高级WINDOWS程序设计技术

王 浩 李 莉 任宇新等 编著

潘金贵 徐殿祥 审校

同济大学出版社

高级 Windows 程序设计技术

王 浩 李 莉 任宇新 等编著

潘金贵 徐殿祥 审校

同济大学出版社

内 容 简 介

本书是一本进行 Windows 程序设计的高级参考读物。全书共分为二十五章，包括以下内容：Windows 编程概述、Windows 编程要素简介、基本输出处理、打印、TrueType 字模、键盘和鼠标输入、定时器、内存管理、资源、滚动条、Windows 中的控件、通用对话框、GDI、绘图、位图、位块传输和元文件、文件管理、帮助系统、多文档界面、剪贴板、动态链接库、OLE、系统登录数据库、QuickCase：W 以及 DOS 保护模式接口等。

本书可供计算机专业人员、广大 Windows 用户、各大专院校师生及有关科研人员参考。

责任编辑　王建中
封面设计　李志云

高级 Windows 程序设计技术
王浩　李莉　任宇新　等编著
潘金贵　徐殿祥　审校
同济大学出版社出版
（上海四平路 1239 号）
新华书店上海发行所发行
同济大学印刷厂印刷
开本：787×1092　1/16　印张：50　字数：1280 千字
1996 年 1 月第 1 版　1996 年 1 月第 1 次印刷
印数：1—5000　定价：52.00 元
ISBN 7－5608－1608－8/TP·173

前　言

目前，Windows 已经售出 1000 万份拷贝，它是目前市场上使用最为广泛的图形用户界面，其市场占有率远远超出了 Macintosh。Windows 的普及主要得益于它使用方便。用户初学 Windows 时，很快就能掌握其使用方法。Windows 操作系统的推出，使得操作计算机（特别是 PC 机）的方法和软件开发方法发生了革命性的变化。而继 Windows 3.0 之后推出的 Windows 3.1，则使这种革命性的变化更为深刻，走上了一个前所未有的新台阶。

Windows 是一个图形窗口操作系统，它是个人机操作系统发展史上的一个里程碑。与 DOS 相比，其功能特点主要表现如下：

- 全新的图形界面。Windows 所定义的图形界面组件目前已成为 PC 机界面的事实上的标准。自 Windows 问世之后，各软件系统纷纷向 Windows 标准靠拢，以期借 Windows 的强大功能，提高自身的性能和价值。
- Windows 3.1 突破了 DOS 内存 640K 的限制，提供了标准模式和 386 增强型模式这两种模式，提供了虚存管理能力。
- 在 Windows 操作系统中可运行 DOS 应用程序。
- 提供了对多任务的并行处理功能，各任务间可方便地进行切换，又可方便地交换信息。
- Windows3.1 中配有多媒体管理器，不仅顺应了多媒体这一潮流，而且较为实用，可望在后续版本中更能有所突破。
- Windows 系统提供了与网络的接口。
- 提供了一系列管理工具，如程序管理器、文件管理器、打印管理器、控制面板等。
- 提供了功能强大的应用程序，如字处理器、画笔软件、终端仿真通信软件等。
- 提供了桌面办公用具，如时钟、卡片、日历、计算器、便笺、记录器、字符图、收录机、多媒体管理器等。
- 为非 Windows 应用程序提供了 PIF 编辑器，为 Windows 应用程序提供了软件开发工具包 SDK。

与 Windows 3.0 相比，Windows 3.1 的新特点主要表现如下：

- 提供了字符图、收录机、多媒体管理器等具有时代特点的桌面办公用具，改进了办公的软环境。
- Windows 3.1 删弃了前期版本中的实模式，将其融入了标准模式及 386 增强型模式之中。
- 带有新版本的 SMARTDRIVE，HIMEM，RAMDRIVE 等辅助工具。
- 在用户界面、操作方法上更为自然、方便，提出了“drag and drop”的新思想。

