OSBORNE

Windows 95 Programming

nuts & BOLTS

程序设计必读

FOR

Experienced Programmers

HERBERT SCHILDT

科学出版社
龍門書局
希望

Windows 95 程序设计必读

Herbert Schildt 著

亦鸥翻译组 译

燕卫华 校

科学出版社

龙门书局

1996

(京)新登字 092 号

内 容 简 介

本书是学习进行 Windows 95 程序设计的高级读物，全书包括 Windows 95 基础、消息和基本 I/O、使用菜单、对话框、更多的控件、管理客户区、Windows 95 的公共控件、上下控件和轨迹条、基于线程的多任务系统、图形的使用等内容，并在附录中给出了 Windows 3.1 转换建议。

本书对软件开发人员和 Windows 95 用户具有重要的参考价值。

需要本书的读者，可与北京海淀 8721 信箱书刊部联系，邮码 100080，电话 2562329。

Herbert Schildt
WINDOWS 95 PROGRAMMING NUTS & BOLTS
FOR EXPERIENCED PROGRAMMERS
McGraw-Hill, 1995

版 权 声 明

Windows 95 程序设计必读

Herbert Schildt 著
亦鸥翻译组 译
燕卫华 校
责任编辑 汪亚文
科 学 出 版 社
龙 门 书 局 出版
北京东黄城根北街 16 号
邮政编码：100717
双青印刷厂印刷
新华书店北京发行所发行 各地新华书店经售
*
1996 年 2 月第 一 版 开本：787×1092 1/16
1996 年 2 月第一次印刷 印张：14 1/4
印数：1—5000 字数：325 000
ISBN 7-03-004918-7/TP・478
定价：21.00 元

前　言

本书旨在指导读者学习如何编写 Windows 95 下的程序。Windows 95 现在已经成为最重要的操作系统。这就是编写本书的原因。

首先，Windows 95 不再使用 DOS 作为其基础环境，因为 Windows 95 不再需要它了，DOS 不再被看作是必要的操作系统。当然，考虑到目前的用户基础，在以后的几年中还要对 DOS 进行维护和作一些小的更新，但不会有新的用户去选择 DOS 而不选 Windows 95 的。随着 DOS 的死亡，基于文本的用户界面也要死亡了。虽然还会有人编写基于文本界面的程序，但大多数新程序都要按 Windows 95 提供的图形化用户界面来进行设计。

其次，Windows 95 解决了过去这些年中困扰程序员们的一些基础问题。例如，Windows 95 最终放弃了段内存模式，而代之以浮动寻址。更进一步，它分配给每个进程以多达 4 兆的系统虚拟地址空间！这正是程序设计人员们所期望的。Windows 95 也支持基于线程的多任务、独立消息队列以及大量的新的公用控件（例如工具条和状态窗口）。与 Windows 3.1（它看起来似乎在挡程序设计的道）不同的是，Windows 95 提供了鼓励人们编写更大程序的特性。我们终于有了构建下一代软件的平台。

第三个原因是 Windows 95 所具有的神奇效果：它提供了好得多的用户界面，而这些用户界面又是非常受欢迎的。Windows 95 的用户界面令人耳目一新，易于使用。另外，使用工具条和以 Explorer 来代替旧的文件管理器使 Windows 95 更易于使用。同时 Windows 95 也比 Windows 3.1 更稳妥，因为它极少崩溃！即使作为程序设计人员的我们能够容忍经常性的崩溃，但对最终用户来说，软件的失败是令人讨厌甚至是危险的。Windows 95 的稳妥性也使它更受用户的欢迎。

在读完本书后，您也许已经能够为这个令人激动的新环境编写程序了。

本书是专为移植到 Windows 95 的熟练程序设计人员而写的。我们假设读者已经能够为其他操作系统（例如 DOS 和 UNIX）编写程序，并且假设读者是一个熟练的 C 或 C++程序设计人员。虽然本书也包括了所有必备的 Windows 95 程序设计基础知识，但并没有深入介绍程序设计基础。

Windows 95 是在 PC 机上最灵活和功能最强大的通用操作系统，它也是非常复杂的。尽管有这些复杂性，Windows 95 也是现有最完美、彻底的操作系统之一，它在逻辑上子系统与另一子系统是一致的。当读者掌握了它的要领并建立了一段可以重用的代码段之后，就很容易与它打交道了。学习编制 Windows 95 下的程序是绝对值得花时间和精力的。

最后要说的是，如果读者以前从来没有编写过 Windows 风格的程序，最好耐心些！Windows 程序与你以前编写过的程序是很不相同的。

本书中的所有代码都是用 Microsoft Visual C++版本 2 细心地编写、编译和测试过的。读者需要备有能生成与 Windows 95 兼容的目标代码的这种编译器，或者备有其他 C/C++编译器。

