

计算机专业人员书库

罗云彬

编著

Windows 环境下 32 位

汇编语言程序设计



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>



计算机专业人员书库

Windows 环境下 32 位 汇编语言程序设计

罗云彬 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

Windows 环境下 32 位汇编语言是一种全新的编程语言。它使用与 C++ 语言相同的 API 接口,不仅可以用来开发出大型的软件,而且是了解操作系统运行细节的最佳方式。

本书从编写应用程序的角度,从“Hello World!”这个简单的例子开始到编写多线程、注册表和网络通信等复杂的程序,通过 60 多个实例逐渐深入 Win32 汇编语言的方方面面。

本书作者罗云彬拥有十余年汇编语言编程经验,是汇编编程网站 <http://asm.yeah.net> 和汇编编程论坛 <http://win32asm.yeah.net> 的站长。本书是作者多年来编程工作的总结,适合于欲通过 Win32 汇编语言编写 Windows 程序的读者。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

Windows 环境下 32 位汇编语言程序设计/罗云彬编著. —北京:电子工业出版社,2002. 10

(计算机专业人员书库)

ISBN 7-5053-8034-6

I. W... II. 罗... III. 汇编语言—程序设计 IV. TP313

中国版本图书馆 CIP 数据核字(2002)第 074991 号

责任编辑:郭立 特约编辑:蔡祖瑛 李建森

印刷:北京民族印刷厂

出版发行:电子工业出版社 <http://www.phei.com.cn>

北京市海淀区万寿路 173 信箱 邮编 100036

经销:各地新华书店

开本:787×1092 1/16 印张:47.25 字数:1210 千字 附光盘 1 张

版次:2002 年 10 月第 1 版 2003 年 1 月第 2 次印刷

印数:5 000 册 定价:78.00 元(含光盘)

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系。
联系电话:(010)68279077

致 谢

首先感谢我的父母亲，如果没有你们从小到大对我的培养，就没有这一切。也感谢我的妹妹，在很多关键的时候，你总是给予我很多的帮助。

感谢我的未婚妻小猪猪，要不是你将逛街、买衣服、旅游的时间慷慨地贡献出来，并不时地为我泡杯咖啡，给我捶捶背并且给我鼓励，我不可能完成这样一部作品。

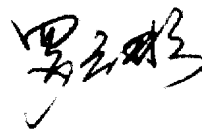
感谢我的母校浙江大学。浙大“求是创新”的校训，“实事求是、严谨踏实、奋发进取、开拓创新”的校风让我能够有一个好的学习习惯，让我在毕业以后的这么多年里能够始终有一种动力去学习最新的知识。

感谢电子工业出版社的郭立编辑，您的支持、鼓励和专业的指点使本书能够按照进度按时完成。

本书的写作始于众所周知的“台州电信黑客案”发生以后，这件事情也让我看到了很多真正的朋友，特别是台州市新时代科技发展有限公司、台州颐高数码广场的董事长胡朝辉先生，感谢你们能始终和我站在一起。如果没有你们的鼓励和帮助，我可能没有机会来完成这本书，你们是我真正的朋友！

感谢网上众多汇编编程论坛中的网友，感谢你们对我的关心和爱护，也感谢你们为我提了很多宝贵的意见和建议。

也感谢所有关心我和爱护我的人，虽然我可能认识你们，也可能不认识你们，但是我永远感觉得到你们对我的情谊。



2002年5月29日

前 言

从 Windows 出现开始, 汇编语言似乎在慢慢地销声匿迹, 但本书可以让人放弃这个观点, 其实在 Win32 环境下, 汇编语言依然强大。

Why——为什么选择 Win32 汇编

选择 Win32 汇编的理由是什么呢?