Windows 3.1 推出后，Windows 才真正像一个操作系统。当系统加电后，进入 DOS 且把计算机系统资源交给 Windows 环境，并受 Windows 管理。在安装了新的 FastDisk 后，Windows 也能管理文件操作，从而使 DOS 只能控制 DOS 应用程序了。

如果系统重新启动，计算机的资源又将由 Windows 管理。Windows 为其环境下运行的应用程序提供多任务、系统管理和资源控制。

Windows 功能强大，为充分发挥其优越性，用户应当把自己的应用程序转向支持 Windows。现在已有很多计算机公司要求为其 PC 机所购买的应用程序必须是可在 Windows 下运行的程序。

本书可供 Windows 用户使用，也可供已熟悉 Windows 的高级程序设计人员使用。对于 DOS/Windows 来说，用 Windows 替代 DOS 的时代已经到来。

参加本书编写的作者有：王浩、李莉、潘金贵、任宇新、韩小琴、刘应木、李阳、何维、王立国、孙朝辉、葛佳、刘秀英、李华、陈越、李燕、朱尽染、姜鸿燕等。潘金贵和王浩同志对全书进行了仔细的修改和统编，并蒙徐殿祥博士审校，值此谨表谢意。

由于水平、时间所限，不妥之处在所难免，欢迎广大读者批评指正。

编著者

1995 年 6 月于南京

目　录

第一章　Windows 编程概述

本章简要描述用户在编写 Windows 应用程序的过程中所必需的知识，包括用户应该备有什么软件以及如何建立一个 Windows 应用程序。

1.1　Windows 综述

Windows 为用户提供了一种多任务、基于图形的窗口环境，在此环境下可运行专门为 Windows 设计的程序。这些程序包括 Microsoft Excel（电子表格和商业图形软件），Microsoft Word（字处理软件），PageMaker（桌面排版系统），Ami（字处理系统）、Designer（绘图软件），Current（个人信息管理软件），TooBook（软件构造工具包）等等。

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，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器（用于运行程序）、一个文件管理器（用于维护文件）和一个打印管理器（用于管理打印机队列）。

尽管 Windows 原本用来运行专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 之下。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。

Windows 为程序开发人员提供了丰富的内部例程，使程序员可以使用菜单、对话框、滚动条等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模进行格式化。程序员可以用一种与设备无关的方式来处理键盘、鼠标、视频显示器、打印机等。

1.2　对用户的要求

阅读本书时，让我们从用户所需的编程经验开始。虽然在无以下知识的情况下用户也能阅读本书，但是我们建议读者尽可能地作进一步的学习，以免误解我们的意图：

- 对于用 C 语言编写程序已有了一些经验。如果用户还未以 C 语言编制过程序的话，那么在这方面有许多参考书可供使用。
- 有基于 PC MS-DOS 方面的经验。
- 写过在 MS-DOS 下运行的应用程序。
- 使用过 Microsoft Windows。

1.2.1 工具

要最有效地运行 Windows 和众多的开发工具，需要以下硬件支持：

- 基于 Intel 80386 微处理器的 IBM 个人机(或兼容机)，配有硬盘、4MB 以上内存、使用 MS-DOS 3.30 或更高版本。
- 图形显示器和视频卡，最好与 IBM VGA(视频图形阵列)或更好的显示器兼容。
- 鼠标。尽管对大多数 Windows 程序来说，鼠标是可选的，但本书中的某些程序需要鼠标支持。

用户将要用到下列软件编写 Windows 应用程序：

- 一个 C 编译器。如果用户是一个有经验的 C 程序员，那么就可能已经对一个编译器有所了解。本书中的源码均采用 Microsoft C 编译器 7.0 版本和 Pressional Development System(PDS)支持的 C 语言。

  用户也可选择使用 Microsoft Quick C for Windows 或 Quick Case:W 开发软件。
- Microsoft Windows Software Development Kit(SDK) 3.1。这个版本包含编写 Windows 应用程序的许多工具。

