简介 …… 1
0.1 本书预期读者 …… 1
0.2 需要什么样的程序设计工具 …… 2
第一章 Windows 95 概述 …… 3
1.1 什么是 Windows 95 …… 3
1.2 Windows 95 使用了基于线程的多任务 …… 4
1.3 Windows 95 基于调用的接口 …… 4
1.4 动态链接库(DLL) …… 5
1.5 Windows 95 和 Windows 3.1 …… 5
1.5.1 用户眼中的区别 …… 6
1.5.2 程序设计人员眼中的区别 …… 7
1.6 NT 连接 …… 8
1.7 需要什么样的软件 …… 8
1.8 转换要点 …… 9
第二章 Windows 95 程序设计基础知识 …… 10
2.1 Windows 95 程序设计观点 …… 10
2.1.1 桌面模式 …… 11
2.1.2 鼠标 …… 11
2.1.3 图标和位图 …… 11
2.1.4 菜单、工具条、状态条和对话框 …… 12
2.2 Windows 95 是如何同程序交互的 …… 12
2.3 Win32 API:Windows 95 API …… 12
2.4 窗口组成成分 …… 13
2.5 一些 Windows 95 应用程序基础知识 …… 14
2.5.1 WinMain() …… 14
2.5.2 窗口函数 …… 15
2.5.3 窗口类 …… 15
2.5.4 消息循环 …… 15
2.5.5 Windows 数据类型 …… 16
2.6 Windows 95 框架程序 …… 16

2.6.1 定义窗口类…… 19
2.6.2 创建窗口…… 21
2.6.3 消息循环…… 23
2.7 窗口函数…… 24
2.8 使用定义文件…… 25
2.9 命名规则…… 25
第三章 消息处理 …… 27
3.1 什么是消息…… 27
3.2 应答一个按键…… 28
3.3 设备文本…… 32
3.4 处理 WM_PAINT 消息 …… 33
3.5 应答鼠标消息…… 37
3.5.1 深入观察鼠标消息 …… 41
3.6 产生 WM PAINT 消息 …… 41
3.7 产生计时器消息…… 45
第四章 消息框与菜单 …… 49
4.1 消息框…… 49
4.2 菜单入门…… 53
4.2.1 使用资源…… 54
4.2.2 编译.RC 文件 …… 54
4.2.3 创建一个简单的菜单…… 54
4.3 在程序中包含一个菜单…… 57
4.4 对菜单选择的响应…… 57
4.4.1 一个示例菜单程序…… 58
4.5 增加菜单加速键…… 61
4.6 装入加速键表…… 64
第五章 对话框入门 …… 67
5.1 对话框如何与用户进行交互…… 67
5.2 模态与非模态对话框…… 68
5.3 接收对话框的消息…… 68
5.4 激活一个对话框…… 69
5.5 创建一个简单的对话框…… 69
5.5.1 对话框资源文件…… 70
5.5.2 对话框窗口函数…… 72
5.6 第一个对话框示例程序…… 73
5.7 增加一个列表框…… 77
5.7.1 列表框基础…… 78
5.7.2 初始化列表框…… 79
5.7.3 处理一个选择…… 79