在 DOS 时代, 学习汇编就是学习系统底层编程的代名词, 仅要成为一名入门级的汇编程序员, 就需要学习从 CPU 结构、CPU 工作方式、各种硬件的编程方法到 DOS 工作方式等范围很广的知识。随着 Windows 时代的到来, Windows 像一堵巨大的墙, 把我们和计算机的硬件隔离开, 对于 DOS 的汇编程序员来说, 就像在一夜之间, 我们发现自己曾经学过的几乎所有的东西都被 Windows 封装到内核中去了, 由于保护模式的存在, 我们又无法像在 DOS 下一样闯入系统内核为所欲为。在 Windows 下用任何语言编程都必须遵循 Windows 的规范, 汇编也不例外, 也就是说, 汇编不再是一种“有特权”的语言。面对汹涌而来的 Visual C++, Visual BASIC, PowerBuilder 和 Java 等各个领域的猛将, 从 DOS 时代“为所欲为”的“系统警察”岗位下岗, 在别的领域又没有一技之长, 汇编语言似乎失去了生存的意义, 有很多人在 DOS 转向 Windows 的时候放弃了汇编语言。

但是经过短暂的失落, 摆正了自己在系统中的位置, 我们发现从“系统警察”转换到遵循 Windows 规范的“好市民”后, 汇编语言又慢慢地在这个世界流行起来了, 毕竟, 不能为所欲为也可以有好的一面, 我们可以不必再考虑一些老大难的问题, 如程序运行时会对什么样的显示卡, 如何驱动不同的打印机, 内存不够了如何用磁盘交换等等; 我们也可以在了解更少硬件知识的情况就可以掌握 Win32 的汇编编程; 而且, 我们惊喜地发现, 做了“好市民”以后, 我们反而拥有了和别的语言同样的权利——为了做图形和界面等方面的功能, 汇编程序员在 DOS 时代连做梦都在羡慕 C 语言庞大的函数库, 而现在, Windows 为我们提供了比这还要多得多的函数, 以至于其他大部分语言可以做出来的东西, 汇编都可以做, 而别的语言做不到的东西, 汇编照样可以做! 所以这里就是理由之一: **Win32 汇编可以当做一种功能强大的开发语言使用, 使用它完全可以开发出大型的软件来。**

正是因为 Win32 汇编看上去不再那样低级, 于是有读者曾经提出: Win32 汇编讲的都是用 API 来写程序, 和高级语言差不多, 以前在 DOS 下使用的中断什么的都不能用了, 所以没有什么新奇的了。言下之意就是: 学汇编就是为了了解高级语言底下一层的功能, 但现在 Win32 汇编却使用和 C++ 等语言相同的 API 接口, 既然和高级语言处于同一个级别, 我们为什么还要去和机器指令打交道呢, 还不如去学 Visual C++ 方便。

但是我们可以这样问一问自己:

问: 在 DOS 汇编中我们为什么用中断功能?

答: 为了使用 DOS 内核提供的功能。

问: 在 DOS 中我们常常自己用操作 I/O 端口的方法读写硬盘或操作显卡吗?

答: 不, 我们用系统提供的 Int 13h 和 Int 10h

.....

同样，在 Win32 汇编里使用 API 也是为了使用 Windows 内核提供的功能。只不过使用的方式不再是中断方式而已，不是 Win32 汇编语言“高级化”了，而是高级语言因为使用 Windows 的 API 接口而“低级化”了，其代价就是无法移植到其他系统，用 Visual C++写的程序是无法移植到别的操作系统平台上的，只有和平台无关的 ANSI C++等才能算是真正意义上的高级语言。

其实，任何汇编语言都是和操作系统密切相关的，不管是 DOS 汇编、Win32 汇编还是 Linux 汇编，都是基于特定的操作系统的，如果一定要绕过操作系统，那么就不会有 DOS 汇编和 Win32 汇编的区别了，但是这样的话我们不是在学汇编，而是在自己开发操作系统。高级语言在不同的操作系统上看起来都差不多，但作为一种低级语言，不同操作系统上的汇编就是不同的世界，所以，既然 Windows 和 DOS 是两个完全不同的操作系统，我们就必须抛弃 DOS 汇编中的大部分概念从头开始学习 Win32 汇编。这就是理由之二：**Win32 汇编是 Windows 环境下一种全新的编程语言。**