本书中包含了许多有用和有趣的程序。如果读者像我一样，那么会很想试一试这些程

序，但又讨厌将这些代码重新敲入计算机。当我键入一本书中的程序代码时总会出错，而调试这些程序还要花费大量的时间。对于 Windows 95 程序尤其如此。因此，作者提供了一张包含了本书中所有程序的磁盘①，只需花费 24.95 美元即可得到。读者可以与下列地址联系：

Herbert Schildt
398 Country Rd 2500N
Mahomet, IL 61853
Phone: (217)586-4021
Fax: (217)586-4997
USA

① 本书中文版不含配套磁盘——译者注。

目 录

前言

第一章 Windows 95 基础 …… 1

1.1 什么是 Windows 95 …… 1

1.2 Windows 95 与 Windows 3.1 …… 2

1.3 与 Windows NT 相比 …… 5

1.4 Windows 95 程序是独特的 …… 6

1.5 Windows 95 和用户程序间的相互作用 …… 6

1.6 Win32：Windows 95 API …… 7

1.7 窗口构成 …… 7

1.8 Windows 95 应用程序基础 …… 7

1.9 Windows 95 框架 …… 9

1.10 窗口函数 …… 18

1.11 命名规则 …… 18

第二章 消息和基本 I/O …… 20

2.1 消息框 …… 20

2.2 理解 Windows 95 消息 …… 22

2.3 键击响应 …… 23

2.4 输出文本到窗口中 …… 25

2.5 设备环境 …… 29

2.7 处理 WM_PAINT 消息 …… 31

2.8 产生 WM_PAINT 消息 …… 35

2.9 响应鼠标消息 …… 38

第三章 使用菜单 …… 44

3.1 介绍菜单 …… 44

3.2 在应用程序中包含菜单 …… 47

3.3 响应菜单选择 …… 47

3.4 菜单程序示例 …… 48

3.5 使用菜单快捷键 …… 51

3.6 加载快捷键表 …… 53

3.7 动态管理菜单 …… 58

3.8 建立动态菜单 …… 64

第四章 对话框 …… 70

4.1 对话框使用的控件 …… 70

4.2 模态和非模态的对话框 …… 70

4.3 接收对话框消息 …… 71
4.4 激活对话框 …… 71
4.5 释放一个对话框 …… 71
4.6 建立一个简单的对话框 …… 72
4.7 第一个对话框示例程序 …… 74
4.8 增加列表框 …… 78
4.9 增加一个编辑框 …… 82
4.10 使用非模态对话框 …… 90
第五章 更多的控件 …… 97
5.1 滚动条 …… 97
5.2 复选框 …… 107
5.3 单选钮 …… 109
5.4 复选框、单选钮及滚动条的演示程序 …… 109
5.5 静态控件 …… 117
5.6 独立的控件 …… 118
第六章 管理客户区 …… 120
6.1 显示一个位图 …… 120
6.2 重画客户区 …… 125
6.3 文本管理 …… 132
6.4 虚拟窗口程序的改进 …… 139
第七章 Windows 95 的公共控件 …… 141
7.1 引入并初始化公共控件 …… 141
7.2 公共控件是窗口 …… 142
7.3 使用工具条 …… 142
7.4 工具条示例程序 …… 146
7.5 GetOpenFileName()和 GetSaveFileName() …… 157
7.6 关于在窗口程序中使用文件的一点说明 …… 160
7.7 公共对话框 …… 161
第八章 上下控件和轨迹条 …… 162
8.1 上下控件的使用 …… 162
8.2 轨迹条的使用 …… 164
8.3 选值控件以及轨迹条的演示 …… 167
第九章 基于线程的多任务系统 …… 179
9.1 创建多线程程序 …… 179
9.2 线程的优先级 …… 185
9.3 线程间的同步 …… 187
9.4 其他的同步机制函数 …… 193
9.5 创建一个独立的任务 …… 194
第十章 图形的使用 …… 197

10.1 图形坐标系统…… 197
10.2 画笔和刷子…… 197
10.3 像素的设置…… 198
10.4 画线…… 198
10.5 当前位置的设置…… 198
10.6 画弧…… 199
10.7 矩形的显示…… 200
10.8 椭圆和馅饼图…… 200
10.9 对画笔进行处理…… 201
10.10 定制刷子的创建 …… 202
10.11 定制对象的删除 …… 202
10.12 输出模式的设置 …… 202
10.13 一个图形演示 …… 204
10.14 下一步是什么 …… 214
附录 A Windows 3.1 转换建议 …… 217

