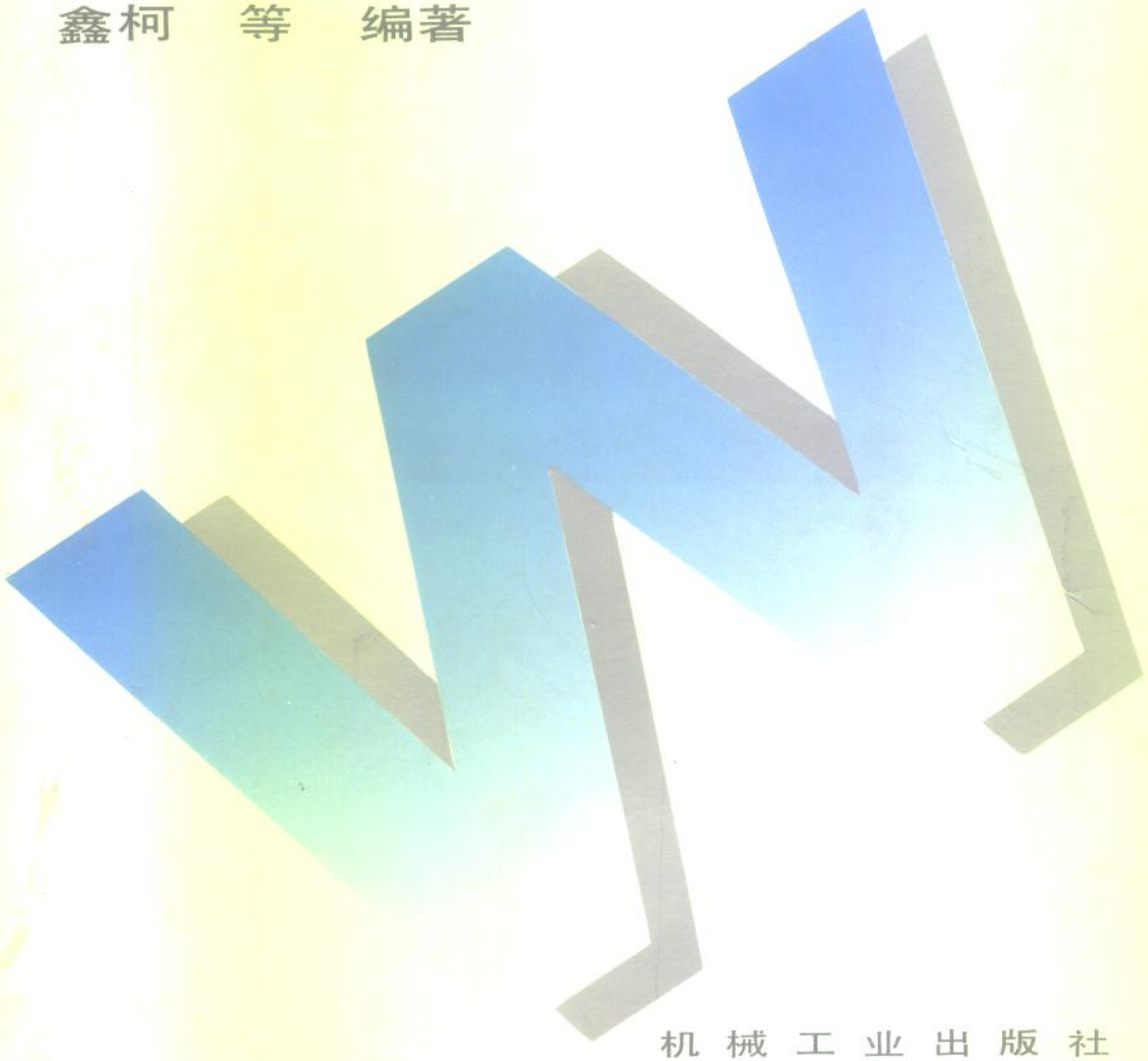
高级 Windows 程序设计技巧

鑫柯　等　编著



机 械 工 业 出 版 社

高级 Windows 程序设计技巧

鑫柯　等编著

机 械 工 业 出 版 社

(京) 新登字 054 号

本书由浅入深地讨论了 Windows 应用程序设计技巧，并给出了大量实例程序。全书共分 15 章，主要内容如下：Windows 程序设计流程、键盘／鼠标／计时器输入、图文输出、内存管理、Windows 资源的设计和使用、剪贴板、多文档界面、动态内存管理、消息分类。

本书内容全面，实例丰富，使用方便，可作为 Windows 程序设计人员的参考书，也可作为计算机应用人员和大中专院校师生的培训教材。

图书在版编目 (CIP) 数据

高级Windows程序设计技巧／鑫柯等编著.–北京：机械工业出版社，1995.3
ISBN 7–111–04573–4

Ⅰ.高… Ⅱ.鑫… Ⅲ.窗口软件–操作系统–程序设计 Ⅳ.TP316

中国版本图书馆 CIP 数据核字 (94) 第 13318 号

出 版 人：马九荣 (北京市百万庄南街 1 号 邮政编码 100037)
责任编辑：王中玉 版式设计：霍永明 责任校对：宁秀娥
封面设计：肖 晴 责任印制：卢子祥

