

适用于 IBM PC AT 286. 386. 486

3.0

Microsoft Windows

最新窗口软件程序设计指南

中国科学院希望高级电脑技术公司



73 .87221
1151

Microsoft Windows V3.0

最新多窗口程序设计指南

夏洪山 仇 钟 等编译
丁丽亚 林志坚



中国科学院希望高级电脑技术公司

一九九一年五月

译 序

Microsoft Windows及其软件开发工具3.0版的问世,标志着PC机系统软件的发展进入了一个新的阶段,为开发从用户界面到系统特性,运行方式和执行效率诸方面都与DOS完全不同的新一代软件,提供了理想环境和有效的开发手段。

Microsoft Windows 3.0 软件开发工具系列技术参考手册包括《多窗口程序设计工具》、《多窗口程序设计指南》和《多窗口程序设计参考手册》。在本书中,将从Windows应用程序开发者的角度出发,由浅入深地系统性地详尽介绍Windows 软件的运行原理,应用程序的组成,以及与各个组成部分有关联的基本概念和构造过程。并通过一系列的具体示范程序,说明如何编写各类 Windows 应用程序,如何避免和解决编程过程中经常碰到的问题,以及如何提高编程技巧和改善程序质量等等。

参加本书编译工作的主要人员有:夏洪山,仇钟、丁丽亚、林志坚和张亦葵等。在编译本书的过程中,得到有关部门和领导的大力支持,值此表示衷心感谢。中科院希望电脑公司对本书的印刷进行了周密的安排和组织,值此一并致谢。

编译者

一九九一年二月十日

目 录

引言.....	(1)
---------	-------

第一部分 Windows程序设计基础

第一章 Wrrndows环境概述

1.1 Microsoft Windows和DOS的比较.....	(6)
1.2 Windows编程模型.....	(8)
1.3 Windows库函数.....	(11)
1.4 建立Windows应用程序.....	(11)
1.5 软件开发工具	(12)
1.6 关于编写Windows应用程序的说明.....	(15)
1.7 小 结.....	(16)

第二章 Windows应用程序—Generic

2.1 Generic 应用程序.....	(18)
2.2 Windows应用程序.....	(18)
2.3 WinMain函数.....	(19)
2.4 窗口函数	(28)
2.5 创建会话框	(30)
2.6 创建模块定义文件	(34)
2.7 组合 Generic	(35)
2.8 把 Generic 当作模板使用.....	(43)
2.9 小 结.....	(44)

第二部分 Windows 程序设计

第三章 窗口的输出

3.1 显示缓冲区	(46)
3.2 创建、选择和删除绘画工具	(49)
3.3 画和写	(50)
3.4 示范程序: output	(51)
3.5 小 结.....	(57)

第四章 键盘和鼠标器输入

4.1 Windows 输入信息.....	(58)
4.2 示范程序: Input.....	(61)
4.3 小 结	(67)

第五章 标象

5.1 什么是标象	(69)
5.2 使用自定义标象	(70)
5.3 指定类型标象	(71)
5.4 显示自定义标象	(71)
5.5 在会话框中显示标象	(72)
5.6 示范程序: ICON	(73)
5.7 小 结	(74)

第六章 光标、鼠标器和键盘

6.1 控制光标的形状	(74)
6.2 显示光标	(76)
6.3 用鼠标器选择信息	(77)
6.4 利用键盘控制光标	(81)
6.5 应用程序举例 Cursor	(84)
6.6 小 结	(90)

第七章 清单

7.1 什么是清单	(92)
7.2 定义一个清单	(92)
7.3 在应用程序中引入一个清单	(94)
7.4 处理清单输入	(95)
7.5 在应用程序中处理清单	(96)
7.6 清单的特定属性	(102)
7.7 一个应用程序范例: EditMenu	(109)
7.8 小 结	(113)

第八章 控制框

8.1 什么是控制框	(115)
8.2 创建控制框	(115)
8.3 使用控制框	(117)
8.4 创建和使用普通控制框	(119)
8.5 示范程序: EditCntl	(132)
8.6 小 结	(135)

第九章 会话框

9.1 什么是会话框	(136)
9.2 使用会话框	(137)
9.3 应用程序范例: FileOpen	(138)
9.4 小 结	(145)