Win32 环境下的很多高级语言，如 Visual C++和 Visual BASIC 等，一如既往地实现的细节进行了或深或浅的封装，就连最能表现 Windows 特征的部分，如消息循环和多线程的处理等内容也都被隐藏封装，使我们在使用它们进行可视化编程的同时，却无法全面了解 Win32 程序运行的具体方式。令笔者印象深刻的一件事是：在初学 Windows 编程的时候笔者学的是 Visual FoxPro，用到窗口的时候，发现窗口总是在屏幕上一闪而过，百思不得其解，后来发现把窗口的属性设置为“模式”就可以了，但还是不知道其中的道理，后来才发现 Windows 本来就是使用两套不同的 API 来处理“模式”和“非模式”对话框的，但是在高级语言中两者封装在一起了。在学习 Win32 汇编以后，这些隐藏在高级语言后面的细节就暴露出来了。

由于封装的关系，各种高级语言或多或少都存在某种“缺陷”，比如 VB 不支持指针，结果很多需要使用指针的 API 用起来就很不方便，结果像多线程一类的特征在 VB 中就无法实现，PowerBuilder 也是如此；C 语言已经是最灵活的高级语言了，但还是无法在代码级别处理某些需求；而汇编语言见到的是一个最真实的操作系统，它可以用最灵活的方式使用各种系统功能，第 13 章中有关进程隐藏的内容就是最好的写照。所以理由之三就是：**使用 Win32 汇编语言是了解操作系统运行细节的最佳方式。**

最后的理由根本不是理由，而是必然的选择，当我们在 Windows 环境下进行加密解密、逆向工程，还有病毒、木马等有害代码的分析和防治工作时，Win32 汇编是惟一的选择，在任何讨论这方面内容的书籍中，汇编代码的篇幅总是很大的，要想深入了解这些内容的前提就是首先深入汇编编程。

How——如何学习 Win32 汇编

以往的汇编书籍往往把重点放在硬件结构和指令上，讲述了一大堆电路框图和指令列表，把大家搞得晕晕乎乎后，然后再举出一些重量级的例子，不是一些像数组、矩阵计算一类的复杂运算，就是开始图形模式画图，以至于大家看完以后就再也找不到北了！实际上，这些例子不是太难了，而是太枯燥了。有人说，学汇编就像考大学，千军万马过独木桥，太多的人中途放弃了，坚持到最后只有少数人。

笔者认为：学习汇编应该在轻松的环境下进行，在学习中使用的例子不一定太复杂，但一定要有吸引力。用汇编写复杂的运算程序固然会比 C 更有效率，但同样的事在 C 中用一个表达式就全部搞定了，从这里开始学汇编，给人的感觉就像从复杂的公式开始学算术，要知

道，加法还没有学会呢！而对于高级语言封装起来的系统功能，用汇编解释起来就非常直接，非常自然，也更容易懂。以笔者自己学汇编的过程来说，那时候是 1990 年，刚好是中国第一次病毒大流行，大家的计算机上都是那个病毒的开山鼻祖——乒乓病毒，在流行 DOS 的时期，看着在屏幕 89-456+1 上蹦的小球，心中就有一个问题：如何编出这样一个东西来呢？要知道 DOS 是单任务的，而那个球在别的程序运行的时候照样蹦！这用当时流行的 FORTRAN、C 等课程中学到的任何东西都无法解释，因为这些课程中不可能有 TSR，中断，引导区等内容，带着这样一个疑问学习汇编，在分析乒乓病毒的过程中啃一条条不懂的指令，病毒分析完了，汇编课也学完了，而且反过来看那些复杂的计算程序都是那么顺理成章，不攻自破了。实际上，从一些实用的系统功能开始学习汇编远比学矩阵计算容易理解。