5.7.4 整个列表框示例 …… 80
5.8 增加一个编辑框 …… 84
5.9 使用一个非模态对话框 …… 86
5.9.1 创建一个非模态对话框 …… 88
第六章 创建定制图标、光标和位图 …… 94
6.1 定义一个图标和一个光标 …… 94
6.2 改变图标和光标 …… 95
6.3 演示一个定制图标和光标的示例程序 …… 95
6.4 定义小图标 …… 98
6.5 使用位图 …… 102
6.5.1 创建一个位图 …… 102
6.5.2 显示一个位图 …… 102
6.6 一个完整的位图示例程序 …… 105
6.7 使用多个位图 …… 107
第七章 控件细述 …… 112
7.1 复选框的使用 …… 112
7.2 复选框的管理 …… 119
7.2.1 复选框切换 …… 119
7.2.2 初始化复选框 …… 119
7.3 增添静态控件 …… 124
7.4 增加单选按钮 …… 124
7.5 使用滚动条控件 …… 130
7.5.1 接收滚动条消息 …… 131
7.5.2 设置滚动条范围 …… 131
7.5.3 设置滚动条中滚动块的位置 …… 132
7.5.4 滚动条实例程序 …… 132
第八章 处理文本 …… 139
8.1 窗口坐标 …… 139
8.2 设置文本和背景颜色 …… 139
8.3 设置背景显示模式 …… 140
8.4 获得文本 Metrics …… 141
8.5 计算字符串的长度 …… 142
8.6 获得系统 Metrics …… 143
8.7 文本大小 …… 144
8.8 解决重画问题 …… 149
8.8.1 虚拟窗口理论 …… 149
8.8.2 另外一些 API 函数 …… 149
8.9 创建和使用虚拟窗口 …… 150
8.9.1 创建虚拟窗口 …… 151

8.9.2 使用虚拟窗口 …… 151
8.9.3 完整的虚拟窗口演示程序 …… 153
8.10 改变字体 …… 157
8.10.1 使用内建字体 …… 157
8.10.2 创建定制字体 …… 163
第九章 处理图形 …… 171
9.1 图形坐标系统 …… 171
9.2 画笔和画刷 …… 171
9.3 设置像素 …… 172
9.4 画线 …… 172
9.5 设置当前位置 …… 172
9.6 画弧 …… 173
9.7 显示矩形 …… 173
9.8 画椭圆和饼图 …… 174
9.9 画笔的处理 …… 175
9.10 创建定制画刷 …… 176
9.11 删除定制对象 …… 176
9.12 图形演示程序 …… 177
9.13 理解映射模式和视口 …… 183
9.13.1 设置映射模式 …… 185
9.13.2 定义窗口区域 …… 185
9.13.3 定义视口 …… 186
9.13.4 设置视口原点 …… 186
9.13.5 一个样本映射模式程序 …… 187
第十章 公用控件介绍 …… 196
10.1 常用控件的包含和初始化 …… 197
10.1.1 常用控件是窗口 …… 197
10.1.2 工具条的用法 …… 197
10.1.3 工具调位图的创建 …… 199
10.1.4 一个简单的工具条样本程序 …… 200
10.1.5 工具提示的添加 …… 210
10.1.6 包括工具提示的整个工具条程序 …… 212
第十一章 再谈公用控件 …… 221
11.1 使用上下控件 …… 221
11.1.1 创建上下控件 …… 221
11.1.2 上下控件消息 …… 222
11.1.3 使用上下控件 …… 223
11.2 创建一个旋转控件 …… 229
11.2.1 旋转控件样本程序 …… 229

11.3 使用轨道条 …… 234
11.3.1 轨道条风格 …… 235
11.3.2 发送轨道条消息 …… 235
11.3.3 处理轨道条通知消息 …… 236
11.3.4 轨道条演示程序 …… 236
11.4 使用进度条 …… 244
11.4.1 发送进长条消息 …… 244
11.4.2 进度条样本程序 …… 244
第十二章 公用控件最后一瞥 …… 250
12.1 使用状态窗口 …… 250
12.1.1 创建状态窗口 …… 250
12.1.2 状态窗口消息 …… 251
12.1.3 使用状态条 …… 251
12.2 制表控件介绍 …… 259
12.2.1 创建一个制表控件 …… 259
12.2.2 发送制表控件消息 …… 260
12.2.3 制表通知消息 …… 262
12.2.4 一个样本制表演示程序 …… 262
12.3 使用制表控件 …… 266
12.4 树型查看控件 …… 274
12.4.1 创建树型查看控件 …… 274
12.4.2 发送树型查看消息 …… 275
12.4.3 树型查看通知消息 …… 278
12.4.4 树型查看演标程序 …… 278
第十三章 Windows 控制台 …… 286
13.1 字符方式理论 …… 286
13.2 分配控制台 …… 287
13.3 指定控制台标题 …… 288
13.4 获取标准输入/输出句柄 …… 288
13.5 向控制台输出文本 …… 289
13.6 由控制台输入 …… 289
13.7 设置光标位置 …… 290
13.8 设置文本及背景颜色 …… 290
13.9 控制台与 C/C++标准 I/O 函数 …… 291
13.10 控制台演示程序 …… 291
13.11 鼠标管理 …… 293
13.12 控制台鼠标程序实例 …… 295
13.13 响应键盘事件 …… 297
13.13.1 键盘事件程序实例 …… 298