第十章 文件输入与输出

10.1 Windows环境中处理文件的规则	(147)
10.2 创建文件	(148)

10.3	打开已存在的文件	(149)
10.4	从文件中读和向文件中写	(149)
10.5	重新打开文件	(150)
10.6	提示文件	(150)
10.7	检查文件状态	(150)
10.8	程序范例: EditFile	(151)
10.9	小结	(160)
第十一章 位图		
11.1	什么是位图	(161)
11.2	创建位图	(161)
11.3	显示位图	(167)
11.4	对单色位图增加颜色	(171)
11.5	删除位图	(172)
11.6	应用程序范例: 位图	(172)
11.7	小结	(181)
第十二章 打印		
12.1	在Windows环境下打印	(182)
12.2	获得当前打印机的有关信息	(183)
12.3	打印一行文本	(184)
12.4	打印位图	(185)
12.5	处理打印期间的错误	(186)
12.6	终止打印操作	(188)
12.7	使用分块技术打印图象	(191)
12.8	应用程序范例: PrntFile	(192)
12.9	小结	(197)
第十三章 数据剪辑区		
13.1	使用剪辑区	(198)
13.2	使用特殊的剪辑区特性	(204)
13.3	应用程序范例: ClipText	(206)
13.4	小结	(211)

第三部分 Windows程序设计再探

第十四章 C与汇编语言

14.1	选择内存模式	(212)
14.2	使用NULL	(213)
14.3	使用命令行参数以及DOS环境	(214)
14.4	输出函数	(214)
14.5	使用C运行期函数	(215)
14.6	编写汇编语言代码	(220)

14.7	小结.....	(224)
第十五章	内存管理	
15.1	使用内存.....	(224)
15.2	分段技术.....	(228)
15.3	应用程序举例: Memory	(230)
15.4	小结.....	(232)
第十六章	再论内存管理	
16.1	Windows 内存结构	(233)
16.2	在 Windows 应用程序中使用数据存储.....	(243)
16.3	使用内存模式.....	(257)
16.4	使用巨型数据.....	(258)
16.5	程序数据管理过程中应避免的问题.....	(258)
16.6	程序代码的内存管理.....	(260)
16.7	小结.....	(262)
第十七章	打印设置	
17.1	Windows 怎样管理打印设置.....	(263)
17.2	使用设备驱动函数	(265)
17.3	检查打印机驱动程序能力	(266)
17.4	处理打印设置.....	(266)
17.5	在驱动程序之间复制打印设置.....	(272)
17.6	维护应用程序的打印设置.....	(272)
17.7	利用旧版本打印机驱动程序.....	(272)
17.8	小结.....	(273)
第十八章	字体	
18.1	书写正文.....	(274)
18.2	书写彩色正文.....	(274)
18.3	使用固有字体.....	(274)
18.4	创建逻辑字体.....	(275)
18.5	一行中使用多种字体.....	(276)
18.6	获得当前选择字体状况.....	(277)
18.7	获得逻辑字体状况.....	(278)
18.8	列举各种字体.....	(279)
18.9	检查设备的正文显示能力.....	(281)
18.10	增加字体资源.....	(282)
18.11	设置正文排列方式.....	(282)
18.12	创建字体资源文件.....	(283)
18.13	示范程序: ShowFont	(285)
18.14	小结.....	(285)
第十九章	调色板	

19.1	调色板的作用.....	(286)
19.2	调色板工作原理.....	(286)
19.3	创建和使用逻辑调色板.....	(288)
19.4	用调色板颜色画.....	(291)
19.5	修改逻辑调色板.....	(293)
19.6	对系统调色板修改的响应.....	(295)
19.7	小结.....	(296)
第二十章 动态链接库		
20.1	什么是 DLL.....	(298)
20.2	何时使用用户 DLL	(301)
20.3	创建 DLL	(310)
20.4	应用程序对 DLL 代码的访问.....	(317)
20.5	Windows 目标所有权规则.....	(319)
20.6	一个简单库: select	(320)
20.7	小结.....	(326)
第二十一章 多道文档显示界面		
21.1	MDI应用程序结构.....	(327)
21.2	初始化MDI应用程序.....	(328)
21.3	编写信息主循环.....	(329)
21.4	编写框架窗口函数.....	(330)
21.5	编写子窗口函数.....	(330)
21.6	与子窗口相关的数据.....	(331)
21.7	控制子窗口.....	(331)
21.8	小结.....	(333)
第二十二章 数据动态交换		
22.1	Windows 中的数据交换.....	(333)
22.2	DDE 概念	(335)
22.3	DDE 信息	(335)
22.4	DDE 信息流	(336)
22.5	DDE 应用举例	(336)
22.6	小结.....	(336)