三河市宏达印刷厂印刷 · 新华书店北京发行所发行

1995 年 3 月第 1 版 · 1995 年 3 月第 1 次印刷

787mm × 1092mm 1/16 · 印张 29.75 · 733 千字

0 001—3 000 册

定价：34.00 元

前　言

Windows 是一种具有窗口功能和多任务功能的操作系统。自 Windows 3.1 推出以来，Windows 已风靡全球。对用户而言，Windows 易学易用，而且具有一致的用户界面：画面多彩多姿，图标生动鲜明，颜色明朗活泼，窗口切换速度较快；对程序员而言，Windows 突破了 640KB 内存限制，提供了多任务环境，具有设备独立性、资源丰富、消息驱动式体系结构新颖别致等优点。因此，很多 DOS 用户开始改用 Windows，许多软件厂商也开始开发并推出 Windows 中的应用程序。目前，Windows 已公认为是微机界面的标准。

Windows 使用起来很简单，但编程却很麻烦。为了让读者少走弯路，本书重点讨论 Windows 环境下的应用程序设计，并给出了大量完整的程序。书中许多程序短而精，并清晰地演示了各种 Windows 程序设计技巧，有些程序还是极有用的工具程序。如果读者能熟练使用 Windows，并具有一定的 C 语言编程经验，将更有助于理解和灵活运用这些实例程序。

初学 Windows 程序设计的人经常会面对许多全新的概念而不知所措，针对这种情况，本书先用两章的篇幅讲述 Windows 应用程序的结构、开发步骤及设计原则，并逐条语句地剖析了一个简单的 Windows 应用程序，由此总结出设计 Windows 程序的难点所在。第 3 章讨论桌面编程，即应用程序的图形／窗口用户界面。随后的三章讲述 Windows 输入的处理，如键盘、鼠标、计时器等，中间穿插介绍了 Windows 内存管理技术（第 5 章，读者可以跳过这一章，直到读完后面的章节后再回头体会这一章的内容）。学完这几章之后，读者就可以开始开发简单的 Windows 应用程序了。第 7～9 章描述 Windows 资源，如图标、光标、加速键、控件及对话框等，这几章对桌面编程作了更深入的讨论。第 10、11 章着重讨论图形输出。随后的三章讨论了其它 Windows 组件的用法。最后一章供读者阅读前面的章节时参考，这一章集中指出了前面涉及到的各种 Windows 消息。

值得指出的是，Windows 是一个复杂的图形环境，本书无法囊括它的全部内容。事实上，设计一个大型 Windows 应用程序时很可能需要考虑窗口界面、键盘／鼠标／计时器输入、图文输出(显示及打印)、动态数据交换和链接等问题。限于篇幅，本书重点讨论了桌面编程及输入／输出部分，而没有讨论 Windows 下的数据交换和链接，也没有讨论如何设计 Windows 应用程序的帮助(Help)系统；有关这部分内容，请参考其它书籍。

本书的第 1、2 章由鑫柯执笔，第 3～6 章由顾铁成、何大有执笔，第 7～11 章由胡俊、方兵、李小明执笔，第 12 章由叶青青执笔，最后三章由刘勇、谷贵明、方衡执笔。刘衍波绘制了书中的所有图形。

编　者

目　录

第 1 章　Windows 应用程序基础

Microsoft Windows 自 1985 年秋问世以来的近 10 年中，已成为 PC 机 (80286，80386 和 80486 等) 上最为流行的标准图形环境，这个图形环境具有窗口 (window) 功能和多任务 (multitasking) 功能，并提供了一个易学易用的图形用户界面 (Graphical User Interface，简称 GUI)。本书的目的在于从程序员的角度出发，讨论在这个图形环境下设计应用程序的方法和技巧。

1.1　背景知识

从用户的角度来看，Windows 提供了一个多任务、基于图形的窗口环境，在此环境下可以运行专门为 Windows 设计的程序。这些程序包括 Microsoft Excel (电子表格和商业图表软件)、Microsoft Word (文字处理软件) 及其它应用程序。为 Windows 编写的程序具有一致的外观和命令结构，因此，它比传统的 MS-DOS 程序更加易学易用。用户很容易在不同的 Windows 程序间切换，也很容易在程序间交换数据。Windows 提供了一个使用方便、基于图标的程序管理员 (用于运行程序)、一个文件管理员 (用于维护文件) 和一个打印管理员 (用于管理打印机队列)。这一切，都使 Windows 深受用户的青睐，其销量也稳步上升。

当然，Windows 并非一开始便成功地雄居图形操作系统软件市场。事实上，Windows 的发展过程中出现过若干个升级版本，而只有 Windows 3.0 和 3.1 是 Windows 系列中最为成功的两个版本。Windows 3.0 的画面丰富多采，图标生动鲜明，颜色明朗活泼，并采用了令人耳目一新的系统字体；窗口间的切换速度较快；内存突破 640KB 限制；Windows 应用程序可以直接访问的存储容量高达 16MB 之多；多任务环境下可以运行更多的应用程序；各种附件也相当实用，提供了 Paintbrush，Write，Notepad，Calculator，Solitaire 等工具或游戏软件，另外还有支持网络等功能。