书中的例子程序采用 Microsoft C 7.0 和 Microsoft Software Development Kit(SDK) 3.1 版本。由 Windows 3.1 来运行已编译的程序。本书还介绍了如何用 Microsoft QuickCase:W 来开发源代码。

C 编译器包括：

- 一个编译器和链接器(CL.EXE 和 LINK.EXE)。这两个程序用于编译源码并且将此源码转变成可执行的程序代码。
- 头文件 DOS.H 和 CONIO.H。头文件包含定义、宏指令和其他重要的编程数据。
- 用于处理不同内存模式的标准库文件：SLIBCE.LIB，MLIBCE.LIB 和 LLIBCE.LIB(代表小、中和大)。

Software Development Kit(SDK)可将一个程序转变成 Windows 应用程序，此 SDK 的功能包括：

- 一个链接器(LINK.EXE 或 LINK4.EXE)和资源编译器(RC.EXE)。
- CodeView，为查找应用程序中的错误而调试应用程序(CVW.EXE)。
- Windows 头文件和库文件。
- 一个字模编辑器(FONTEDIT.EXE)、一个对话框编辑器(DLGEDIT.EXE)和一个图像编辑器应用程序(IMAGEDIT.EXE)。IMAGEDIT 类似于 Microsoft Windows Paintbrush，但是 IMAGEDIT 允许用户在 Windows 应用程序中产生光标、位图和图标等。

1.2.2 安装

我们建议用户先安装 Microsoft C 编译器，再安装 SDK。举例来说，我们将 Microsoft C 安装在 D:\C7.0 目录中，将 SDK 安装在 D:\WINDEV 目录中，再安装如下内容到 AUTOEXEC.BAT 文件中：

```
PATH = C:\DOS;C:\WINDOWS;D:\C700\BIN;D:\WINDEV;D:\WINDEV\INCLUED;D:\
```

```
        WINDEV\INCLUED\SYS
SET LIB=D:\WINDEV\LIB;D:\C700\LIB
SET INCLUED=D:\WINDEV\INCLUED;D:\C700\INCLUED
SET HELPFILES=D:\C700\HELP\*.HLP
SET INT=D:\C700\INI
```

1.3 Windows 与 DOS 应用程序

让我们看看 Windows 应用程序怎样区别于以标准 DOS 为基础的应用程序。我们从开发一个以 DOS 为基础的应用程序的源代码开始(本例使用 C 语言)。

一旦开发人员写下了源代码并且确信代码无错的话,代码就可以通过一个编译程序(CL. EXE)来运行。源码(我们在本例中将称作 HELLO. C)在编译程序中引用头文件(以 .H 为扩展名的文件)。

这些头文件(有时被称作 include 文件,因为 C 语言的 # include 语句在源代码内请求这些文件)包含完整程序所需的信息(例如用于屏幕显示)。

编译器产生一个目标文件(此处命名为 HELLO. OBJ)。

还剩一步:.OBJ 文件和一个库文件(? LIBCEW. LIB)必须由 LINK. EXE 链接以形成一个可执行文件。最后的结果文件是 HELLO. EXE。

图 1.1 说明了一个 DOS 应用程序是怎样编译、并从 C 源码转变为一个可执行文件的:

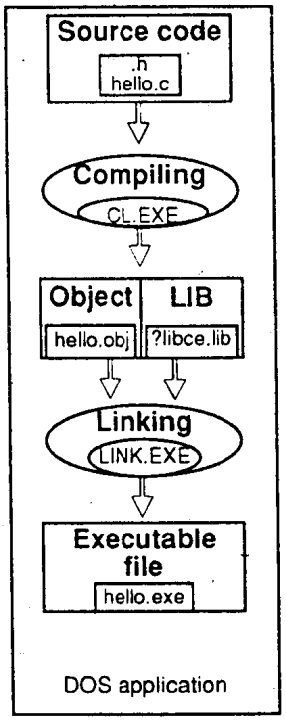

图 1.1 DOS 应用程序怎样从 C 源码转变成可执行文件

生成 Windows 程序时要作同样的处理(编译和链接),除了以下几点不同以外:

(1)一个以 WINDOWS. H 命名的头文件、其他包含文件及源代码被一起引用。WINDOWS. H 包含 Windows 所专用的类型声明、宏和其他信息。

(2)与? LIBCE. LIB 文件由 C 使用不同,Windows 专用程序库由链接器使用(? LIBCEW. LIB 和 LIBW. LIB)。

(3)一个模块定义文件(由 . DEF 扩展名说明)与目标文件程序库链接。模块定义文件定义名称、描述、数据类型和所需的其他资源。

链接好之后还不能在 Windows 下运行,还需要资源,比如位图、字符串、图标和光标等。

资源描述文件(带 . RC 扩展名)使用资源编译器 RC. EXE 来编译。结果文件由 RC. EXE 加到链接好的 . EXE 文件中。

最后结果就是一个与 Windows 3. 1 兼容的应用程序。

如果用户未编译资源,略为简单的应用程序可能会在 Windows 3. 1 下运行,但可能产生错误。

图 1. 2 说明 Windows 应用程序如何从源代码转变为可执行文件。

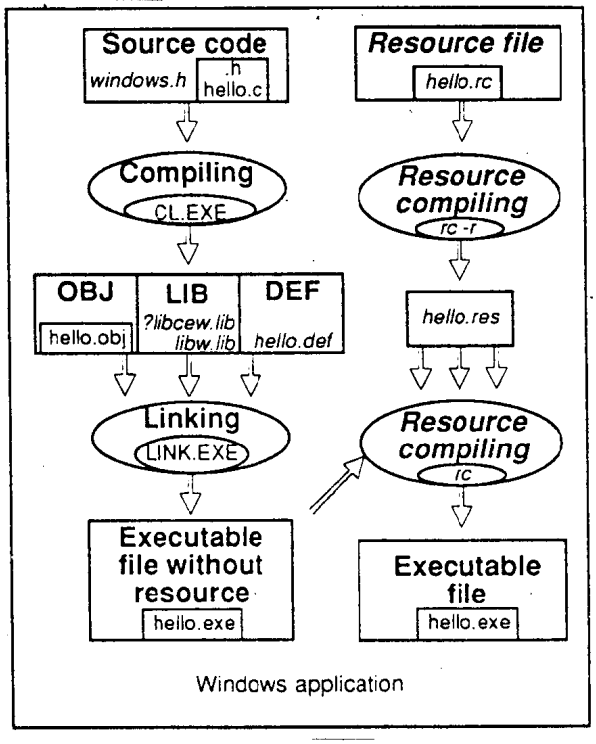

图 1. 2 Windows 应用程序如何从源代码转变成可执行文件

1. 4 Windows 3. 1 略览

Windows 3. 1 是在 1992 年春季推出的。Windows 3. 1 图形用户接口是一个适于多任务操作的 DOS 用户环境。

1.4.1 Windows 模式

Windows 3.1 有两种不同的模式：标准模式和 386 增强型模式。更早的 Windows 版本有第三种模式，叫做实模式，但 Windows 3.1 不支持这种模式(以后详述)。这两种模式定义如下。

1. 标准模式

标准模式可突破 MS-DOS 的内存限制。在常规内存之上加上一定量的扩展内存，总称为邻接内存。另外，高位内存区域(HMA)的一块内存也加在扩展和常规内存之上。

MS-DOS 应用程序必须在常规内存中运行，正如实模式下运行的 Windows 一样。

2. 386 增强型模式

386 增强型模式只适用于 80386 或 80486 处理器。在此方式下，为每一单独的 DOS 程序创立虚拟 DOS 设备。虚拟存储器用于所有的 Windows 应用程序，也就是说也可使用“页面内存”。

Windows Virtual Memory Manager(VMM)负责对最近未使用的页面和硬盘进行存储和交换。4K 页中的每一页可存在一个临时的或永久性的交换文件中。Windows 给临时文件取名为 WIN386.SWP，永久性文件命名为 386SPART.PAR，并且具有隐含属性和系统属性。