第十四章　多进程和多线程 …… 301
14.1　建立独立任务 …… 301
14.1.1　多进程程序实例 …… 304
14.2　建立多线程程序 …… 310
14.2.1　线程的建立 …… 311
14.2.2　线程的终止 …… 311
14.2.3　多线程程序实例 …… 312
14.2.4　使用多线程 …… 318
14.3　同步 …… 324
14.3.1　了解串行问题 …… 325
14.3.2　Windows 95 同步对象 …… 326
14.4　利用信号灯使线程保持同步 …… 327
14.5　使用事件对象 …… 334
14.6　下一章内容简介 …… 336
第十五章　API 剖析：剪贴板、打入记号和文件拖放 …… 337
15.1　剪贴板的使用 …… 339
15.1.1　剪贴板子系统 …… 339
15.1.2　将数据放到剪贴板上 …… 340
15.1.3　从剪贴板上读取数据 …… 343
15.1.4　剪贴板演示程序 …… 344
15.2　使用文本光标 …… 348
15.2.1　建立一个插入记号 …… 349
15.2.2　显示和隐藏一个插入记号 …… 349
15.2.3　设置一插入记号的位置 …… 350
15.2.4　撤消插入标记 …… 350
15.2.5　一个小型插入记号演示程序 …… 350
15.3　拖放文件 …… 353
15.3.1　接收拖放文件 …… 353
15.3.2　获取拖放文件的名称 …… 354
15.3.3　获取拖放文件的位置 …… 354
15.3.4　释放拖放文件的句柄 …… 355
15.3.5　文件拖放程序实例 …… 355
15.4　总结 …… 358
附录 A　资源描述语言快速参考 …… 359
A.1　ACCELERATORS …… 361
A.2　AUTO3STATE …… 362
A.3　AUTOCHECKBOX …… 362
A.4　AUTORADIOBUTTON …… 362
A.5　BITMAP …… 363

A.6 CAPTION …… 363
A.7 CHARACTERISTICS …… 364
A.8 CHECKBOX …… 365
A.9 CLASS …… 365
A.10 COMBOBOX …… 366
A.11 CONTROL …… 367
A.12 CTEXT …… 372
A.13 CURSOR …… 372
A.14 DEFPUSHBUTTON …… 373
A.15 DIALOG …… 374
A.16 DIALOGEX …… 374
A.17 EDITTEXT …… 375
A.18 EXSTYLE …… 375
A.19 FONT …… 376
A.20 GROUPBOX …… 376
A.21 ICON …… 377
A.22 LISTBOX …… 378
A.23 LTEXT …… 378
A.24 MENU …… 379
A.25 MENUEX …… 380
A.26 MENUITEM …… 380
A.27 POPUP …… 381
A.28 PUSHBOX 和 PUSHBUTTON …… 382
A.29 RADIOBUTTON …… 382
A.30 RCDATA …… 383
A.31 RTEXT …… 384
A.32 SCROLLBAR …… 384
A.33 STATE3 …… 385
A.34 STRINGTABLE …… 385
A.35 STYLE …… 386
A.36 User-Defined …… 388
A.37 VERSION …… 388
A.38 VERSIONINFO …… 389
附录 B OLE2 的一些术语 …… 395
B.1 什么是链接和嵌入 …… 395
B.2 成份对象模型 …… 396
B.3 OLE2 界面 …… 396
B.4 OLE 自动化 …… 397
B.5 OLE2 是 Window 的未来吗 …… 397