除了 Windows 3.0 提供的所有功能和具备的特色之外，1992 年推出的 Windows 3.1 具有更高的运行效率，更适合初学计算机的用户；检错和纠错方法更为精确；加入了许多新的重要功能；其中提供的 True Type 向量字体非常漂亮，可放大任意倍数而不失真，这更是引人注目的特色。

绝大部分软件厂商在 Windows 这股潮流涌来之际，纷纷将其应用软件移植到 Windows 系统内运行，包括一般商业应用程序、电子报表、数据库系统、文本编辑软件、排版软件、绘图软件等。目前，Windows 已获得绝大多数人的认可，以往在 DOS 操作系统下操作计算机的用户，都已转入 Windows 环境中。

从程序员的角度来看，Windows 提供了丰富的内部例程，使程序员可以使用菜单、对话框、滚动条和其它组件来构造友好的用户界面。Windows 还给出了一种外在的图形程序设计语言，这种语言可以对各种不同的字体进行排版。程序员可以用一种与设备无关的方式来处理键盘、鼠标、显示器、打印机、系统定时器和 RS-232 通信端口。Windows 在各种

不同的硬件配置下有相似的运行效果。

尽管 Windows 原本用来运行专门为它编写的应用程序，但许多为 MS-DOS 编写的程序也能运行于 Windows 中。当然，这些程序无法利用 Windows 的许多优点，但某些情况下，它们也可以窗口化，并与 Windows 程序并发执行。

受这种功能强大的操作系统的吸引，大多数软件设计人员都积极地投入 Windows 应用程序的开发行列，因为他们了解到，Windows 已经带来了革命性的冲击，它为计算机的操作界面指出了新方向。

Windows 的成功并非偶然。Microsoft 公司为这一系列产品花了 8 年多的时间，经过不断改进，Windows 才有今天的局面。大约在 1970 年，由 Xerox Palo Alto Research Center (Xerox PARC) 率先提出"窗口"这种概念。这个研究中心的计算机专家想让计算机变得更友好、更方便。他们发现，如果要使计算机有一个新的突破，一定要改造传统的用户界面。DOS 与用户之间的界面就属于传统界面，这种界面常常使一个初学计算机的人感到茫然无措，于是"窗口"这种概念便应运而生。

窗口能够让用户纵观整个计算机的操作，每个窗口都存储各种不同的信息，屏幕上的窗口可以重叠，又可以相互切换，想看某个窗口时可以切换到那个窗口。当时只衍生出窗口这种概念，但实用性的窗口界面却没有被有效地推广开来。直到 1983 年初，Apple 公司推出 Lisa，才将窗口概念实际呈现在用户面前，但由于价格昂贵，Lisa 的发行量一直很低。

到了 1984 年初，Apple 公司终于推出了 Macintosh，这是第一台突破种种限制的计算机。对一台容易接近的计算机，用户不必学习什么计算机科技知识就能轻松地使用。Macintosh 成功的要素是：造价合理而且改用友好的窗口界面。Macintosh 的确深深地打动了用户，学习使用 Macintosh 要比学习使用 IBM PC 系统轻松愉快，这当然影响到 Microsoft 公司的 MS-DOS 市场。Microsoft 公司认识到 MS-DOS 的传统界面比不上 Macintosh，于是在 1983 年 11 月，Microsoft 公司也投入窗口界面的研究。他们希望在 IBM PC 上也能有一个使用窗口界面的操作环境，这个操作环境将比 DOS 操作系统更能够让用户接受。经过两年的努力，Windows 1.01 版在 1985 年 11 月上市。这个软件是一种多任务环境，同时提供了窗口功能，还有一些标准应用程序。它的运行速度太慢，而且不是每一种硬件和外围设备都与 Windows 兼容，再加上 Windows 应用软件不够多，因此 Windows 1.01 版并没有被用户接受。

1987 年 11 月，Windows 2.0 版问市，这个版本修改了 1.01 版中整齐的并列式窗口排列方式，把它变成重叠式窗口 (overlapped window)，也就是今天我们所常见的窗口排列方式。更多的机型和打印机、键盘、屏幕等外围设备都可与 Windows 兼容，同时增强了键盘与鼠标功能，改善了内存管理，当初妨碍 1.01 版流行的最大因素 (即可在 Windows 环境下运行的应用程序太少) 也有所改善。许多软件厂商渐渐地把其应用软件移植到 Windows 上，使得 Windows 市场崭露曙光。随后，Windows 2.0 版分成两种版本：一种是 Windows / 286，它是在 80286 上运行的版本，另一种是 Windows / 386，它是在 80386 上运行的版本。Windows / 386 主要用来支持 80386 的保护模式 (protected mode)，让 Windows 运行得更平稳、更有效。

Windows 3.0 版是在 1990 年 5 月 22 日举办的一次产品发布会上推出的，该版本将 Windows / 286 和 Windows / 386 结合到一种产品上。Windows 3.0 有了一个很大的改变，

这就是对 80286，80386，80486 微处理器保护方式的支持。这能使 Windows 和 Windows 应用程序访问高达 16MB 的内存。Windows 的"外壳"程序 (程序管理员、任务管理员和文件管理员) 得到了全面改进。