1.4.2 动态链接

如果读者曾进行过 MS-DOS 编程，也许会猜测，Windows 程序是不是通过一个软件中断与 Windows 联系的呢，如 MS-DOS 中断 0x21？有人会猜测，链接程序向 Windows 程序中加入联编，将 Windows 函数调用转换成软件中断。但是这种观点是不对的。Windows 程序与 Windows 的接口是一个“动态链接”过程。

与 MS-DOS 程序一样，Windows 可执行程序具有文件扩展名 .EXE。但是，它使用所谓“新的可执行文件格式”(New Executable file Format)作为 .EXE 格式。每当 Windows 程序调用 Windows 函数时，C 编译程序就产生远程调用的汇编语言代码。.EXE 文件中存在一个表，它用一个动态链接库名和此库中函数的名字或编号(称为序号)来标识被调用的函数。

Windows 本身包含三大类动态链接库。它们是 KRNL286 和 KRNL386(负责内存管理、加载、执行及调度程序)，USER(用户界面和窗口)和 GDI(图形)。这些库包含了大多数 Windows 函数的代码和数据。这三个动态链接库在 Windows 目录的 SYSTEM 子目录中。

在 Windows 程序装入内存后，程序中的远程调用被转换为指向动态链接库中函数的入口，动态链接库也被装入内存。这就是为什么要将所有 Windows 函数定义为 far 的原因：动态链接库中的代码与程序代码不在同一段中。另外，传给 Windows 函数的指针也必须定义为 far，避免与动态链接库自身的代码和数据相混淆。

一般来说，程序员不用担心远程调用和远程指针的使用，因为函数已在 WINDOWS.H 中用远程指针声明为远程函数，C 编译程序会自动进行必要的地址转换。

在链接 Windows 程序以产生可执行文件时，必须使用 Windows 软件开发工具包(SDK)或 Borland C++编译程序中提供的专门“输入库”。这个输入库包含动态链接库名和所有

Windows 函数的序数号。链接程序使用这个信息，在 .EXE 文件中构造一个表，在载入程序时 Windows 用这个表将调用转换为 Windows 函数。

1.4.3 面向对象的程序设计

进行 Windows 程序设计，实际上就是进行一种面向对象的程序设计。这一点在“窗口”中显得最为明显。窗口是我们在 Windows 中使用得最多的对象，该对象也正是 Windows 之所以命名为“Windows”的原因。

窗口是屏幕上的矩形区域。它从键盘或鼠标接受输入，并在其表面显示图形输出。

一个应用程序窗口通常包含程序的标题条、菜单、边框，也许还有一些滚动条。对话框也是一种窗口，不同的是，对话框表面通常包含几个其他窗口，称为“子窗口”。这些子窗口的形式有按钮、单选按钮、复选框、文本输入区、列表框和滚动条。

用户将这些窗口看成屏幕上的对象，可通过按下一个按钮或滚动一个滚动条与这些对象直接交流。有趣的是，程序员的观点与用户的观点极其类似。窗口以“消息”的形式接收窗口的输入，窗口也用消息与其他窗口通信。

对消息的理解将是读者成为 Windows 程序员所必须逾越的障碍之一。

1.4.4 消息驱动

我们可以演示一个在窗口中运行的最基本的字处理程序。在程序窗口的大小改变时，字处理器会重新格式化其中的文本。

很明显，窗口大小改变的细节是由操作系统处理的，程序能够响应这个系统函数。程序是怎样“知道”窗口的大小发生改变的呢？操作系统使用了什么技术将这一信息传递给窗口？在苦思冥想这些问题时，我们的传统编程经验似乎毫无用处。

问题的关键在于理解图形用户界面所使用的体系结构。在 Windows 中，当用户改变窗口的大小时，Windows 给程序发送一条消息指出新窗口的大小。然后，程序就可以调整窗口中的内容，以响应大小的变化。