第一章 Windows 95 基础

在进行 Windows 95 编程前，读者必须基本了解 Windows 95 的运行方式、它和用户程序如何相互作用、Windows 95 应用程序的基本元素，以及编程时所必须遵循的规则。弄清 Windows 95 和其前身 Windows 3.1 的区别也是非常重要的。为此，本章中首先对 Windows 95 作一个简略而完整的介绍，讨论了它和前身的相同与不同之处，并以一个 Windows 95 框架程序结束本章。此程序作为本书后面所给出程序的基础。

Windows 95 是一个庞大而复杂的编程环境，在一本书中是无法全部完整叙述的。本书只涉及 Windows 95 编程的一些基本元素，它们适用于所有程序，并被经常使用，或是 Windows 95 专有的重要的更新之处。实际上，本书提供了进入 Windows 95 编程世界的入门捷径。读者学习本书之后，将对 Windows 95 具有了充分的理解，足以编写自己的程序并进一步开发其子系统。

注意：如果你从未编写过 Windows 程序，那么本书中的大多数内容都将是崭新的。请耐心向下阅读。即使是一个熟练的程序员，在开始时也会对 Windows 95 中的许多基本概念感到困惑不解。若你有过 Windows 3.1 编程的经历，就会掌握得快一些，但要细心。Windows 3.1 和 Windows 95 之间存在差别，它影响了用户编写程序的方式。

1.1 什么是 Windows 95

Windows 95 是直到下个世纪初的 PC 机操作系统的一部分。读者可能知道，Windows 95 提供了使用平台模型的图形用户界面。它支持鼠标和键盘作为输入设备，克服了其前身 Windows 3.1 的一些局限和不足，同时增加了新的功能并提供了新的改进了的用户界面。

Windows 95 最重要的特点大概是它是一个 32 位的操作系统，这样，它就远远摆脱了 16 位操作系统所带来的缺憾和问题。进入 32 位操作系统对用户来说是全透明的，它使 Windows 95 编程变得更加容易。

Windows 95 在设计时首先考虑到了它与 Windows 3.1 及 DOS 以及可在它们下面运行的程序的兼容性。也就是说，Windows 95 与大量已存在的 PC 应用程序向前兼容，这样，Windows 95 下可运行三种类型的程序：DOS 程序、Windows 3.1 程序以及 Windows 95 程序，它为用户运行的程序正确建立了类型合适的环境。例如，若运行的是 DOS 程序，Windows 95 将自动建立一个窗口化的命令提示；程序在其中运行。

让我们来看一看 Windows 95 的其他一些重要功能吧。

1.1.1 Windows 95 使用基于线程的多任务处理

读者可能知道，Windows 95 是一个多任务的操作系统，它可以并行运行两个或更多个

程序。当然，程序是共享 CPU 而不是真正技术上的同时运行，但由于计算机的速度较快，看起来是在并行处理的。Windows 95 支持两种格式的多任务处理：基于进程型和基于线程型。进程(process)是一个正在运行的程序，由于 Windows 95 可以进行多任务处理，它可以同时运行多于一个的程序。这样，Windows 95 必然支持传统的、基于进程的多任务处理。

Windows 95 多任务处理的第二种类型是基于线程型。线程(thread)是可执行代码的不可拆散的单位。线程这个名字起源于“执行调度”的概念。所有的进程必须具有至少一个线程。然而，Windows 95 的进程可以具有多个线程。

由于 Windows 95 的多任务线程和每个进程可有多个线程，意味着每个进程可能具有其自身的两块或多块在同时运行，实际表现正是如此。因此，使用 Windows 95，就可能同时给多个程序和单个程序的多片段分配多任务。这样，就可以编写出效率非常高的程序。

1.1.2 Windows 95 基于调用的界面

读者若熟悉 DOS 环境，就会知道访问 DOS 的程序使用了不同的软件中断。例如，标准 DOS 中断是 0x21。尽管使用软件中断去访问 DOS 服务是可接受的(DOS 操作系统有一些局限性)，但作为与像 Windows 95 这样的全功能、多任务的操作系统接口的手段则是完全不合适的。Windows 95 如同其前身 Windows 3.1，使用的是基于调用的接口(call-based interface)来访问操作系统。

Windows 95 基于调用的接口使用一组丰富的系统定义函数来调用操作系统的功能。整体上讲，这些函数称为应用程序编程接口(API)。API 包含了数百个函数供用户应用程序调用以与 Windows 95 通信，它们包括了所有必须的与操作系统相关的活动，诸如内存分配、向屏幕输出、建立窗口等等。

1.1.3 动态链接库(DLL)