Microsoft Windows 3.1 于 1992 年 4 月推出。Windows 3.1 中具有时代意义的新功能是 Apple 公司和 Microsoft 公司共同研制的 TrueType 无极轮廓字体技术。另外，Windows 3.1 还包含了对多媒体技术 (声音和音乐)、对象链接和嵌入 (OLE) 技术及通用对话框的支持。Windows 3.1 以保护方式运行，需要 80286 或 80386 处理器，且要求至少配置 1MB 内存。

与此同时，Microsoft 公司还扩充了 Windows 应用程序接口 (API) 函数库，并推出了 Windows Software Development Kit 3.1 版 (简称 SDK)。这是一套 Windows 的软件开发工具，让计算机用户甚至软件厂商能够开发 Windows 应用程序。

现在，Microsoft 公司正在推出完全支持 32 位单一地址空间和 32 位指令的 Windows 版本 (Windows NT)。这是一种 32 位 Windows，在使用 Intel 80386 或 80486 微处理器的 MS-DOS 机器上运行，同时又可在基于 RISC 的工作站上运行。

1.2 本书编排

Windows 使用简单，但编程却很麻烦。有志成为优秀 Windows 程序员的人们经常要走一些弯路，才能满足大量程序设计的需求。为了使程序员们少走弯路，本书重点讨论 Windows 环境下的应用程序设计，并给出了大量完整的程序，这也是本书的一大优点。许多程序短而精，并清晰地演示了各种 Windows 程序设计技巧；有些程序稍长，用来显示将各种组件结合在一起的方法；还有些程序是极有用的工具程序。

本书没有讲述如何使用 Windows。如果读者尚无使用 Windows 环境的经验，则需要略费功夫安装并熟悉之。Windows 学习起来并不困难。本书也不致力于指导读者怎样编写 C 程序。在使用 Windows 编程之前，必须先对传统环境 (如 MS-DOS) 下的 C 程序设计有丰富的实践知识。如果对 C 语言过于陌生，则有必要花费一些时间熟悉结构和指针等概念。熟悉 80x86 系列微处理器的分段体系结构有助于读者理解本书的内容。如果读者了解 80x86 在实模式和保护模式下是如何寻址的以及远程指针、函数等预备知识，学习效果将会更好，这对设计程序大有帮助。没有这方面的认识也不要紧，因为这并不影响阅读本书。

本章简单地讲解 Windows 的背景知识和 Windows 应用程序的基础知识，第 2 章介绍简单的 Windows 应用程序，让读者对 Windows 程序的结构和编制流程有一个感性认识，同时引伸出 Windows 程序设计的若干重要概念。当然，我们不可能一开始就探究功能齐全的应用程序的复杂特性，所以，接下来的章节着重讨论如何处理 Windows 程序的各种不同组件，程序员可以将这些组件结合在一起，以形成整个程序。

需要指出的是，本书中的所有程序均可用 Microsoft 或 Borland 编译器进行编译，也可以 C++方式进行编译。尽管本书使用了较少的具体的 C++特征，但是，如果希望以后向代码中添加 C++功能，则不妨先在 C++方式下对程序代码进行编译。要有效地运行 Windows 和本书中的程序，需要以下硬件支持:

(1) 与 IBM PC-AT 兼容的 PC 机：80286 以上的 CPU，至少有 640KB 内存;

(2) 软盘驱动器：容量为 1.2MB，5.25 英寸（1 英寸＝25.4mm，后同）的软盘驱动器或容量为 1.44MB，3.5 英寸的软盘驱动器；

(3) 硬盘驱动器：容量在 40MB 以上；

(4) 一套显示系统：单色 MGA、彩色低分辨率 CGA 或者彩色高分辨率 EGA，VGA 或 Super VGA；

(5) 鼠标：与 MS 鼠标兼容。

各种硬件设备的档次当然越高越好，例如，所使用的 CPU 越快越好，这样可缩短编译程序的时间，又如采用比较高级的 VGA 彩色显示器，可使程序运行时的画面比较漂亮。

软件部分必须具备：

(1) MS-DOS 3.1 版以上；

(2) Microsoft Windows 3.0 版以上；

(3) Borland C++ 3.0 版以上；

(4) Microsoft C／C++ Professional Development System for Windows 7.0。

如果还没有安装编译器，则应知道本书的程序只需要 small 模式下的函数库，也可以使用 Microsoft 和 Borland 之外的 C 编译器，只要这种编译器适合于编译 Windows 程序。大多数普通 C 编译器不能用来编译 Windows 程序。

1.3 Windows 的设计思想

图形用户界面 (GUI，也称为“图形窗口环境”) 是 Windows 的核心，这种类型的用户界面概念始于 20 世纪 70 年代中期。如前所述，先驱性的工作是由 Xerox PARC 完成的。Apple 公司将 Xerox PARC 所做的工作引入主流，并使之流行起来。首先用于运气不佳的 Lisa 中；一年以后，又引入 Macintosh，这种 1984 年 1 月推出的新机型无疑要成功得多。Apple 公司的 Macintosh 目前仍然是 IBM 公司在个人计算机市场上的有力竞争者。虽然 Macintosh 的硬件并非一流，但是它的操作系统方便实用。Mac 机的学习和使用比运行 MS-DOS 的 IBM PC 要容易得多。