关于作者

Herbert Schildt 是世界级的C/C++作者。它的程序设计用书在世界各地已经售出150多万册，并已翻译成所有主要的外国语言。他是《Osborne Windows Programming Series，Windows NT Programming Handbook》，和最畅销书《Tearch Yourself C》和《Teach Yourself C++》三卷书的作者。他还写了《The Annotated ANSI C Standard》，《C：The Complete Reference（第二版）》，《C++：The Complete Reference》和《C++ from the Ground Up》，以及其它"无数"C/C++书的作者。Schildt 是在 Illinois 州 Mahomet 的软件咨询公司 Unversal Computing Laboratories 的总裁。他有 Illinois 大学的计算机科学硕士的学位。

我第一次看到 Windows 95 运行的时候，就感到计算机世界中将发生一些重要事件。一开始使用 Windows 95，我就认为计算机的外观将有很大改变。我看到的版本是 Windows 95 的 β 测试版，其用户界面的外观也已相当出色，获得了极大的成功。我立即意识到了 Windows 95 不仅仅是 Windows 的下一版本，它是专为 21 世纪设计的操作系统。

我们完全有理由说 Windows 95 是最灵活、最强大的 PC 机通用操作系统。它还相当复杂。尽管其比较复杂，Windows 95 不失为现有操作系统中考虑最周密的一个。从逻辑上看，Windows 95 的诸子系统是一致的。从逻辑上看，Windows 95 的基础知识，再创建一些可再用的代码段，就会觉得 Windows 95 用起来是很愉快的。毫无疑问，学习编写 Windows 95 程序是值得花费一定的时间和精力的。

写关于 Windows 95 的书的乐趣之一，是它提供了几乎无限的话题可供讨论。写完上一章，下一章的内容几乎自然地涌上心头。当然，这样也有缺点，有时几乎很难停下笔来。啊，这是个多么令人激动的系统啊！有这么多的话题可作为素材使用。写作本书的最困难的地方是，决定哪些话题选用，哪些话题留等下一本书再作讨论。在这么多的话题前面，我根据如下原则进行取舍：所有的程序设计基础知识都是本书内容之一，也就是说，本书详细介绍了所有 Windows 95 应用程序的基本成分；从 Windows 3.1 到 Windows 9.5 的转换也是本书内容之一，因为有许多程序设计人员都将需要把以前的 Windows 代码转换到新的 Windows 95 系统上来，所以，本书详细介绍了转换过程；最后，本书重点介绍了 Windows 95 独具的一些特点，如新公共控件、控制台和基于线程的多任务。

无论你是打算转换以前的 Windows 3.1 程序，还是第一次编写 Windows 风格的程序，本书都将对你有所帮助。本书包含了刚开始编写 Windows 95 程序时所需要掌握的全部信息，还着重介绍了如何将 Windows 95 用于一些特定场合。最后，本书给出了一些特殊的 Windows 3.1 转换要点，以帮助读者和用户进行顺利的转换。

本书是为 Windows 95 程序设计人员编写的。读者不必一定要有 Windows 3.1 程序设计经验，但必须熟悉 C 或 C++ 语言。要是读者先前没有用过 C/C++，建议先花点时间学习一下 C/C++ 语言，因为许多 Windows 95 构造都使用了相当复杂的程序设计技巧。

虽然读者不必具备操作系统理论基础知识，但若有这方面知识，学习起来将更快点。

最后一点：要是以前从来没有编写过 Windows 风格的程序，请一定要有耐心。Windows 风格的程序同以往的程序类型有许多不同之处。尽管如此，学习完前四章后，读者应当对 Windows 95 程序的整体结构很清楚，而且能够编写 Windows 95 程序了。

[redacted]

本书中所举的示例是用 Microsoft Visual C++ 2.0 编写、编译和测试的。读者也需要有这个编译器，或其他任一种能产生 Windows 95 兼容目标码的 C/C++ 编译器。

HS

1994.10.24

Mahomet，Il

[redacted]