正如最经典的 C 程序就是那个“Hello, World!”一样，这个程序的有名并不是因为它用高深复杂的语句放倒了一大批人，而是它以最简单易懂的方式让人们走入 C 语言的大门。对于 Win32 汇编也是如此，从最简单的例子开始总是没错的，笔者建议读者跟随本书中从简到繁的例子，努力做到理解并灵活引用这些例子中的各种功能，正如“熟读唐诗三百首，不会写诗也会吟”，最后能够熟练地使用 Win32 汇编来解决各种编程需求就是最大的胜利。

另外，正如前面讲到的，汇编语言的学习必须和操作系统紧密结合。经过简单的调查，笔者发现很多高校使用的汇编教程还是停留在清华 91 版《IBM-PC 汇编语言程序设计》之类的教材上，虽然这些教材中基础知识部分永远不会过时，但涉及操作系统的部分还是停留在 DOS 阶段，随着 DOS 操作系统的悄然引退，继续把精力花在上面是一种浪费，因为任何语言都必须有应用的平台，否则课程学完之后会尴尬地发现没有地方可以应用。笔者认为，在《IBM-PC 汇编语言程序设计》之类传统教材中的基础部分学习完毕以后，重点就应该转向 Win32 汇编以及保护模式方面的知识。

关于本书的内容

本书尝试从编写应用程序的角度，从“Hello, World”这个简单的例子开始到编写多线程、注册表和网络通信等复杂的程序，通过 60 多个从简单到复杂的例子，逐步深入 Win32 汇编编程的方方面面。笔者从事汇编编程已经有十几年的历史了，从 8086 时代的 DOS 汇编编程开始到当前的 Win32 汇编编程，从一个初学者到现在能利用 Win32 汇编来解决大部分编程需求，中间也经过了很长时间的摸索和大量的挫折，所以笔者很清楚初学者在哪些地方会遇到问题，但是涉及 Win32 汇编的书籍却实在太少了，正是因为如此，笔者决心把本书的目标定为：能让读者入门并在最后能熟练掌握 Win32 汇编编程，而不是那种深入系统奥秘一类的书籍。

从这个目标出发，本书的选材中尽量去掉已经有其他书籍详细讨论的部分，因为要一本书涉及全部方面是不现实的，内容全面就必然不精，内容深刻就必须围绕一个中心点，所以本书的内容并不详细讨论一般汇编教材的基础部分，如处理器结构和保护模式等，也不准备涉及 Windows 驱动程序、COM 编程或者其它能够冠以“密技”头衔的内容。本书主要的内容将放在 32 位宏汇编对比 DOS 汇编所不同的部分以及 Win32 应用程序的汇编实现上。不求全面，只求精也！（说句老实话，也不敢对自己不精通的地方妄加评论，以免破坏自己的良好形象☺）

在笔者的汇编编程论坛上(<http://win32asm.yeah.net>)，经常有初学者问到 MASM 和 TASM 有什么不同，用哪个比较好，@@ 标号是什么意思，为什么用下载的汇编编译器无法编译程

序等问题，虽然这些都属于最基本的问题，但是的确没有一个地方或者从一本书能系统地讲解这些问题，所以本书的基础篇将系统地讲解这些问题，它们是：

- 第 1 章 背景知识
- 第 2 章 准备编程环境
- 第 3 章 使用 MASM

当搭建编译和对编译器的使用不再成为绊脚石的时候，初学者的问题往往集中在对 Windows 程序结构的迷惑上，消息驱动体系、窗口过程、与硬件隔绝的图形接口及资源文件等相对于 DOS 程序来说都是全新的内容，接下来的 4 章将深入讨论这些内容，通过这几章，读者应该开始习惯以 Windows 的方式考虑问题了（脑海中的 DOS 逐渐远去...），这就是本书的初级篇：