自从 Macintosh 问世以来，图形用户界面就如同雨后春笋，在计算机工业界蓬勃发展起来。对于运行 MS-DOS 的 IBM 兼容机，Windows 是其中的独领风骚者；对于运行 UNIX 的机器，X-Windows 系统已成为国际标准；对于 Sun 工作站，Sunview 也拥有大量的市场需求。虽然各种图形用户界面在细节上有所区别，但它们具有相似的特点。

所有图形用户界面都在位图型的视频显示器上显示图形。图形提供了对屏幕实际能力的较好应用，它是一种表达信息的丰富的视觉环境，且为视频图形和为文档打印而准备的格式文本提供了所见即所得 (WYSIWYG) 的可能性。

早期的视频显示器仅用于回显用户用键盘输入的文本。在图形用户界面中，视频显示器成为用户输入的来源。视频显示器以图标和输入设备 (如按钮和卷滚条) 的形式显示各种图形对象。用户可以用键盘 (或更直接的定位设备如鼠标) 直接在屏幕上操纵这些对象。可以拖曳图形对象、按下鼠标按钮以及操纵滚动条。

因此，用户与程序的交流变得更为密切，这不再是一种从键盘到程序、再到视频显示器的单程循环，用户已能与显示器上的对象直接进行交流。

Windows 的设计思想表现在下列几个方面:

(1) 一致的用户界面

长期以来，许多用户对 C:>或 A:>这种命令行式的接口感到厌倦。如今，Windows 提供了一个友好的图形用户接口，用户再也不用花费很长的时间来学习如何使用计算机及掌握新程序了。Windows 对初学者帮助很大，因为所有 Windows 程序都具有基本相同的外观。

当前正运行的程序占据一个"窗口"，即屏幕上的一块矩形区域。图 1-1 就是一个典型的 Windows 应用程序窗口，程序名显示在标题条 (caption bar) 中。大部分操作都是以鼠标来完成或以下拉式菜单来处理的，大部分 Windows 应用程序兼具鼠标和键盘接口，Windows 应用程序还用对话框 (dialog box) 以取得外部输入信息。菜单和对话框使用户可以尝试性地使用新程序，并发掘其特点。

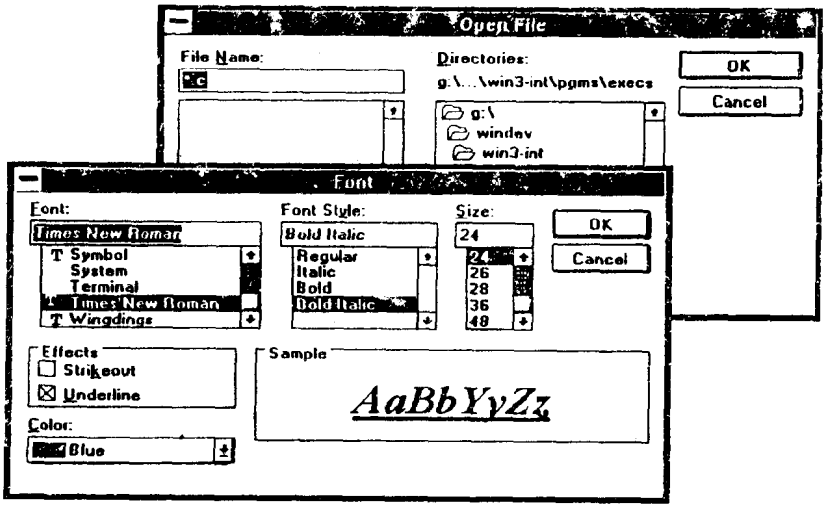


图 1-1 一个典型的 Windows 应用程序窗口

正因为 Windows 应用程序有着大致相同的外观，所以一旦用户熟悉了这种接口方式，就不必再花很长的时间去学习如何操作一个新的应用程序。即使碰到一个从未用过的 Windows 应用程序，也能很轻松地掌握它的使用。接口的一致性大大降低了学习 Windows 的难度，甚至可以说大大降低了学习计算机的难度。

从程序员的观点来看，一致的用户界面是由于使用 Windows 的内部例程构造菜单和对话框而形成的。所有菜单均具有同样的键盘和鼠标界面，因为是由 Windows 处理这项工作，而不是由应用程序处理的。

(2) 多窗口、多任务操作

Windows 环境的最大特色就在于提供了多个窗口。一个应用程序占用一个窗口，用户可以随时在屏幕上移动任何一个窗口，改变一个窗口的大小，在不同的窗口间切换，窗口间还可相互传递信息。一个操作环境中允许多个应用程序同时运行，就表明该环境具备多任务操作的能力。

计算机主机中只有一个 CPU，实际上它一次还是只能做一件事，只是它从一件工作转换到另一件工作的速度非常快，所以感觉上像是在同一时间运行。Windows 会将 CPU 分配