本书首先而且主要是一本实用指南，介绍如何编写 Windows 95 程序。正因为如此，本书没有涉及到多少理论知识，除非直接与程序设计有关。一般而言，本书提供的都是一些很实用的方法，旨在帮助大家尽快编写出 Windows 95 的程序。

在编写 Windows 95 程序前，有必要先了解一下 Windows 95 的整体运行方法、它所引入的设计概念以及它是如何管理计算机的。理解 Windows 95 同其前辈 DOS 和 Windows 3.1 有何不同之处也相当重要。基于以上认识，本章简要了介绍了 Windows 95 及其同 DOS 和 Windows 3.1 的异同之处。

要是读者以前从来没有编写过 Windows 程序，可能会觉得本书中的大部分内容都很陌生。请耐心些！只要有条不紊地认真学习本书，学习完毕后就将成为一个熟练的 Windows 95 程序设计人员。要是以前编写过 Windows 3.1 程序，那么就能快一点学完本书，但亦请注意细心，因为 Windows 95 和 Windows 3.1 毕竟有一些不同之处，而这些不同之处将会影响到程序设计方式。

还有一点，Windows 95 是一个非常庞大而且复杂的程序设计环境，不可能在本书里介绍完所有方面（这样做的话，可能要写许多本书）。本书仅介绍那些对所有程序来说都共用的 Windows 95 程序设计成分，它们最常使用，或是 Windows 95 独具的重要革新之处。学习完本书后，再学习其他子系统就容易得多了。

1.1 [redacted]

Windows 95 是面向 21 世纪的下一代操作系统的一部分，它克服了其前身 Windows 3.1 所强加的一些限制，而且还增加了许多新特点，在用户界面也有全新而重要的改进。

要是一定要找出哪一个特点是重要的话，那么 Windows 95 是一个 32 位操作系统，这一点可能最重要。读者亦将在学习本书的过程中体会到这一点。由于已转向 32 位实现，所以，Windows 95 抛掉了许多早期的 16 位系统所带来的问题和怪异之处。

与 Windows 3.1 和 DOS 兼容是 Windows 95 的主要设计目标之一。也就是说，大量现有的 PC 应用程序可以在 Windows 95 上运行。在 Windows 95 上可以运行四种类型的应用程序：DOS 应用程序、Windows 3.1 应用程序、Windows NT 应用程序以及专门为 Windows 95 编写的 Windows 95 应用程序。无论运行哪一种应用程序，Windows 95 都能自动建立合适的

环境。比如说，如果运行一个 DOS 应用程序，Windows 95 就会自动建立一个窗口化的命令提示符，DOS 应用程序就在其中运行。

1.2 [redacted]

正如大家所知道的那样，Windows 95 是一个多任务操作系统，也就是说，它能同时运行两个或更多个应用程序。当然，从技术角度而言，这些应用程序实际上是在分享 CPU，而不是同时使用 CPU。但由于计算机的处理速度相当高，对用户而言看起来似乎这些应用程序是在同时运行。Windows 95 支持两种方式的多任务：基于进程的多任务和基于线程的多任务。所谓进程，实际上就是正在执行中的程序。基于进程的多任务是指：在同一时刻，可以同时运行多个程序。也就是说，Windows 95 支持这种为人们所熟悉的传统的基于进程的多任务。

Windows 95 的第二种多任务方式是基于线程。所谓线程，实际上是指可执行代码的一个可分配单元。线程的英文名叫 thread，源于"thread of execution"(执行线索)。所有的进程都至少有一个线程，一个 Windows 95 进程可以有多个线程。

那么，我们能不能从基于线程的多任务和一个进程可以有多个线程得出结论：在 Windows 95 中，可以将一个进程分成数片，同时运行？结论非常正确。也就是说，在 Windows 95 中，不但可以使多个程序同时运行，还可以使用一程序的不同代码段同时运行。大家将在本书中看到，正是由于这种能力，才使得编写效率极高的程序成为可能。

1.2 [redacted]