由于 API 包含了数百个函数，读者可能会想到每个为 Windows 95 所编译的程序中都链接了大量代码，这样每个程序中都含有许多重复的代码，但事实上却不是这样。Windows 95 API 函数包含在动态链接库中，每个程序在执行时才调用它。下面介绍一下动态链接的操作方式。Windows 95 API 函数以可重新分配的格式存储在 DLL 中。在程序编译阶段，用户程序调用 API 函数时，链接器并不把该函数代码增加到用户程序的可执行段中，而是增加该函数的加载指令，诸如所处的 DLL 名及函数名。用户程序执行时，所需的 API 例程由 Windows 95 载入器调用。这样，每个应用程序不必含有实际的 API 代码，只有在应用程序装载进入内存中执行时，API 函数才被加入。

动态链接能够带来许多好处。首先，由于所有的程序都要使用 API 函数，DLL 防止了假若 API 函数被实际增加到每个程序的可执行文件时，由于建立了许多重复的目标代码而造成的磁盘空间浪费。其次，更改并升级到 Windows 95 可以通过改变动态链接库例程来完成，这样已存在的应用程序就不需重新编译。

1.2 Windows 95 与 Windows 3.1

由于阅读本书的大多数读者都可能熟悉 Windows 3.1，将其与 Windows 95 简要对比一

下应该是顺理成章的事。尽管自从 1985 年推出 Windows 以来，Windows 95 是 Windows 产品系列的下一阶段，它还代表了操作系统发展的主要方向，但从程序员的角度而言，编程方法都是相近的。

对于熟悉 Windows 3.1 的读者来说，学习使用或为 Windows 95 编程不是一件难事。从用户的角度而言，Windows 95 增加了一个改进了的界面并朝着以文档为中心的组织更进了一步。特别是一些基本项目，诸如 Program Manager 和 File Manager 被 Start 菜单和 Explorer 所代替。但读者若运行过 Windows 3.1，就会对 Windows 95 感到很熟悉。它们的操作方式是相同的。

从程序员的角度来看，Windows 95 编程方式与 Windows 3.1 是相同的。Windows 95 保留了原 Windows API 函数的命名空间，在增加功能时，通常都是采用增加新功能的方法完成。尽管 Windows 95 和 Windows 3.1 存在一些差别，但这些差别的大部分都是易于理解的。而且，旧的 Windows 3.1 程序在 Windows 95 下运行良好，所以不必更改旧的应用程序。

下面几节更详细地讨论了 Windows 3.1 和 Windows 95 之间的区别。

1.2.1 用户差异

从用户的角度来看，Windows 95 主要在以下四个方面与 Windows 3.1 不同：

- 改变了平台界面
- 变换了窗口风格
- 适用于应用程序的新的控件元素
- 不再需要 DOS

如前所述，Windows 3.1 中的 Program Manager 被 Start 菜单取代。而且，平台中当前还含有 Task Bar(任务条)。Task Bar 中显示了系统中所有激活任务的清单，用户使用鼠标即可进行任务切换。大多数用户将会发现 Start 菜单和 Task Bar(相比于原先的 Program Manager)有明显的改进。

Windows 95 下的窗口外观已被重新设计。对于大多数用户来说，新的外观更加符合时尚和方便。Windows 3.1 的缺点之一是它的外观显得过于传统，而 Windows 95 就改进了这一点。同时，在使用 Windows 95 应用程序时，可以看到一些新的控件，诸如工具条、选值控件、树和状态条。用户可使用这些控件更加方便地设置与程序相关的各种属性。

Windows 95 不再需要 DOS。读者或许知道，Windows 3.1 并不是一个完全独立的操作系统，它必须在 DOS 的基础上运行，因为 DOS 提供了对文件系统的支持。Windows 95 则是一个完整的操作系统，它不再需要 DOS。但 Windows 95 中仍提供了对 DOS 程序的支持。实际上，一些 DOS 程序在 Windows 95 下要比在 DOS 下运行状况还要好。

Windows 95 还增加了一些附加的功能，包括完全透明地运行 DOS 程序的性能。用户运行 DOS 程序时，将自动建立一个窗口化的命令提示界面。这个窗口化的命令提示完全集成到 Windows 95 图形界面中。例如，直接从提示下可以运行 Windows 程序。而在 Windows 3.1 中，只有在 Windows 下才能运行 Windows 程序。

Windows 95 的另一个新功能是支持长文件名。DOS 和 Windows 3.1 只允许 8 个字符的文件名后跟 3 个字符的扩展名。Windows 95 允许文件名最多可达 255 个字符。