给要同时运行的各个应用程序使用，这样，运行的程序越多，分工就越明显。

尽管有些人不断询问在单用户计算机上执行多任务是否是必要的，但用户确实乐于使用多任务，而且从中受益不少。MS-DOS 的 RAM 驻留程序 (如 Sidekick) 的流行证明了这一点。尽管严格地讲，弹出式程序并不是多任务程序，但它们允许快速的环境切换，这涉及到许多与多任务相同的概念。

在 Windows 下，每个程序实际上都是一个驻留的弹出式程序。用户可以同时显示并运行几个 Windows 程序，每个程序在屏幕上占据一个矩形窗口，如图 1-2 所示。用户可以在屏幕上移动、在不同程序之间切换，以及在程序之间传输数据。由于这种显示看起来很像桌面 (当然，是计算机还未主宰办公桌之前的桌面)，所以有人用"桌面"来比喻显示多个程序的 Windows。

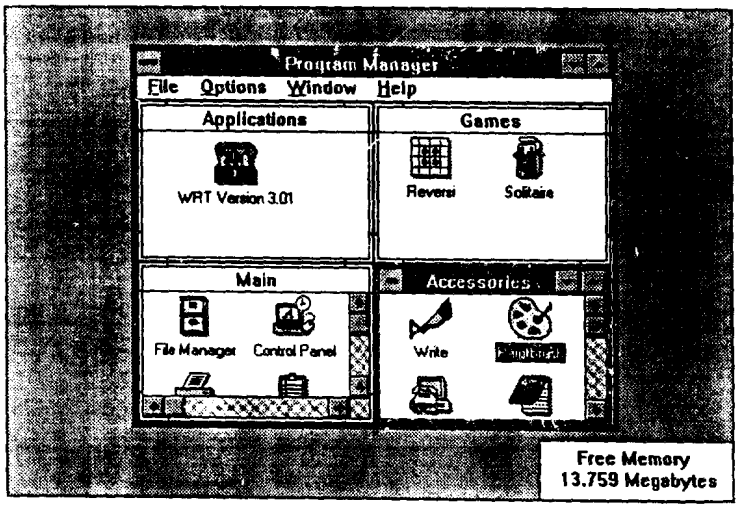


图 1-2　在 Windows 中同时运行多个应用程序

(3) 内存管理

如果不在内存管理方面作一些努力，操作系统是无法实现多任务的。在新程序启动、旧程序终止时，内存会变得零乱。系统必须能将空闲的内存空间组织在一起。这就要求系统移动内存中的数据和代码块。

在 8088 微处理器上运行的 Windows 1.0 就已能实现这种类型的内存管理了。在实模式下，这只能被看成是软件工程的一桩令人惊喜的成就罢了。在 Windows 下运行的程序可以超额分配内存；在任意时间，程序可以包含内存中装不下的代码量。程序可以放弃内存中的代码，然后从程序的.EXE 文件中重新装入之。用户可以运行一个程序的多个副本 (instance，又称"实例")；所有这些实例共享内存中同样的代码。在 Windows 中运行的程序可以共享位于其它.EXE 文件中的例程，这叫作"动态链接库"。Windows 包含一种技术，能在运行时将程序与动态链接库中的例程相链接。Windows 本身是一个动态链接库的集合。

因此，即使在 Windows 1 中，PC 体系结构的 640KB 内存限制实际上已被突破了，而且不要求任何附加内存。但 Microsoft 并未就此止步：Windows 2.0 使得 Windows 应用程序能访问扩展内存 (EMS)；Windows 3.0 能在保护模式下运行，使 Windows 应用程序能访

问高达 16MB 的扩充内存。

(4) 与设备无关的图形界面

Windows 是一个图形界面，Windows 程序能完全利用视频显示器和打印机上的图形和格式化文本。图形界面不仅在外观上吸引人，而且还能给用户传递比较高级的信息，如图 1-3 所示。

为 Windows 编写的程序不直接访问屏幕和打印机等图形显示设备硬件。相反，Windows 提供了一种图形程序设计语言 (称为图形设备接口，即 GDI)，它使得显示图形和格式化的文本很容易。Windows 虚构了显示硬件。为 Windows 编写的程序可以在任意视频卡和任意打印机上运行，只要 Windows 有它的设备驱动程序。程序不需要确定系统连接的是哪种设备。

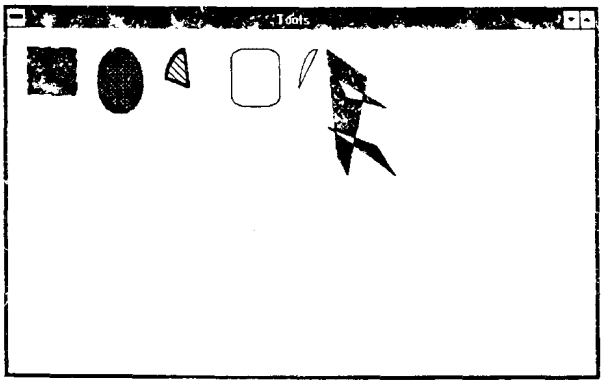


图 1-3 用刷子和画笔画出的简单图形

对于 Windows 的开发人员来讲，将一个与设备无关的图形界面输出到 IBM PC 上不是一件简单的事情。PC 的设计是基于开放体系结构的原则，第三方的硬件制造商被鼓励开发 PC 机的外设，并研制出大量外设。尽管已经出现了一些标准，PC 上的传统 MS-DOS 程序仍然必须分别支持许多不同的硬件配置。例如，对于一个作为商品发售的 MS-DOS 字处理程序而言，带有一两个较小的磁盘文件，其中每个文件支持一种特定的打印机，这是非常常见的。

