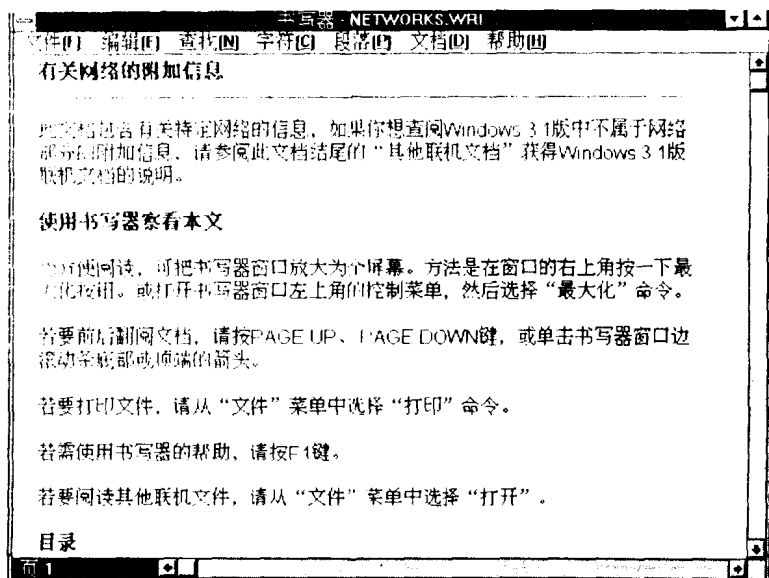


图 1.3 Windows“书写器”的屏幕

1.3.2 多任务

多任务操作系统允许用户同时运行几个应用程序或同一应用程序同时发出几种请求。尽管一些人仍怀疑多任务在微机上是否必要,但象 Borland 的 Sidekick 和 Norton 实用程序这样的 TSR(终止并驻留)应用程序的成功证明了它们的必要性。图 1.4 给出了几种平铺的 Windows 应用程序,每一程序在屏幕上都占用一个矩形窗口,用户能随时在屏幕上移动窗口、改变尺寸、切换不同应用程序以及交换窗口间的信息等。

图 1.4 的例子表示四个并发运行的进程。它们中仅有一个能在任一给定时间使用处理器。在处理任务和运行任务之间的区别是很明显的。一个应用程序很可能处在第三种状态,即所谓的活动状态。一个活动的应用程序能够接受用户的输入。正如在任一时刻只能运行一个应用程序一样,某一时刻也只有一个活动应用程序。不过,存在大量的同时运行任务。Windows 负责分配微处理器时间,它通过使用排队输入和消息来控制对处理器的共享。

在多任务操作系统以前,应用程序已全部控制了计算机资源,包括输入输出设备、内存、视频显示、甚至 CPU 本身。然而,在 Windows 环境,所有这些重要资源都必须共享。例如,对于系统、程序本身和任何 TSR 程序遗留下来的内存、标准 C/C++ 程序不再全部占用。

1.3.3 内存管理

在 Windows 中,内存是最重要的共享设备。当不只一个应用程序在同一时刻运行时,为避免它用完内存,这些程序必须协调共享内存。此外当旧程序结束,新程序开始时,内存就会变得不连续。Windows 可通过移动内存中的代码和数据块来优化空闲存储空间。