引 言

引言部分提供与本书有关的背景情况介绍,包括以下一些内容:

- 阅读本书之前必须具备的知识
- 本书的目的和内容
- 用来生成Windows应用程序的工具
- 使用本书中介绍的示范程序
- 本书中使用的符号约定
- 随Microsoft Windows软件开发工具提供的各种手册介绍

阅读本书之前必须具备的知识

开始阅读本书之前,你应该具备以下的知识和经验:

- 接触过Windows,并熟悉Windows用户界面。

开始开发Windows应用程序之前,应该首先安装 Windows 3.0版并学会怎样使用它。一定要弄懂Windows 应用程序各个部分(例如窗口、会话框、清单、控制框和滚动杠等)的名称、作用和操作。只有在此基础上,才能正确地运用Windows 的各种特性,开发你的Windows应用程序。

- 理解Windows用户界面的通用格式

Windows 的一个设计目标,就是为所有应用程序提供一种通用的用户界面。这使得编程人员设计用户界面时能做到简单明了,用户也不必特别费心去熟悉每一个 Windows 应用程序的用户界面。为了实现这种目标,你必须按照推荐使用的应用程序格式编写用户界面。《高级界面设计指南》中介绍了推荐使用的程序格式。

- 曾经编写过C语言程序并使用过C标准运行时函数。

C语言是开发Windows应用程序的首选语言。Windows的许多编程特性主要是为C程序设计人员而设计的。(也可以用汇编语言或PASCAL语言开发Windows 应用程序,但是需要完成一些额外的准备工作,而使用C语言编程却不需要。)

关于本书

本书打算帮助有经验的C编程人员,逐渐适应编写Microsoft Windows 3.0版格式的应用程序。书中介绍了怎样使用Windows函数、信息和数据结构,完成所有Windows 应用程序中普遍存在的工作,并且提供了各种示范程序,以便在 Windows 3.0 版下编译和执行它们。

本书由三部分组成,每一部分包含几章。

第一部分“对编写Windows应用程序的介绍”提供Windows 环境的概况,并且深入地分析了一个Windows应用程序。第一部分由以下两章组成。

■ 第一章“Windows环境概况”描述了Windows的概况,比较了Windows环境和标准C语言环境,并且说明了Windows编程模式和开发Windows应用程序的过程。

■ 第二章“Windows应用程序Generic”介绍怎样创建一个名为Generic的Windows应

用程序。这个应用程序可以作为以后各个示范程序的基本框架。

第二部分“编写Windows应用程序”解释了编写Windows应用程序的主要技术，如列出清单、打印、使用数据传递E等。每一部分介绍一个主题，并提供一个说明这个主题的示范程序。第二部分由以下各章组成：