Windows 95 中还包含了一组 Windows 3.1 所不具有的附件和管理工具。例如，支持便携

机、e-mail、网络和远程机。它还支持“即插即用”(plug and play),即允许很轻易地安装新的硬件构件。

1.2.2 编程差异

从程序员的角度来看,Windows 3.1 和 Windows 95 之间存在两个主要的差别。首先,Windows 95 支持 32 位寻址并使用虚拟内存。Windows 3.1 使用 16 位的分段寻址方式。对于很多应用程序这个区别起不到什么作用。但对于一些程序,32 位寻址的作用将是显著的。编程者将会发现 Windows 95 32 位的内存模型使编程更加方便。

第二个差别和完成多任务处理的方式有关。Windows 3.1 在任务切换时使用的是非占先方式。这意味着为运行另一任务,一个 Windows 3.1 任务必须靠手动返回控制到调度器下,也就是说,Windows 3.1 程序保留了对 CPU 的控制直至其自身决定放弃它。这样,一个出现了错误的任务将独占 CPU,导致死机。相对而言,Windows 95 使用了占先式的、基于时间片的任务处理。这种调度方式下,任务自动由 Windows 95 所预占,CPU 自动分配给下一任务。占先式的多任务处理是一种比较先进的方式,因为它允许操作系统完全控制任务分配并防止了某一任务独占 CPU。这不能不说是一种进步。

除了上面两种主要的变化之外,Windows 95 和 Windows 3.1 还在一些小的方面存在差别,详见下文。

1. 输入队列

在输入队列(input queue)方面 Windows 3.1 和 Windows 95 之间也存在差别。输入队列存储了诸如敲键或鼠标活动之类的消息,直至它们被发送到程序中。在 Windows 3.1 中,对于系统中所运行的所有任务只有一个输入队列,但 Windows 95 则是对于每个线程都分配一个属于其自身的输入队列,这样做的好处在于进程不会因为对消息反应太慢而降低了系统的性能。

尽管多个输入队列是一个重要的补充,但此变化对于 Windows 95 编程却是没有影响的。

2. 线程和进程

Windows 3.1 只支持基于进程的多任务处理,即进程是 Windows 3.1 的不可分割的最小单位。如前所述,Windows 95 支持线程型和进程型的多任务。尽管旧的 Windows 3.1 程序不需要作任何改变就能够在 Windows 95 下正常运行,但有些程序员仍然可能想增强它们,以利用基于线程的多任务处理的优势。

3. 控制台

过去,基于文本的(即非窗口化的)应用程序在 Windows 下使用是很不方便的。但 Windows 95 支持了一种特殊类型的称为控制台(console)的窗口,它提供了标准文本型、命令提示环境。除了是文本型之外,控制台的操作和管理都和其他窗口相同。增加文本型控制台不仅允许非窗口化应用程序能在全 Windows 环境下运行,而且更加方便于建立短的、简捷的应用程序。但更重要的意义可能还在于,在 Windows 95 中包括控制台是对一些基于文本

型应用程序的最终承认，这样这些程序就可以作为整个 Windows 环境的一部分来管理。本质上，控制台窗口的增加也完善了 Windows 应用程序环境。

4. 平面寻址和虚拟内存

Windows 95 的应用程序可以在 4 吉字节(Gb)的虚拟内存中运行，而且这个寻址空间是平面型的。不同于 Windows 3.1 和其他一些 8086 族操作系统所使用的分段管理内存的方式，Windows 95 把内存看作线性的。而且由于可以把内存虚拟化，每个应用程序都可以使用其需要数量的内存。改变到平面寻址方式对编程者来说使内存管理更加透明化，而且减轻了由旧的分段方式所造成的繁琐和困扰。

由于转变为平面的、32 位的寻址方式，每个 Windows 95 进程都可以在其自身的寻址空间内运行并与其他进程隔离开来。这样在一个进程崩溃时，其他进程将不受影响。也就是说一个出现了错误的程序不会造成整个系统的失效。

5. 消息和参数类型的改变

由于 Windows 95 改变为 32 位的寻址方式，一些传递给 Windows 95 程序的消息可能与传递给 Windows 3.1 程序时不同。而且用于声明窗口函数的参数类型也已经改变。消息和参数类型的改变将在本书的后面讨论。

6. 新的公用控件

如同 Windows 3.1，Windows 95 也支持标准的控件，诸如按钮、复选框、单选钮、编辑框等等。除了这些标准的控件之外，Windows 95 还支持一些新的控件，称为公用控件(common control)。公用控件包括工具条、工具触点、状态条、进度条、跟踪条和树等。使用新的公用控件使用户的应用程序更加具有现代的外观和感觉，使 Windows 95 程序更胜一筹。