Windows 程序不需要这些驱动程序，因为对这些设备的支持是 Windows 的本职工作。这使用户获益非浅，因为大多数 Windows 程序在安装方式方面没有什么要求。在此情况下，程序中所需要的一切都可以包含在程序的单个.EXE 文件中。用户能经常将.EXE 文件拷贝到硬盘上，装入 Windows，然后运行。

(5) Windows 与 MS-DOS

尽管 Windows 原本用来运行专为 Windows 环境设计的程序，但它也能运行非 Windows 的 MS-DOS 程序。

MS-DOS 程序可以分为两大类，其中较好的应用程序是那些使用 MS-DOS 和 PC ROM BIOS (基本输入／输出系统) 软中断来读键盘和视频显示器的程序；而较差的应用程序是那些直接写视频显示器、使用图形或控制硬件键盘中断的程序。所谓"较差"并非指程序的质量 (许多为 PC 编写的最好的应用程序，在 Windows 中却是较差的程序)，而是指程序使用 PC 硬件的方式。在基于 286 的机器上运行时，Windows 完全没有办法允许此类程序窗口化或多任务化。不过，Windows 可以用 386 微处理器的虚 86 模式来"窗口化"或"多任务化"较差的应用程序。

Windows 的启动与在 MS-DOS 下运行的其它普通应用程序相同。但是，一旦 Windows 装入内存，它就成为一个功能完备的操作系统。当然，还不能完全说 Windows 是操作系统，因为它是在 MS-DOS 之上运行的。在 Windows 运行时，它与 MS-DOS 共同负

责管理计算机的硬件资源。基本上由 MS-DOS 管理文件系统，而 Windows 做除此之外的一切事情。Windows 管理视频显示器、键盘、鼠标、打印机和串行端口，并负责内存管理、程序执行和调度。

Windows 与 MS-DOS 各有所长，互为补充。Windows 几乎不提供对文件 I/O 的支持，但这是即使最小的操作系统 (如 MS-DOS) 也具备的最基本的功能之一。这就产生了一些令人哭笑不得的结果。在 Windows 中，创建一个基于磁盘的、包含了一系列复杂图形绘制命令的元文件比创建一个简单的 ASCII 文件更加容易，因为前者是 Windows 完成的，而后者是 Windows 用 MS-DOS 来完成的。

编写 Windows 程序要么是完备的，要么就完全不予编写，无法折衷。例如，不能编写一个 MS-DOS 应用程序 (即使是所谓"较好"的程序)，而在其中只用 Windows 处理图形。如果想使用 Windows 的部分功能，就必须全身心地投入，编写功能完备的 Windows 程序。

在知道了 Windows 程序的结构之后，原因就变得非常明显了。Windows 中的一切都是相互联系的。如果想在视频显示器上绘制一些图形，就需要一个"设备环境句柄"；要获取设备环境句柄，就需要一个"窗口句柄"；要得到窗口句柄，必须创建一个窗口，并准备接收发送给窗口的"消息"；要接收和处理消息，又需要一个"窗口过程"。这时候，就已经在编写 Windows 程序了。

1.4 重要的 Windows 程序设计概念

人们都知道，Windows 对用户而言较方便，但对程序员却很复杂。如果读者以前没有设计图形用户界面的经验，那么以后肯定会碰到一些陌生的概念。几乎每个刚开始编写 Widnows 程序的程序员都经历了思维转向，以消化和吸收这些概念。

如果刚开始时发现编写 Windows 程序困难、枯燥，而且还充满陌生的概念，这是很正常的反应。许多优秀的 Windows 程序员一开始也是如此。只要掌握了这些概念和一般的编程方法，以后编程时就不会感到太棘手了。

(1) 窗口

从用户的视觉观点来看，"窗口" (window) 就是屏幕上的一个矩形区域，也就是程序的工作空间。Windows 的应用程序通常拥有一个窗口 (或再加上一些附属窗口)，应用程序就在这个窗口中输入、输出和完成各种操作。

对程序员而言，窗口是一个接收和处理信息的对象，也是一个虚拟屏幕。以往 MS-DOS 下的应用程序可以占据整个实际屏幕，但 Windows 应用程序则不行，所有操作都放在一个窗口中，用户的键盘及鼠标输入都送到该窗口中处理，输出文本或绘图都必须限制在该窗口中。

不论输入、输出操作，还是其它可能影响窗口的事件，都以"消息" (message) 的形式通知该窗口。程序设计员的责任便是为每个窗口编写一个专用的函数，处理各种"消息"。程序设计员通过对消息的处理，达到操作和控制窗口的目的。