■ 第三章“窗口的输出”介绍了图形设备界面（GDI），并说明了怎样使用GDI工具产生你自己的输出。

■ 第四章“键盘和鼠标器输入”说明怎样处理键盘和鼠标器输入。

■ 第五章“标象”说明怎样建立和显示应用程序的标象。

■ 第六章“光标、鼠标器和键盘”解释光标、鼠标器和键盘的作用，并说明怎样在应用程序中使用它们。

■ 第七章“清单”说明怎样建立应用程序的清单和怎样处理清单输入。

■ 第八章“控制框”说明怎样设置和使用诸如回答键和列表框等控制框。

■ 第九章“会话框”说明怎样设置和使用会话框，怎样在会话框中放置控制框。

■ 第十章“文件输入和输出”介绍OpenFile函数和磁盘文件操作规则。

■ 第十一章“位图”说明怎样设置和使用位图。

■ 第十二章“打印”说明怎样在Windows下使用打印机。

■ 第十三章“数据传递区”解释数据传递区的概念，并且说明怎样在应用程序中使用它。

第三部分“高级编程技术”介绍内存管理和动态数据交换等高级技术。第三部分由以下各章组成，每一章说明一个主题：

■ 第十四章“C和汇编语言”提供编写C语言和汇编语言Windows应用程序的方法。

■ 第十五章“内存管理”说明怎样分配全局内存和局部内存。

■ 第十六章“再论内存管理”更详细地说明应用程序如何有效地管理内存。这一章还说明在各种不同的内存配置下，Windows怎样管理内存。

■ 第十七章“打印设置”说明怎样根据应用程序的需要设置打印参数（如页面大小和打印方向等）。

■ 第十八章“字体”说明怎样创建和装入字体，怎样在TextOut函数中使用字体。

■ 第十九章“调色板”说明怎样利用Windows调色板，在应用程序中最有效地使用颜色。

■ 第二十章“动态链接库”说明怎样创建和使用Windows动态链接库。

■ 第二十一章“多文档界面”说明怎样编制和使用Windows多文档界面（MDI）的应用程序，让用户同时处理多个文档。

■ 第二十二章“动态数据交换”说明怎样利用基于信息的动态数据交换规程，把数据从一个应用程序传递给另一个应用程序。

需要的工具

建立Windows 3.0版应用程序，需要使用以下的工具：

■ Microsoft C优化编译器：CL

■ Microsoft分段可执行链接程序：LINK

■ Microsoft Windows资源编译器：RC

■ Microsoft Windows绘图程序: SDKPAINT

■ Microsoft Windows对话框编辑器: DIALOG

建立Windows库和字体资源文件, 需要使用以下的工具:

■ Microsoft 宏汇编: MASM

■ Microsoft Windows字体编辑器: FONTEDIT

建立和调试Windows应用程序, 可能还需要使用以下的工具:

■ Microsoft程序维护实用程序: MAKE

■ Microsoft符号调试器: SYMDEB

■ Microsoft Windows的CodeView: CVW

■ Microsoft Windows动态分析器: PROFILER

■ Microsoft Windows转贮程序: SWAP

■ Microsoft Windows堆检测程序: HEAPWALK

■ Microsoft Windows信息观察程序: SPY

这些工具中的大多数由Microsoft Windows 软件开发工具3.0版提供, 只有C编译器、链接程序、宏汇编和程序维护实用程序例外。《工具》一书中对所有这些工具有详细介绍。

Windows 3.0版软件和硬件需求的清单参见《安装和更新手册》

使用示范程序

本书中的示范程序都是用C语言编写的, 它们采用了推荐的Microsoft Windows应用程序用户界面格式。

所有示范程序的源程序文件都在随SDK一起提供的示范程序源代码盘上。一边阅读本书中的说明, 一边观察对应的示范程序效果很好。为了查找方便, 包含示范程序源文件的目录按照各章命名。你可以把这些示范程序作为需要开发应用程序的基本框架。

用语习惯

本书是为Windows应用程序开发人员而编写的。书中的“你”既可以指程序开发人员, 有时也可以指开发的应用程序。例如,

“你使用SDKPaint编辑器建立标象、光标和位图。”

“你可以用TextOut函数显示正文。”

“你的应用程序需要刷新其用户区, 将会收到一个WM_PAINT信息。”

在本书中, “用户”一词不是指作为应用程序开发人员的你, 而是指最终将使用你编写的应用程序的人。例如,

“当用户选择About清单时, 你的应用程序显示About对话框。”

“你还可在清单项边上显示一个检查标记, 表示用户选择了该项。”

符号约定

在本书中, 除非特别指明, “DOS”项都表示MS_DOS或PC_DOS。

以下是本书中用到的符号约定。

约定	约定的说明
黑体正文	黑体正文表示必须逐字母如实使用的特定域或标点符号: 语言中的关键字或函数名(如EXETYPE或CreateWindow)、DOS命令行参数(如/

	Zi) 等。你必须准确地按照所示的正文送入项和标点符号。然而, 不必区分大小写。例如, 可以在DOS提示符之后送入LINK、link或Link, 启动链接程序。
()	在语法说明中, 括号括住传递给函数的一个或几个参数。
斜体正文	斜体正文表示一个替代符, 需要你提供实际值。例如, SetCursorPos 函数的语法是: SetCursorPos (X, Y), 指示你必须代替 X 和 Y 坐标的值。
等线体	代码用笔划粗细一样的字形显示。
BEGIN	程序例子中的垂直省略号表示省略了这一部分程序。
:	
END	
...	一个项后面的水平省略号表示可能出现多个同样形式的项。在下例中, 水平省略号表明你可以在g命令中指定多个breakaddress值。
	g [[=startaddress]] [[breakaddress]] ...
[]	双括号括住命令行或语法说明的可选择域或参数。在下例中, option和executable_file是RC命令的可选择参数:
	RC [[option]] filename [[executable_file]]
	竖线表示可以输入两边项中的任意一个。在以下的命令行语法说明中:
	DB [[address range]]
	竖线表示可以在DB命令之后, 指定address或range。
" "	双引号表示正文形式的项。
{ }	大括号表示可以指定其中一项。
小的大写字母	小的大写字母表示键或键序列的名称, 例如ALT+SPACEBAR。