7. 可安装的文件系统

在 DOS 和 Windows 3.1 之下，文件系统是通过中断 0x21 访问的。文件系统使用的是 16 位代码并在实模式下运行。但 Windows 95 所使用的是 32 位保护模式的可安装文件系统。Win 95 提供了一个可安装的文件系统管理器以协调对文件系统和设备的访问。由于新文件系统运行在保护模式下，不需要花费转换到 16 位实模式下以访问文件系统的时间(Windows 3.1 必须在两种模式之间转换)。这样就获得了较高的文件系统性能。由于文件系统的访问通常都是通过使用与 Windows 95 兼容的 C/C++编译器所提供的高级函数，这个升级并不改变在用户文件中处理文件的方式。但它提高了所编写的程序的性能。

1.3 与 Windows NT 相比

读者可能听说过 Windows NT，也可能使用过它。Windows NT 是 Microsoft 的高起点的基于 Windows 的操作系统，它和 Windows 95 有很多相同之处。它们都支持 32 位平面寻址，都支持基于线程型的多任务处理，而且都支持基于控制台的界面。但 Windows 95 不是 Windows NT。例如，Windows NT 使用基于客户/服务器模型的操作系统补充方式，而

Windows 95 就不是这样。Windows NT 支持全安全性的系统，Windows 95 并不支持。尽管为 Windows NT 使用所开发的许多基本技术都可以在 Windows 95 看到踪影，但它们并不是一样的。

1.4 Windows 95 程序是独特的

读者以前若从未编写过 Windows 程序，可能会对 Windows 程序的编程方式感到奇怪。Windows 程序的编程方式可能和你以前使用的结构方式是不同的。Windows 风格程序的独特结构由两个约束所构成。其一是由用户程序和 Windows 交互作用的方式所决定，其二是由建立一个标准的、Windows 风格应用程序界面所必须遵循的规则所制约。

Windows 95(以及所有的 Windows)的设计目标是使一个基本熟悉系统的人在不需培训的情况下能坐在计算机前运行任务应用程序。为此，Windows 向用户提供了一个基本不变的界面。在理论上，你若能运行一个基于 Windows 的程序，就一定能运行全部程序。实际上，大多数程序仍然需要一些培训，至少是告诉他程序的作用，但不需告诉他如何和程序相互配合。事实上，Windows 应用程序中的大部分代码都是用于支持用户界面的。

尽管建立一个不变的、Windows 风格的界面是编写 Windows 95 应用程序的关键部分，但它并不能自动完成。编写一个不利用 Windows 界面元素的 Windows 程序是可能的。要建立一个 Windows 风格的程序，就需要使用本书中所介绍的技术。只有那些编写用来利用 Windows 的程序看起来才像 Windows 程序。尽管你可以越过基本的 Windows 设计理论不用，但请在具有非常充分的理由时才这么做，因为你的用户将会对使用中的差异而感到困扰。总之，若你设计的是 Windows 95 程序，务必使用通常的 Windows 界面并符合标准的 Windows 设计实例。

1.5 Windows 95 和用户程序间的相互作用

当你为多种操作系统编写程序时，正是你的程序引发了与操作系统间的相互作用。例如，在 DOS 程序中，正是需要诸如输入输出之类进程的程序。所编写的程序都以一种传统的方式来调用操作系统，而不是操作系统来调用程序。但从大的方面来看，Windows 95 使用的却是一种与上述相反的方式，是 Windows 95 来调用用户程序。过程大致如下：Windows 95 程序等待直至由 Windows 发送给它一个消息。消息通过 Windows 所调用的特殊函数传递到程序中。一旦收到了消息，用户程序就开始采取适当的操作。尽管程序在响应消息时可以调用一个或多个 Windows API 函数，但这种活动仍然是由 Windows 起始的。总之，与 Windows 95 的基于消息的交互作用构成了所有 Windows 95 程序的总体方式。

Windows 95 发送给用户程序的消息类型是很多的。例如，每次鼠标在属于用户程序的窗口中单击时，鼠标单击消息将发送到程序中。每次窗口收缩时，将发送另一种类型的消息。当用户程序在等待输入时，每次用户击键也将发送一种类型的消息。务必记住：就程序而言，消息是随机到达的。这就是 Windows 95 程序很像中断驱动程序的原因。你无法知道下一个消息是什么。

1.6 Win32：Windows 95 API

如前所述，Windows 环境是通过一个称为 API(应用程序接口)的调用型接口来访问的。API 中存在数百个函数供用户程序调用。API 函数提供了由 Windows 95 执行的所有的系统服务，其子集之一称为图形设备接口(GDI)，它是 Windows 提供与设备无关的图形支持的部分。GDI 函数使 Windows 应用程序在不同硬件上运行成为可能。