- 第 4 章 第一个窗口程序
- 第 5 章 使用资源
- 第 6 章 定时器
- 第 7 章 图形操作

Windows 系统不像 DOS 系统，它的应用程序界面是规范化的，统一的界面来自大量的系统界面控件，学习这些控件就等于学习如何编写 Windows 界面，下面的界面篇中的两章将探讨这方面的内容：

- 第 8 章 通用对话框
- 第 9 章 通用控件

到这里为止，读者学后应该可以写出界面规范的标准 Win32 程序了，但还是无法用这些程序来解决一些具体问题，因为有关 Windows 系统的高级特征的介绍还没有开始，如内存管理、文件操作和多线程等，这些就是本书高级篇中将要介绍的内容，通过这些内容，读者将比较深入地了解 Windows 的工作方式：

- 第 10 章 内存管理和文件操作
- 第 11 章 动态链接库和钩子
- 第 12 章 多线程
- 第 13 章 进程控制
- 第 14 章 异常处理

相信到这里为止，读者对 Windows 的了解已经是比较系统了，虽然 Windows 中还存在其他很多方面的内容，如管道，邮件槽，ODBC 接口，如何写控制面板程序，屏幕保护程序和驱动程序等，但是有了前面的基础以后，读者自己去了解这些内容就不成问题，因为掌握了“渔”，得到“鱼”又有什么困难呢？在最后的几章中，本书将从应用的角度再补充介绍一些常用的网络编程，注册表和 PE 文件操作方面的内容，这就是应用篇：

- 第 15 章 注册表和 INI 文件
- 第 16 章 TCP/IP 和网络通信
- 第 17 章 PE 文件

在本书中，笔者特别以显著的方式标出了一些经验之谈，这些是笔者在长期的汇编编程中得到的体会，可能是任何一本教科书或者手册里都没有的。希望这些能对读者带来帮助！



用“灯泡”标出的部分表示一些小技巧，可以对编程的理解有促进作用。



用“惊叹号”标出的部分表示容易出错的部分，可以帮助读者避免一些难以理解的错误。

对读者的假设

有了内容的定位，读者的定位也就比较清楚了，本书适合于以下读者：

- 想用 Win32 汇编写 Windows 程序的读者。
- 想从 DOS 下的 16 位汇编转向 Windows 下 32 位汇编的读者。
- 欲了解 Win32 汇编，以便为 Windows 下的加密解密、系统安全、逆向工程等方面打基础的读者。

● 欲了解 Win32 汇编，以便为用汇编写 Windows 驱动程序打基础的读者。

● 正在学习汇编课程，需要补充汇编课程中 Win32 部分的学生。

在开始本书之前，读者应该有以下的基础知识：

- 计算机的基础知识，如进制转换、逻辑运算、变量类型和指针的概念等。
- 数据结构的基础知识，因为 Win32 编程涉及大量的数据结构。
- C 语言的基础知识，因为 Win32 编程的绝大部分参考资料都是以 C 的格式出现的。
- Intel 80x86 处理器的基础知识，如寻址方式和指令的使用等等。

本书并不是为以下读者准备的：

● 欲详细了解保护模式的读者——因为 Windows 并不是一个开放的平台，Windows 的开放只限于应用程序接口，所以要用 Windows 做背景研究保护模式只能是自讨苦吃，如果读者需要这方面的深入了解，最好的方法就是去研究 Linux 的核心代码并在 Linux 上实验。

● 欲了解 Windows 核心“机密”的读者——汇编并不等同于深入操作系统的内部，所以本书不是《Windows 内核分析》，而真正意义上的《Windows 内核分析》除了 Microsoft，恐怕谁也写不出来。

关于附书代码和读者反馈

为了更好地说明 Win32 汇编的编程方法，本书附带了 60 多个例子，这些例子的源代码全部可以在附书光盘中找到，代码全部采用 MASM 格式编写，推荐使用的编译软件为 MASM32 软件包。