大家都知道，在 DOS 环境中，程序设计人员是通过各种软件中断与 DOS 打交道的，比如标准的 DOS 中断 0x21。虽然使用软件中断来存取 DOS 服务是完全可以接受的(撇开 DOS 操作系统的限制不谈)，但对于像 Windows 95 这样的功能完备的多任务操作系统而言，是远远不够的。因此，同 Windows 3.1 一样，Windows 95 也使用了基于调用的接口。

Windows 95 基于调用的接口提供了非常多的系统定义的函数，供存取操作系统特点使用。这些函数组织在一起，就称为 API(Application Programming Interface，应用程序设计接口)。API 共包含了数百个函数，供程序设计人员用来与 Windows 95 打交道。这些函数包括了所有必要的与操作系统相关的活动，如内存分配、输出至屏幕、建立窗口等等。

1.1 动态链接库(DLL)

既然 API 是由数百个函数组成的，是不是在 Windows 95 编译时将要把大量代码链接进每一个程序，从而导致程序中包含大量重复代码呢？答案是否定的，实际情况并不是这样。在 Windows 95 中，API 函数是包含在动态链接库(DLL，Dynamic Link Libraries)中的，它们将在运行时动态链接，而不是在编译时静态链接。本书就着重介绍一下动态链接库的工作方式。

Windows 95 API 函数以可再定位格式存放在一个动态链接库中。编译时，若程序调用了一个 API 函数，链接器并不会将该函数所对应的代码实际链接进程序可执行文件中，而是光增加对应于该函数的装入指令，如该函数是什么名称，它存放在哪一个动态链接库中。当程序执行时，必要的 API 例程再由 Windows 95 装入器装入。按照这种方式，每一个应用程序都不需要包含实际的 API 代码，只是在应用程序已装入内存准备执行时，API 代码才会也装入内存。动态链接的优点相当重要。首先，由于实际上所有的程序都需要使用 API 函数，所以，这样做可以防止程序中出现大量重复代码，以免浪费磁盘空间。其次，更新和增强 Windows 95 时，只需要修改动态链接库例程，现有的应用程序无需重新编译。

1.2 Windows 95 和 Windows 3.1

Windows 95 是 Windows 产品序列中的下一个产品，该产品序列和最早产品是于 1985 年面世的，代表着操作系统设计的重要突破。好消息是，只要以前熟悉 Windows 3.1，现在学习和使用 Windows 95，甚至对 Windows 95 进行程序设计，就不会有什么大问题。从用户的观点来看，Windows 95 只是增加了一个改进后的用户界面，并且转向了以文档为中心的组织结构。特别是，像 Program Manager(程序管理器)和 File Manager(文件管理器)这样的基本成分已被 Start 菜单和 Expolyer 所取代。但是，尽管有了这些变化，如果用户以前会使用 Windows 3.1，现在使用 Windows 95 就不会有什么问题。从根本上来说，仍旧是相同的方式，并没有什么太大的变化。

从程序设计人员的观点看，最重要的好消息是，以前在 Windows 3.1 中怎么编程，现在在 Windows 95 中仍旧怎么编程。Windows 95 保留了原先的 Windows API 函数，在 Windows 95 中新增加的函数基本上都是一些新函数。总之，虽然 Windows 3.1 和 Windows 95 之间有一些区别，但还是易于处理的。另外，以前的 Windows 3.1 程序在 Windows 95 中运行得很好，所以也不必马上打算将所有应用程序全部移植到 Windows 95 上来。

下一节进一步介绍了 Windows 3.1 和 Windows 95 之间的一些区别。

1.5.1 用户眼中的区别

从用户角度看来，Windows 95 主要是在四个方面用 Windows 3.1 不同，如：

- 桌面界面已改变
- 窗口的风格已改变
- 新控件成分
- DOS 不再是必需的

正如前面所说，Windows 3.1 中的 Program Manager(程序管理器)已被 Start 菜单所取代。更进一步，桌面上现在多了一个任务条(Task Bar)，任务条上列出的都是系统中当前处于活动状态的任务，要想切换进入哪一个任务，只要用鼠标单击哪一个任务即可。绝大多数用户都认为，Task Bar 和 Start 菜单要比 Program Manager 先进得多。