Windows 95 程序使用了 Win32 API。Win32 的大部分是旧的 Windows 3.1 API(Win 16)的超集。实际上，大部分函数都是使用相同的名字来进行调用，使用的方式也是相同的。尽管它们风格和目的方面是相同的，两个 API 之间仍存在差别，因为 Windows 95 支持 32 位的平面寻址，而 Win 16 只支持 16 位的分段内存模型。这个差别使一些 API 函数被扩展以接受 32 位的参数并返回 32 位数值；而且，还有一些 API 函数必须被改变以适应 32 位的结构。另外还增加了一些 API 函数以支持新的多任务处理方式、新的界面元素以及其他增强了的 Windows 95 功能。如果读者初次接触 Windows 编程，这些变化的影响将不会太大。但如果你是在将 Windows 3.1 的代码移植为 Windows 95，就必须仔细检查传递给 API 函数的每个参数。

由于 Windows 95 支持 32 位的寻址，意味着整数是 32 位长度。这样就不存在 Windows 3.1 中的 16 位长度的 unsigned 整数类型，因为 int 和 unsigned 类型的整数都是 32 位的。若想使用 16 位长度的整数，就必须把它声明为 short(这些类型的可移植的 typedef 名是由 Windows 95 提供的)。因此在移植 16 位环境下的代码时，就仔细检查所使用的整数，因为它们将自动从 16 位扩展为 32 位而导致副作用。

32 位寻址的另一个结果是指针不必声明为 near 或 far。任何指针都可以到达内存区的任何部分。Windows 95 中，near 或 far 都没有什么作用。这样在移植程序到 Windows 95 下时，可不管程序中指针的 near 或 far。

1.7 窗口构成

在讨论 Windows 95 编程之前，我们先定义一些重要的术语。图 1.1 中给出了一个标准窗口，其中指示了组成元素。

所有的窗口都具有边界用于限定窗口区并可用于改变窗口的大小。在窗口的顶部存在一些项目。最左边的是系统菜单图标，也称作标题条图标。单击该图标时将显示系统菜单。系统菜单框的右边是窗口标题。最右端是缩小、放大和关闭图标。以前的 Windows 版本中不含有关闭图标，这是 Windows 95 的革新之处。客户区是用户程序活动发生的窗口部分。大多数窗口还具有水平和垂直滚动条，用于移动窗口中的文本。

1.8 Windows 95 应用程序基础

在开发 Windows 95 应用程序的框架之前，有必要先讨论一下适用于所有 Windows 95 程序的一些基本概念。如果读者已经知道怎样编写 Windows 3.1 程序，那么对本节和以下几

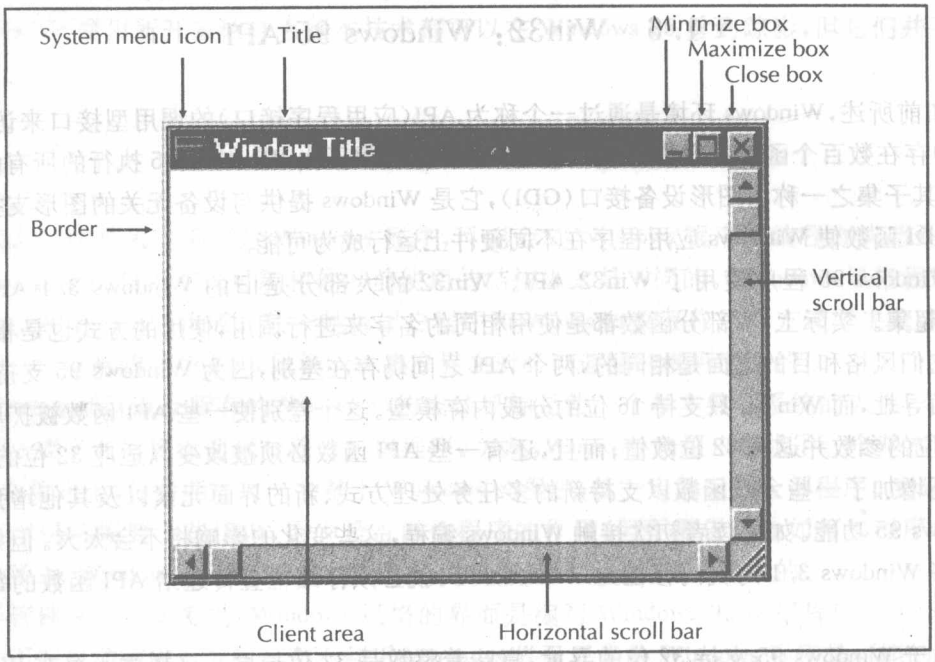

图 1.1 标准窗口的元素

节所介绍的内容应该是很熟悉的(实际上,Windows 3.1 和 Windows 95 程序在表面上看起来是相同的)。但我们仍希望你能阅读一下这些内容,因为 Windows 3.1 和 Windows 95 间存在一些重要的差别。