究竟这些消息是从什么地方来的？是谁传送它们过来的？大部分消息都来自 Windows，但窗口也可以发送消息给自己，或发送消息给别的窗口。例如用户在窗口中按

下鼠标按钮，Windows 就必须将此事件以一个消息告诉其专用函数，由该函数决定针对此消息应做哪些操作。Windows 系统中的消息种类非常繁多 (约 220 种)，程序员可对每种消息作必要的处理。

(2) 窗口函数

"窗口函数"也称为窗口过程 (window function 或 window procedure)。每个窗口都有一个专用函数，负责处理该窗口的消息，这个专用函数就叫作"窗口函数"。窗口函数的作用在于对 Windows 所传来的消息作适当的响应。程序员的主要工作就是设计窗口函数。

(3) Windows 函数调用

当前，Windows 系统中可供应用程序调用的函数已超过 550 个，所有函数名称均根据其代表的含义而定，且大小写字母混合书写，这样不但易懂，且便于查询和使用。例如 CreateWindow 函数显然就是"create a window"的意思。

虽然 Windows 对用户而言是一个易学易用的系统，但对程序员来说却非常困难。试想，一个含有 550 个函数的软件开发工具会容易吗？编写 Windows 应用程序势必要用到这 550 个函数，因此大多数 Windows 程序员在每次编写 Windows 程序时都备有《Windows 程序员参考手册》，以便随时查阅。

Windows 函数调用和窗口函数并不相同，这一点用户必须十分清楚。前者是指由 Windows 所提供的库函数，它们有固定的名称，由应用程序来调用；后者是某个窗口的专用函数，由程序员负责编写，并自定义名称，而且它是供 Windows 调用的函数。

(4) PASCAL 调用方式

Windows 函数的调用类型常声明成 FAR PASCAL，FAR 指远程调用 (far call)。由于 Windows 函数和调用该函数的程序代码可以位于不同的程序段 (code segment) 内，因此需用 32 位地址来传送函数所在的"段地址" (16 位的 segment 地址) 及"偏移地址" (16 位的 offset 地址)，所以必须声明成 FAR 类型。

PASCAL 类型则是 Windows 惯用的调用方式，它已被公认为标准。这种类型的函数在编译器产生机器码时会依据参数在函数中出现的顺序，将各参数放入堆栈中，以供函数使用。参数在堆栈中的顺序恰好与普通的 C 函数相反。

(5) Windows 包含文件

Windows 中 550 个函数的原型 (prototype) 均声明在 WINDOWS.H 文件中，这个文件中还包括标识符以及数据结构的定义。编写每个 Windows 应用程序都必须在开头将它包含进去:

```
#include  <WINDOWS.H>
```

WINDOWS.H 文件中包括很多东西，它是一个很大的包含文件 (又称头文件)。编译 Windows 程序比编译一般 C 程序要花更长的时间，就是因为有这个包含文件的缘故。

此文件不但大而且相当重要，它包含 1200 个以上的#include 指令，它们大都用来定义标识符常数，例如:

```
#define  WM_DESTROY  0x0002
```

几乎每个 Windows 使用的常数都在 WINDOWS.H 中有一个对应的标识符。程序员编写程序时，不可能记得住所有常数值，因此必须使用这些定义好的标识符 (标识符是有意义的字符串，便于记忆)。以下列举一些标识符常数:

CS_HREDRAW　　IDI_APPLICATION
WS_OVERLAPPED　　WM_QUIT
DT_SINGLELING　　WS_HSCROLL
IDC_ARROW　　CS_VREDRAW
WM_DESTROY　　DT_CENTER
CW_USEDEFAULT

前两个字母为某一类东西的缩写，加一下划线，再加上一些有含义的字母拼凑起来，就成为 Windows 标识符。缩写字母的含义如表 1-1 所示。

表 1-1　Windows 中的缩写字母

缩写字母	英 文 原 意	含　　义
CS	class style	窗口类的类型
IDI	ID for an icon	图标标识符的 ID 号
IDC	ID for a cursor	光标标识符的 ID 号
WS	window style	窗口类型 (也称窗口风格)
WM	window message	窗口消息
CW	create window	建立窗口 (也称创建窗口)
DT	draw text	输出文本 (在图形环境下也称写文本)

WINDOWS.H 还包括 100 个以上的 typedef 指令，它们是关于数据类型和数据结构的声明，例如:

```
typedef  int  BOOL
```

声明 BOOL 这个数据类型是 int (整数) 类型。从 BOOL 字面上看，这种类型的变量用于“布尔值” (boolean value，真或假)。另外还有一些其它的 typedef 声明，见表 1-2。

表 1-2　Windows 中的类型标识符

WINDOWS.H 类型	英 文 原 意	含　　义
BYTE	unsigned char	无符号字符
WORD	unsigned integer (16 bits)	无符号整数
LONG	signed long integer (32 bits)	符号长整数
DWORD	unsigned long integer (32 bits)	无符号长整数
LPSTR	far pointer to a character string	远程字符串指针
FARPROC	far pointer to a procedure	远程过程指针

这里不一一列举。typedef 也能用来定义结构，WINDOWS.H 定义了很多数据结构，例如:

```
typedef struct tagMsg
{
    HWND    hWnd;
    WORD    message;
    WORD    wParam;
    LONG    lParam;
    DWORD   time;
```