在 Windows 95 中，窗口的外形也已重新设计。大多数用户将觉得新的外观更漂亮。原先对 Windows 3.1 的外观有很多批评，认为它们太呆板。现在，Windows 95 的外观已大大改善了。

使用 Windows 95 应用程序时，可以发现，有几个新的控件成分经常出现，如工具条、Spin(旋转式)控件、树形查看控件以及状态条等。Windows 95 定义了一些可供所有应用程序使用的激动人心的共有控件(本书后面部分将要介绍如何处理这些控件)。使用这些新控件，将使得应用程序看起来更现代化，而且一眼就能看出它们是 Windows 95 应用程序。

正如大家所期望的那样，Windows 95 不再需要 DOS。虽然 Windows 3.1 不是一个独立的操作系统，它需要在 DOS 之上运行，实际上它需要的是 DOS 对文件系统的支持，但 Windows 95 是一个完整的操作系统，DOS 不再是必需的。换言之，运行 Windows 95，并不需先启动 DOS。但是，Windows 95 仍提供对 DOS 应用程序的支持(事实上，有些 DOS 程序在 Windows 95 中运行要比在 DOS 中运行还要好)，而且，使用 Windows 95，可同时进行多个"DOS"会话。

Windows 95 还新增了一些功能，如透明运行 DOS 应用程序等。运行 DOS 应用程序时，一个窗口化的命令提示界面将会自动建立。更进一步，这个窗口化的命令提示符界面完全集成进了整体 Windows 95 图形界面中。而且，现在还可以在提示符下直接执行 Windows 应用程序(在 Windows 3.1 中，必须在 Windows 中才能执行 Windows 应用程序)。

Windows 95 的另一个新特点是，支持长文件名。大家可能已经知道，在 DOS 和 Windows 3.1 中，文件名至多只能是 8 个字符，扩展名至多只能是 3 个字符。而在 Windows 95 中，文件名可以长达 255 个字符。

Windows 95 附带提供了许多 Windows 3.1 不曾提供的附件程序和管理用工具。比如，Windows 95 支持便携计算、电子邮件、笔式计算以及远程计算等，Windows 95 还支持"插上即用"(Plug and play)特点，以简化新硬件成分的安装过程。

1.5.2 程序设计人员眼中的区别

从程序设计人员角度来看，Windows 3.1 和 Windows 95 之间主要有两个区别。首先，Windows 95 支持全 32 位寻址和虚存，而 Windows 3.1 则使用的是 16 位分段或寻址方式。对于许多应用程序而言，这样的区别影响不大。当然，对于某些应用程序，也可能影响很大。坦率地说，从 Windows 3.1 向 Windows 95 转换不会有什么困难，有时甚至会觉得在 Windows 95 的 32 位内存模式下更易于进行程序设计。

第二个主要区别在于多任务的实现方式。在进行切换时，Windows 3.1 使用的是非优先方法。也就是说，一个 Windows 3.1 任务必须手工将控制返回给调度程序，以便其他任务有机会得到运行。换言之，一个 Windows 3.1 程序将一直保留对 CPU 的占有，直到它主动放弃为止。因此，一个“设计不好”的程序可能会垄断 CPU。与此相反，Windows 95 使用的是优先分时式任务切换。在这种方式下，正在运行的任务可以被 Windows 95 自动中断，将 CPU 分配给下一个任务（如果有任务的话）。优先多任务通常非常出色，因为它允许操作系统完全控制任务切换，以防止某一个任务独占 CPU，从而导致其他任务使用不到 CPU。大多数程序设计人员都认为向优先多任务的转换是向前进了一大步。

除了上述两个主要区别外，Windows 95 和 Windows 3.1 之间还有其他一些小变化。下面就具体介绍一下。

1. 输入队列

输入队列中存放的是消息，例如按键或鼠标活动，直到它们被送给程序。在 Windows 3.1 中，系统中运行的所有任务共享一个输入队列。而在 Windows 95 中，则是每一个线程都有其自己的输入队列。这样做的优点在于，即使某一个任务应答消息的速度较慢，也不会降低系统性能。