“Windows 给程序发送消息。”我们希望读者不要对这句话视而不见。它到底表达了什么意思？我们在这里讨论的是程序代码，而不是一个电子邮件系统。操作系统怎么能给程序发送消息呢？

其实，所谓“Windows 给程序发送消息”，是指 Windows 调用程序中的一个函数。该函数的参数描述了这个特定消息。这种位于 Windows 程序中的函数称为“窗口过程”。

1.4.5 窗口过程

无疑，读者对程序调用操作系统的思路是很熟悉的。例如，程序在打开磁盘文件时，就要使用有关的系统调用，读者所不习惯的，可能是操作系统调用程序。而这正是 Windows 面向对象体系结构的一大基石。

程序创建的每个窗口都有相关的窗口过程。这个窗口过程是一个函数，既可以在程序中，也可以在动态链接库中。Windows 通过调用窗口过程来给窗口发送消息。窗口过程根据此消息进行处理，然后将控制返回给 Windows。

更精确地说，窗口通常是在“窗口类”的基础上创建的。窗口类标识了处理窗口消息的窗

口过程。使用窗口类使得多个窗口能基于同一个窗口类，且使用同一个窗口过程。例如，所有 Windows 程序中的所有按钮均基于同一窗口类。这个窗口类与一个处理所有按钮消息的窗口过程(位于 Windows 的 USER.EXE 动态链接库中)联结。

面向对象程序设计中，对象是代码与数据的组合。窗口是一种对象，其代码是窗口过程，数据是窗口过程保存的信息和 Windows 为每个窗口和系统中每个窗口类保存的信息。

窗口过程处理发送给窗口的消息。这些消息经常告知窗口，用户正用键盘或鼠标进行输入。这正是压入按钮窗口知道它被“按下”的奥妙所在。在窗口大小改变，或者窗口表面需要重画时，由其他消息通告窗口。

Windows 程序开始执行后，Windows 为该程序创建一个“消息队列”。这个消息队列用来存放该程序可能创建的各种不同窗口的消息。程序中有一小段代码，叫做“消息循环”，它用来从队列中取出消息，并将它们发送给相应的窗口过程。有些消息直接送给窗口过程，不用放入消息队列中。

若读者注意这段对 Windows 体系结构的简略描述，也许对理解在真实的程序环境中窗口、窗口类、窗口过程、消息队列、消息循环和窗口消息是如何相互配合的会有所帮助。

1.5 Windows 应用程序的组成

许多 Windows 应用程序与本书所列出的所有应用程序一样采用 C 语言。我们曾提到过，为产生一个可执行的应用程序，除源文件以外，还需要其他文件。图 1.3 列出这些增加的文件。在本书中，我们试图采用最通用的文件。

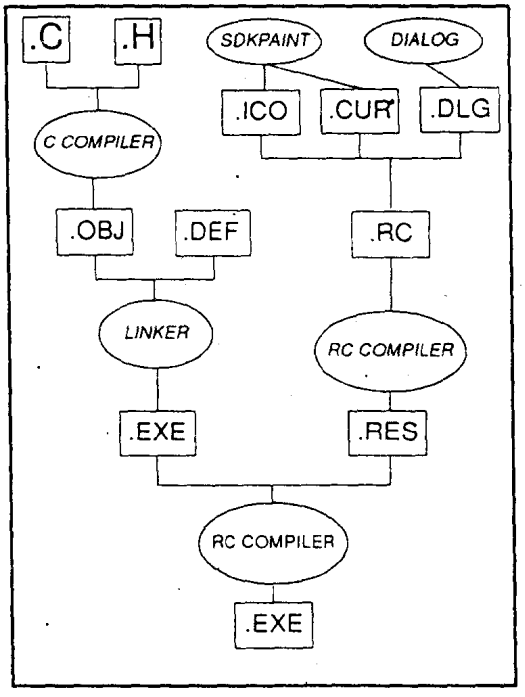

图 1.3 需加到源文件中的增补文件