MASM32 软件包可以在以下地址下载：

MASM32 官方站点

<http://www.movsd.com>

作者的 Win32 汇编编程站点

<http://asm.yeah.net>

虽然本书中所有的例子代码都已经在 Windows 98、Windows 2000 和 Windows XP 下测试通过，但也有存在 Bug 的可能，如果发现代码存在错误或者发现书中有其他问题，请告知作

者，以便在下一个版本中改进。如果读者有任何的反馈意见——不管是批评还是鼓励，也请和作者联系，联系方法是：

在作者主页的论坛中通知作者

<http://asm.yeah.net>

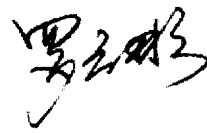
<http://win32asm.yeah.net>

给作者发 E-mail

luoyunbin@sina.com

luoyunbin@hotmail.com

bigluo@telekbird.com.cn



2002年8月

目 录

第 1 章 背景知识	1
1.1 Win32 的软硬件平台	1
1.1.1 80x86 系列处理器简史	1
1.1.2 Windows 的历史	3
1.1.3 Win32 平台的背后——Wintel 联盟	4
1.2 Windows 的特色	5
1.3 必须了解的东西	6
1.3.1 80x86 处理器的工作模式	6
1.3.2 Windows 的内存管理	8
1.3.3 Windows 的特权保护	15
第 2 章 准备编程环境	19
2.1 Win32 可执行文件的开发过程	19
2.2 编译器和链接器	20
2.2.1 MASM 系列	21
2.2.2 TASM 系列	24
2.2.3 其他编译器	26
2.2.4 MASM, TASM 还是 NASM	27
2.2.5 我们的选择——MASM32 软件包	28
2.3 创建资源	29
2.3.1 资源编译器的使用	29
2.3.2 所见即所得的资源编辑器	30
2.4 make 工具的用法	32
2.4.1 make 工具是什么	32
2.4.2 nmake 的用法	33
2.4.3 描述文件的语法	34
2.5 获取资料	38
2.5.1 Windows 资料的来源	38
2.5.2 Intel 处理器资料	40
2.6 构建编程环境	40
2.6.1 IDE 还是命令行	40
2.6.2 本书推荐的工作环境	41
2.6.3 尝试编译第一个程序	42
第 3 章 使用 MASM	44
3.1 Win32 汇编源程序的结构	44
3.1.1 模式定义	46

3.1.2	段的定义	48
3.1.3	程序结束和程序入口	51
3.1.4	注释和换行	51
3.2	调用 API	52
3.2.1	API 是什么	52
3.2.2	调用 API	54
3.2.3	API 参数中的等值定义	58
3.3	标号、变量和数据结构	60
3.3.1	标号	60
3.3.2	全局变量	61
3.3.3	局部变量	63
3.3.4	数据结构	66
3.3.5	变量的使用	68
3.4	使用子程序	73
3.4.1	子程序的定义	73
3.4.2	参数传递和堆栈平衡	74
3.5	高级语法	77
3.5.1	条件测试语句	77
3.5.2	分支语句	79
3.5.3	循环语句	81
3.6	代码风格	83
3.6.1	变量和函数的命名	83
3.6.2	代码的书写格式	85
3.6.3	代码的组织	87
第 4 章	第一个窗口程序	88
4.1	开始了解窗口	88
4.1.1	窗口是什么	88
4.1.2	窗口界面	89
4.1.3	窗口程序是怎么工作的	90
4.2	分析窗口程序	97
4.2.1	模块和句柄	97
4.2.2	创建窗口	99
4.2.3	消息循环	106
4.2.4	窗口过程	108
4.3	窗口间的消息互发	112
4.4	实验	116
4.4.1	MsgWindow 程序	116
4.4.2	开始实验	120
第 5 章	使用资源	125
5.1	菜单和加速键	125