Microsoft Windows软件开发工具

本书中的“SDK”特指Microsoft Windows软件开发工具及其内容。SDK 包括以下手册:

标题	内容
《安装和更新手册》	提供SDK的设计指南, 说明怎样安装SDK软件, 着重说明3.0版的改进之处。
《编程指南》	介绍怎样编写 Windows 应用程序, 并提供一些示范程序, 供你编写应用程序时参考。《编程指南》还涉及一些编写 Windows 应用程序的高级技术。
《工具》	说明怎样使用诸如调试器和特定的 SDK 编辑器这样一些软件开发工具, 用来建立你的Windows应用程序。
《参考》	对Microsoft Windows应用程序编程界面(API)作了广泛而详尽的描述。《参考》按照字母顺序列出API当前所有的函数、信息和数据

结构，并且说明怎样使用API。

《普通用户可涉及的系统应用程序结构：高级界面设计指南》或称：《高级界面设计指南》提供编程指导和规范格式介绍，以便使所编写的 Microsoft Windows 应用程序保持一致性。

第一部分 Windows程序设计基础

Windows 应用程序通常是用C语言编写的，但是在许多方面与标准C语言程序有很大不同。因为应用程序如果要在Windows下顺利执行，就必须与Windows和其它应用程序相互作用，随时有可能把控制交给Windows，而且必须与Windows和其它应用程序共享各种资源。

第一部分介绍Windows环境，并把它与标准C程序通常运行的环境进行比较。这一部分还介绍Windows应用程序的基本结构，并用一个简单的例子说明这种结构。

读完第一部分之后，你将对Windows环境以及典型的Windows应用程序的结构有一个大致了解。

第一章 Windows环境概述

Microsoft Windows 3.0版具有标准DOS环境不具备的许多特性。因此Windows应用程序在许多方面比标准DOS程序要复杂得多。

本章介绍以下内容：

- Windows应用程序与标准DOS应用程序的比较。
- Windows环境的某些特性，以及它们对开发应用程序所起的作用。
- Windows编程模型。
- 开发Windows应用程序的过程。

1.1 Microsoft Windows和DOS的比较

Microsoft Windows具有标准DOS环境不具备的许多特性。因此，Windows应用程序初看起来比标准DOS程序复杂。下面是Windows提供的一些附加特性：

- 应用程序中窗口、清单、会话框和控制框的图形化用户界面。
- 队列化输入。
- 具有设备独立性的图形
- 多任务。
- 应用程序间数据交换。

大多数C语言编程人员在编写DOS环境下的应用程序时，需要使用C运行时库来完成程序的输入、输出、内存管理和其它的活动。C运行时库需要在一个由用户输入/输出字符终端构成的操作环境下执行，对系统内存和输入/输出设备进行互斥的访问。在Windows下，情况并不是这样。Windows下的许多应用程序共享计算机的各种资源，包括CPU资源。Win-

dows应用程序通过图形显示器、键盘和鼠标器与用户相互作用。

以下各节说明标准DOS应用程序与Windows应用程序的主要差别。

1.1.1 用户界面

设计Windows的基本原则之一，是同时显示多个（可能并不是全部的）应用程序。在多任务环境下，让每个应用程序都占有一部分屏幕是很重要的，这样可以使用户对所有应用程序进行控制。在某些系统下，一个应用程序完全占有屏幕，其它的应用程序在后台等待。而在Windows下，每个应用程序都可以始终占有屏幕的一部分。

一个应用程序使用一个“窗口”与用户交互，以此方式与其它应用程序共享屏幕。实际上，窗口是系统允许应用程序使用的一块矩形显示区，它是整个屏幕的一部分。窗口由各种视频构件组成，如清单、控制框和滚动杠等。用户通过它们可以方便地指示应用程序执行的动作。