虽然多输入队列可以说是一个极其重要的新增特点，但这并不会直接影响 Windows 95 的程序设计方式。

2. 线程和进程

Windows 3.1 只支持基于进程的多任务。也就是说，进程是 Windows 3.1 的最小分配单位。正如前面所说，Windows 95 既对线程进行切换，又对进程进行切换。以前的 Windows 程序在 Windows 95 中可以不加修改地运行，当然，也可充分利用基于线程的多任务，以增强其功能。

3. 控制台

过去，基于文本（非窗口化）的应用程序在 Windows 中使用起来相当不方便。但是，Windows 95 支持一种特殊类型的窗口，名为控制台。控制台窗口提供了标准的基于文本的界面、命令提示符环境。但是，撇开基于文本之处，控制台可像其他窗口一样处理。增加了基于文本的控制台后，不仅允许非窗口化的应用程序能在一个完全窗口化的环境中运行，而且更便于建立临时性的应用程序。更重要的，在 Windows 95 中新增控制台表明，基于文本的

应用程序仍将得到重视，并且可作为 Windows 整体环境的一部分进行管理。从根本上来说，控制台窗口的加入进一步完善了 Windows 应用程序环境。

4. 平面寻址

Windows 95 应用程序至多可有 4GB 虚存供使用。更进一步说，该地址空间是平面式的。同 Windows 3.1、DOS 以及其他 8086 系列操作系统不一样(这些操作系统都采用分段式内存)，Windows 95 视内存为线性分布。由于内存被虚拟化了，所以每一个应用程序都可使用比它可能想要的内存还要多的内存。请注意，向平面寻址的转变对程序设计人员来说是透明的，它解除了以前处理老式分段内存的负担。

5. 消息和参数类型方面的改变

由于 Windows 95 已转向 32 位寻址，所以传递给 Windows 程序的一些消息将有所不同。另外，用来说明窗口函数的参数类型也由于向 32 位寻址的转变而发生了变化。具体将在本书后面使用到它们时再作介绍。

6. 新的公共控件

Windows 3.1 支持一些标准的控件，如命令按钮、核选框、单选钮以及编辑值等等。而 Windows 95 除了仍支持这些标准控件外，还定义了一些新控件。Windows 95 新增加的控件称为公共控件，包括工具条、工具箱(tool tips)、状态条、进展条、跟踪条以及树形查看等。新控件进一步改进了用户界面，并使应用程序看起来更"现代化"。

1.5 NT 环境

大家可能听说过 Windows NT，也许还使用过它。Windows NT 是 Microsoft 推出的基于 Windows 的高级操作系统。Windows NT 在许多方面与 Windows 95 一样，它们都支持 32 位平面寻址，都支持基于线程的多任务，都支持基于控制台的用户界面。但是，Windows 95 不是 Windows NT。比如说，Windows NT 使用了特殊的方法来实现客户机/服务器式操作系统，而 Windows 95 则不然。Windows NT 是一个全安全系统，而 Windows 95 不是。虽然有许多 Windows NT 开发技术用到了 Windows 95 的开发过程中，但这两者还是不同的。

1.6 学习什么和如何学习

本书中所举的示例都是用 Microsoft Visual C++ 2.0 编译的。学习完本书后，大家也许将能自己挑选一种编译器来建立 Windows 95 程序。应当用任一种兼容于 Windows 95 的编译器编译本书中所举的示例。这些示例是用标准的 C/C++ 编写的，所以可用大多数 C/C+

+编译器进行编译。

1.5 [illegible]

在编写 Windows 95 程序时，首先碰到的可能就是 Windows 3.1 程序向 Windows 95 的转化。为了帮助进行这种转化，本书给出了许多要点。从根本上来说，Windows 3.1 程序向 Windows 95 转化并不困难，本书着重介绍了其中的一些重要区别。