5.1.1	菜单和加速键的组成	125
5.1.2	菜单和加速键的资源定义	126
5.1.3	使用菜单和加速键	132
5.2	图标和光标	145
5.2.1	图标和光标的资源定义	145
5.2.2	使用图标和光标	146
5.3	位图	150
5.3.1	位图简介	150
5.3.2	在资源中定义位图	151
5.4	对话框	152
5.4.1	对话框简介	152
5.4.2	对话框的资源定义	154
5.4.3	使用对话框	156
5.4.4	在对话框中使用子窗口控件	159
5.5	字符串资源	183
5.6	版本信息资源	184
5.6.1	版本信息资源的定义	185
5.6.2	在程序中检测版本信息	188
5.7	二进制资源和自定义资源	189
5.7.1	使用二进制资源	189
5.7.2	使用自定义资源	190
第 6 章	定时器	192
6.1	定时器简介	192
6.2	定时器的使用	193
6.3	取 Windows 时间	197
第 7 章	图形操作	199
7.1	GDI 原理	199
7.1.1	GDI 程序的结构	199
7.1.2	设备环境	203
7.1.3	色彩和坐标	210
7.2	绘制图形	212
7.2.1	画笔和画刷	220
7.2.2	绘制像素点	223
7.2.3	绘制图形	224
7.2.4	绘图模式	228
7.3	创建和使用位图	229
7.3.1	一个使用位图的时钟例子	229
7.3.2	创建和使用位图	241
7.3.3	使用设备无关位图	242
7.4	块传送操作	245

7.4.1	块传送方式	245
7.4.2	块传送函数	246
7.5	区域和路径	250
7.5.1	使用区域	250
7.5.2	使用路径	252
第 8 章	通用对话框	254
8.1	通用对话框简介	254
8.2	使用通用对话框	262
8.2.1	“打开”文件和“保存”文件对话框	262
8.2.2	选择字体对话框	264
8.2.3	选择颜色对话框	266
8.2.4	查找和替换文本对话框	267
8.2.5	页面设置对话框	270
8.3	浏览目录对话框	272
8.3.1	浏览目录对话框简介	272
8.3.2	使用浏览目录对话框	277
第 9 章	通用控件	280
9.1	通用控件简介	280
9.1.1	通用控件的分类	280
9.1.2	使用通用控件	282
9.2	使用状态栏	286
9.2.1	创建状态栏	292
9.2.2	状态栏的控制消息	293
9.2.3	在状态栏上显示菜单提示信息	295
9.3	使用工具栏	296
9.3.1	创建工具栏	304
9.3.2	工具栏的控制消息	308
9.3.3	工具栏的通知消息	310
9.4	使用 Richedit 控件	314
9.4.1	创建 Richedit 控件	328
9.4.2	Richedit 控件的控制消息	330
9.4.3	Richedit 控件的通知消息	339
9.5	窗口的子类化	340
9.5.1	什么是窗口的子类化	340
9.5.2	窗口子类化的实现	341
9.6	控件的超类化	348
9.6.1	什么是控件的超类化	348
9.6.2	控件超类化的实现	348
第 10 章	内存管理和文件操作	354
10.1	内存管理	354

10.1.1	内存管理基础	354
10.1.2	内存的当前状态	355
10.1.3	标准内存管理函数	359
10.1.4	堆管理函数	368
10.1.5	虚拟内存管理函数	372
10.1.6	其他内存管理函数	376
10.2	文件操作	378
10.2.1	Windows 的文件 I/O	378
10.2.2	创建和读写文件	379
10.2.3	查找文件	394
10.2.4	文件属性	402
10.2.5	其他文件操作	404
10.3	驱动器和目录	405
10.3.1	逻辑驱动器操作	406
10.3.2	目录操作	409
10.4	内存映射文件	411
10.4.1	内存映射文件简介	411
10.4.2	使用内存映射文件	413
第 11 章	动态链接库和钩子	423
11.1	动态链接库	423
11.1.1	动态链接库的概念	423
11.1.2	编写动态链接库	424
11.1.3	使用动态链接库	429
11.1.4	动态链接库中的数据共享	437
11.2	Windows 钩子	439
11.2.1	什么是 Windows 钩子	439
11.2.2	远程钩子的安装和使用	441
11.2.3	日志记录钩子	449
第 12 章	多线程	453
12.1	进程和线程	453
12.2	多线程编程	454
12.2.1	一个单线程的“问题程序”	454
12.2.2	多线程的解决方法	458
12.2.3	与线程有关的函数	463
12.3	使用事件对象控制线程	467
12.3.1	事件	468
12.3.2	等待事件	469
12.3.3	进一步改进计数程序	470
12.4	线程间的同步	473
12.4.1	产生同步问题的原因	473