在标准DOS环境下，系统自动地为应用程序准备系统显示区，最典型的是传递一个文件把柄给应用程序。应用程序利用这个文件把柄，通过使用约定的C运行时库函数或是DOS系统调用，把输出送到系统显示区。在Windows下，你必须首先创建自己的窗口，然后才能输出信息或是接收输入。一旦创建了窗口之后，Windows提供许多有关用户对此窗口操作的信息，并且自动完成许多用户要求的动作，如移动窗口和改变窗口大小等。

与标准C语言程序相比，Windows环境的另一个优越性是可以创建并使用任意数目的覆盖式窗口，来显示各种各样的信息。而标准C语言程序只能访问单层的“屏幕面”。Windows负责管理屏幕，控制各个窗口的位置和显示的内容，并且保证两个应用程序不会同时试图访问同一块系统显示区。

1.1.2 队列化输入

Windows应用程序和标准C程序最主要的区别之一，是它们接收用户输入的方式。

在DOS环境下，程序通过显式地调用象getchar这样的函数，从键盘读数据。这种函数需要等到用户按键之后，才返回字符码给程序。而在Windows环境下，Windows从键盘、鼠标器和定时器接收全部的输入，并把它们放到恰当的应用程序的“信息队列”中。应用程序需要读输入时，只要读它的信息队列就可以了。

在标准DOS环境下，典型的输入形式是从键盘得到的8位字符。标准的输入函数getchar和fscanf从键盘读字符，并返回与所按键有关的ASCII码或其它码。程序也可以拦截鼠标器和定时器等输入设备产生的中断，并把从这些设备得到的信息作为输入。

在Windows下，应用程序接收“输入信息”形式的输入。这些输入由Windows发送，其中包含有比标准DOS环境下更丰富的内容。它们包括系统时间、鼠标器位置、键盘状况、键的扫描码（如果按了键）、被按下的鼠标键以及产生的设备。例如，两个键盘信息WM_KEYDOWN和WM_KEYUP，对应于按下和放开特定的键。在每个键盘信息中Windows提供用以指定键的与设备无关的虚拟键码、与键盘设备有关的扫描码，还有键盘上SHIFT、CONTROL、NUMLOCK等键的状态。键盘、鼠标器和定时器信息具有相同的格式，可以用相同的方式处理。

1.1.3 独立于设备的图形

Windows为你编程提供了丰富的独立于设备的图形，你可以在应用程序中很方便地画线、矩形、圆和其它复杂图形。由于Windows提供设备独立性，要在点阵打印机或高分辨

率图形显示器上画一个圆，可以使用相同的函数。

Windows要求“设备驱动程序”把图形输出要求转换为打印机、绘图仪、显示器或其它输出设备的输出。设备驱动程序是一种特殊的可执行库，应用程序把它装入并连接到特定的输出设备和端口上。“设备输出缓冲区”表示设备驱动程序、输出设备和可能使用的通讯端口。应用程序在特定设备的“输出缓冲区”中完成对图形的操作。

1.1.4 多任务

Windows是一个多任务系统，多个应用程序可以同时执行。在标准DOS环境下，不具备多任务的特性。在DOS环境下编写的程序，总是假定为可以独占计算机中所有资源，包括输入和输出设备、内存、系统显示区，甚至CPU本身。而Windows下的应用程序必须与当前运行的所有其它应用程序共享各种资源。因此，Windows小心谨慎地控制这些资源，同时也要求各个应用程序使用指定的编程接口。

例如，在标准DOS环境下，一个程序可以访问未被系统驻留程序(TSR)、这个程序本身占用的所有内存。也就是说程序可以任意地使用所有可用的内存。

在Windows下，内存是一种共享资源，因为多个应用程序可以同时执行，它们必须相互合作共享内存，以免用尽了这种资源。应用程序可以从系统内存中分配到它们需要的内存。Windows分配的内存来源于两处：用来分配大块内存的全局内存和用来分配小块内存的局部内存。为了最有效地使用内存，Windows经常移动甚至放弃内存块。当你把一个内存位置赋给一个目标之后，不要假定它总在原处。在多个应用程序同时执行的情况下，Windows可能频繁地移动和放弃这个内存块。

系统显示器是共享资源的另一个例子。在标准DOS环境下，系统允许应用程序独占系统显示器。你可以随意地使用显示器，既可以改变正文和背景的颜色，又可以正文视频模式变成图形视频模式。Windows下的可用程序必须与其它应用程序共享显示器，所以不允许它直接控制显示器。