1.8.1 WinMain()

所有的 Windows 95 程序都以执行一个对 WinMain()的调用开始。通常,Windows 程序不含有 main()函数。WinMain()具有一些特殊的性质,使之和应用程序中的其他函数区分开来。首先,它必须使用 WINAPI 的调用约定进行编译(你也可能看到使用 APIENTRY。它们是同一件事)。缺省时,C 或 C++程序中的函数使用 C 调用约定。但也可能编译一个函数以使它使用一个不同的调用约定。例如,通常也可以使用 Pascal 调用约定。由于一些技术上的原因,Windows 95 用于调用 WinMain()的调用约定是 WINAPI。WinMain()的返回类型应该是 int。

1.8.2 窗口函数

所有的 Windows 95 程序必须含有一个特殊的函数,它不是由用户程序调用,而是由 Windows 95 调用。此函数通常称为窗口函数或窗口进程。窗口函数在 Windows 95 需要向用户程序传递消息时调用。Windows 95 通过该函数与用户程序联系。窗口函数在其参数中接收消息。所有的窗口函数的返回类型必须被声明为 LRESULT CALLBACK。LRESULT 类型是一种长整数名的 typedef,CALLBACK 调用约定用于那些由 Windows 95 所调用的函数。在 Windows 术语集中,所有由 Windows 调用的函数都称为回调函数(callback function)。

除了接收由 Windows 95 发送的消息之外，窗口函数必须初始化由一个消息所标明的活动。通常，一个窗口函数中含有一个 switch 语句，它连接了对程序即将响应的每个消息的特定响应动作。用户程序中不必对 Windows 95 发送的每个消息作出响应。对于程序中不必过问的消息，可让 Windows 95 提供缺省处理。由于 Windows 95 可能产生数百种不同的消息，大多数消息由 Windows 95 进行简单处理是很正常的。

所有的消息都是 32 位整数值，而且，所有消息都和消息需要的附加信息相连接。

1.8.3 窗口类

用户的 Windows 95 程序第一次开始执行前，需要定义和注册一个窗口类(Window class)。这里，类一词并不是它在 C++中所使用的意义，它在该处的意义是格式(style)或类型(type)。在注册窗口类时，Windows 95 将告诉有关窗口的格式和函数。但注册窗口类并不能使窗口实际存在。要想建立窗口还需要附加的步骤。

1.8.4 消息循环

如前所述，Windows 95 通过发送消息与程序相联系。所有的 Windows 95 应用程序都必须在 WinMain()函数之中建立一个消息循环(message loop)。此循环从应用程序消息队列中读取待读的消息，然后将消息发送回 Windows 95，后者将使用消息作为参数来调用用户程序的窗口函数。这个传递消息的过程看起来可能过于复杂，但这是所有 Windows 程序都必须使用的方式。这样做的部分原因是将控制返回 Windows 95 以使调度器能合理分配 CPU 时间，而不是等待应用程序的时间段结束。

1.8.5 Windows 数据类型

读者将会看到，Windows 95 程序并没有全部使用标准的 C/C++数据类型，诸如 int 和 char * 。Windows 95 使用的所有数据类型已在 WINDOWS.H 文件和其相关文件中进行了类型定义。此文件由 Microsoft(以及制作 Windows 95 C/C++编译器的其他公司)所支持，而且必须包括在所有的 Windows 95 程序中。一些常用的数据类型有 HANDLE、HWND、BYTE、WORD、DWORD、UINT、LONG、BOOL、LPSTR 和 LPCSTR。HANDLE 是用作句柄的 32 位的整数。读者将会看到，有一些句柄类型，它们的大小是与 HANDLE 类型相同的。句柄只是标识某个资源的数值，例如，HWND 是用作窗口句柄的 32 位整数。所有的句柄类型必须以 H 开始。BYTE 是 8 位无符号字符。WORD 是 16 位无符号的短整数。DWORD 是无符号的长整数。UINT 是无符号的 32 位整数。LONG 是长整数的另一个名字。BOOL 是一个用于标明正误的数值。LPSTR 是一个字符串的指针，LPCSTR 是一个字符串的常数指针。

除了上述基本类型之外，Windows 95 还定义了一些结构。下面的框架程序中需要的两个是 MSG 和 WNDCLASSEX。MSG 结构存储 Windows 95 消息，WNDCLASSEX 是定义窗口类型的结构。这些结构将在本章后面讨论。

1.9 Windows 95 框架

前面已经介绍了所需的基础知识，现在我们来开发一个短小的 Windows 95 程序。前面