12.4.2	临界区	478
第 13 章	进程控制	484
13.1	环境变量和命令行参数	484
13.1.1	环境变量	484
13.1.2	命令行参数	487
13.2	执行可执行文件	493
13.2.1	方法一: Shell 调用	493
13.2.2	方法二: 创建进程	494
13.3	进程调试	504
13.3.1	获取运行中的进程句柄	505
13.3.2	读写进程的地址空间	511
13.3.3	调试 API 的使用	516
13.4	进程的隐藏	526
13.4.1	在 Windows 9x 中隐藏进程	526
13.4.2	Windows NT 中的远程线程	528
第 14 章	异常处理	540
14.1	异常处理的用途	540
14.2	使用筛选器处理异常	541
14.2.1	注册回调函数	541
14.2.2	异常处理回调函数	543
14.3	使用 SEH 处理异常	546
14.3.1	注册回调函数	548
14.3.2	异常处理回调函数	550
14.3.3	SEH 链和异常的传递	554
14.3.4	展开操作 (Unwinding)	555
第 15 章	注册表和 INI 文件	559
15.1	注册表和 INI 文件简介	559
15.2	INI 文件的操作	560
15.2.1	INI 文件的结构	560
15.2.2	管理键值	561
15.2.3	管理小节	570
15.2.4	使用不同的 INI 文件	571
15.3	对注册表的操作	572
15.3.1	注册表的结构	572
15.3.2	管理子键	574
15.3.3	管理键值	586
15.3.4	子键和键值的枚举	587
15.3.5	注册表应用举例	590
第 16 章	TCP/IP 和网络通信	594
16.1	网络基础知识	594

16.1.1	TCP/IP 协议	594
16.1.2	一些重要概念	598
16.2	WinSock 接口	601
16.2.1	WinSock 接口简介	601
16.2.2	WinSock 编程概述	603
16.3	TCP 协议编程	611
16.3.1	TCP 协议简介	611
16.3.2	TCP 聊天室例子——客户端	613
16.3.3	TCP 聊天室例子——服务器端	624
16.4	UDP 协议编程	632
16.4.1	UDP 聊天室例子——客户端	633
16.4.2	UDP 聊天室例子——服务器端	640
16.5	ICMP 协议编程	646
16.5.1	题外话：控制台程序	647
16.5.2	ICMP 协议	651
16.5.3	一个 Ping 程序例子	654
第 17 章	PE 文件	667
17.1	PE 文件的结构	667
17.1.1	概论	667
17.1.2	DOS 文件头和 DOS 块	668
17.1.3	PE 文件头 (NT 文件头)	670
17.1.4	节表和节	675
17.2	导入表	691
17.2.1	导入表简介	691
17.2.2	导入表的结构	693
17.2.3	查看 PE 文件导入表举例	696
17.3	导出表	699
17.3.1	导出表的结构	699
17.3.2	查看 PE 文件导出表举例	702
17.4	资源	706
17.4.1	资源简介	706
17.4.2	资源的组织方式	707
17.4.3	查看 PE 文件中的资源列表举例	710
17.5	重定位表	716
17.5.1	重定位表的结构	716
17.5.2	查看 PE 文件的重定位表举例	718
17.6	应用实例	721
17.6.1	动态获取 API 入口地址	721
17.6.2	在 PE 文件上添加执行代码	728
参考文献		736