1.2 Windows 编程模型

大多数Windows应用程序利用以下几种“部件”实现与用户的交互操作：

- 窗口
- 清单
- 会话框
- 信息环

下面几节详细说明这几种组成的作用。

1.2.1 窗口

窗口是任何一个Windows应用程序基本的输入和输出形式，它是应用程序单独访问的一块系统显示区。窗口占据系统显示器上一块矩形区，其中包含有标题杠、清单杠、滚行杠、边框和其它一些特性。你可以在创建窗口时指定需要的特性，然后由Windows画出并管理这个窗口。图1.1列出了窗口的主要特性。

尽管应用程序创建窗口并对它拥有主权，但是这个窗口的管理是由应用程序和Windows共同完成的。Windows维护窗口的位臵和外观，管理象边框、滚行杠、标题栏等窗口的标准特性，并根据用户初始化要求完成一些直接影响这个窗口的动作。应用程序完成剩下的其

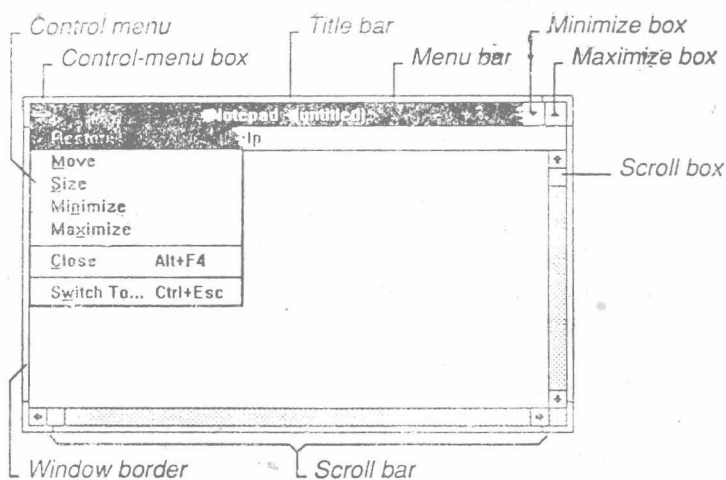


图1.1 窗口特性

它工作，特别是管理窗口“用户区”（窗口边框以内的区域）的工作。窗口用户区的内容完全由应用程序控制。

为了实现这种共同管理的方式，Windows把可能对窗口有影响的动作通知每个窗口。因此，每个窗口相应地有一个“窗口函数”，它接收窗口管理信息并作出适当的回答。窗口管理函数既可以指定需要窗口函数完成的动作，也可以要求从窗口函数得到信息。

1.2.2 清单

清单是Windows应用程序基本的用户输入方式。清单上列出一些命令，并允许用户从中选择。在创建应用程序时，你可以提供清单和命令名。Windows负责管理清单，并根据用户的选择发送信息给窗口函数。这个信息通知应用程序执行这条命令。

1.2.3 会话框

会话框是一种临时窗口，用户可以通过它提供命令所需的详细信息。会话框中含有一个或几个“控制框”，它是一种具有简单输入或输出功能的小窗口。例如，“编辑框”是一种允许用户输入和编辑正文的简单窗口。用户可以通过会话框中的各种控制框，提供文件名、在选择项中作选择，或是指示与执行命令有关的动作。

1.2.4 信息环

由于应用程序总是从队列中得到输入，因此“信息环”就成为Windows应用程序的关键部分。信息环从应用程序队列中得到信息，并把它们分配给适当的窗口。

图1.2说明了Windows和应用程序是怎样共同处理键盘输入信息的。当用户按下或放开一个键时，Windows得到键盘输入，并把键盘信息从系统队列复制到应用程序队列。信息环得到这个键盘信息，把它转换成ANSI字符信息WM_CHAR，然后把WM_CHAR信息连同键盘信息传递给应用程序窗口函数。窗口函数使用TextOut函数在窗口用户区中显示这个字符。

Windows一次可以接收和分配多个应用程序的输入信息。正如图1.3所示的，Windows从系统队列中收集信息形式的输入，并把每个输入信息复制到适当应用程序的队列中。每个应用程序的信息环获得信息，并通过Windows把它们分配给应用程序的适当窗口函数。