

精通

Windows API

——函数、接口、编程实例

范文庆 周彬彬 安靖 编著

从Windows应用程序开发工具讲起，引导读者快速入门
详细讲解API和关键结构的使用方法
提供156个Windows API使用实例



人民邮电出版社
POSTS & TELECOM PRESS



CD-ROM

精通

TP316.7
330
12

光盟

Windows API

— 函数、接口、编程实例

范文庆 周彬彬 安靖 编著



人民邮电出版社
北京

图书在版编目 (C I P) 数据

精通Windows API: 函数、接口、编程实例 / 范文庆, 周彬彬, 安靖编著. —北京: 人民邮电出版社, 2009. 2
ISBN 978-7-115-19095-6

I. 精… II. ①范…②周…③安… III. 窗口软件, Windows—软件接口—程序设计 IV. TP316.7

中国版本图书馆CIP数据核字 (2008) 第168772号

内 容 提 要

Windows API (Windows Application Program Interface, Windows 应用程序接口) 是一系列函数、宏、数据类型、数据结构的集合, 运行于 Windows 系统的应用程序, 可以使用操作系统提供的接口来实现需要的功能。本书由浅入深、循序渐进地教授读者如何使用 Windows API 进行 Windows 应用程序开发。全书共 18 章, 分为 3 个部分, 第 1 部分 (第 1 章~第 3 章) 介绍 Windows 程序设计基础; 第 2 部分 (第 4 章~第 17 章) 按照程序设计的各个方面进行划分, 包括文件系统、内存管理、进程与线程、用户界面、Shell 程序开发、Windows GDI、Socket 网络通信驱动程序开发、安全机制等内容; 第 3 部分 (第 18 章) 作为全书的总结和补充。

本书适合广大的 Windows 应用程序开发人员、Visual C++ 开发工程师、网络游戏开发人员、软件培训机构学员和高校学生阅读。

精通 Windows API——函数、接口、编程实例

◆ 编 著 范文庆 周彬彬 安 靖

责任编辑 屈艳莲

执行编辑 黄 焱

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

北京鑫正大印刷有限公司印刷

◆ 开本: 787×1092 1/16

印张: 35

字数: 922 千字

2009 年 2 月第 1 版

印数: 1-4 000 册

2009 年 2 月北京第 1 次印刷

ISBN 978-7-115-19095-6/TP

定价: 69.00 元 (附光盘)

读者服务热线: (010)67132692 印装质量热线: (010)67129223

反盗版热线: (010)67171154

前 言

关于 Windows API

Windows API (Windows Application Program Interface) 是一系列函数、宏、数据类型、数据结构的集合。运行于 Windows 系统的应用程序可以使用操作系统提供的接口来实现应用程序需要的功能。Windows 开发平台是所有程序开发平台中非常重要的一种，是程序设计中一个重要的方面。本书介绍了 Windows 应用程序开发的各个方面。

本书的写作方法

程序设计是一门实践性很强的学科，本书在编写过程中十分注重与实际开发工作相联系，在讲解每一个编程要点时，都以实例为核心进行分析、演示，并把实际工程中开发人员可能遇到的问题以实例的方式向读者讲解。

与此同时，本书同样重视基础知识的讲解，重视原理的说明，对程序所使用的每一个 API 和每一个结构的使用方法都尽量详细说明。本书还在第 18 章详细介绍了 Windows 系统调用的内部过程。

解决开发人员实际编码过程中可能出现的问题也是本书的一个主要目的，因此本书使用了大量篇幅来讲解编译选项等最容易被忽略，但是却是最容易在程序设计中出现问题，也是常常使得程序员不知道如何解决的问题。

本书主要内容

本书共分 18 章，涵盖了 Windows 应用程序设计的基本内容。第 1、2、3 章是 Windows 程序设计的基础。第 4 章~第 17 章按程序设计的方面划分，每个方面一章。第 18 章是全书的总结和提高。

第 1 章主要讲解如何配置 Windows 应用程序的开发工具，从最简单的实例入手，使读者对 Windows 应用程序的开发有一个初步的认识。

第 2 章对 Windows API 作了简要的介绍，包括 Windows API 基本的功能分类、数据类型等。

第 3 章介绍了开发工具配置与使用，包括基本的编译链接工具 `cl.exe`、`rc.exe`、`link.exe`，Platform SDK 的使用，Makefile 的编写以及使用 WinDbg 的调试方法等。

第4章文件系统，主要介绍了磁盘和驱动器管理、文件和目录管理、内存映射文件等内容。

第5章内存管理，主要介绍了内存管理原理、堆原理、全局和局部内存管理、虚拟内存管理、内存操作与内存消息管理等内容。

第6章进程、线程和模块，主要介绍了进程管理，线程、纤程的创建、删除以及获取方法，获取进程状态信息，动态链接库的加载、释放和获取方法等内容。

第7章线程同步，主要介绍了线程同步的基本原理、同步对象等内容。

第8章服务，主要介绍了基本概念、服务程序的编写、对服务程序的控制和管理等内容。

第9章图形用户界面，主要介绍了字符界面程序，窗口、控件、菜单、对话框等界面元素的使用方法。

第10章系统信息的管理，主要介绍了系统基本信息、时间信息以及注册表信息的管理。

第11章进程间通信，主要介绍了邮槽、管道、剪贴板的创建与使用方法，数据复制消息的使用方法等内容。

第12章 Windows Shell 程序设计，主要介绍了 Windows Shell 的目录管理、文件协助、Shell 扩展等内容。

第13章 Windows GDI，主要介绍了使用 GDI 指定文字的字体、绘制线条、绘制图形、对位图的操作、坐标变换的方法等内容。

第14章网络通信与配置，主要介绍了 Socket 通信的相关内容以及 IP Helper 的使用方法。

第15章程序安装与设置，主要介绍了 cab 文件的创建、INF 文件的创建、安装程序 setup.exe 的编写、使用 msi 文件进行安装等内容。

第16章设备驱动管理与内核通信，主要介绍了与设备有关的 API 函数、如何对设备驱动进行控制、如何编写设备驱动程序等内容。

第17章用户、认证和对象安全，主要介绍了数据认证中的基本概念、安全机制程序示例以及用户的增加、删除、权限更改等内容。

第18章 Windows API 的内部原理，主要介绍了 Windows 系统中的对象封装、x86 平台程序函数调用、可执行程序结构与 API 函数接口内部机理等内容。

参与本书编写的人员

本书由范文庆、周彬彬、安靖负责编写并统编全书稿，另外感谢以下人员为本书所做的工作：张墨、郭永红、周瑜、王建伟、孙琼、田旭、范文庆、钟金鑫、王欣、张曦文、尚玉珊、张丛辉、王玮、刘超、张圣亮、李凡、马堃、徐路迎、赵国锋、孙颂武、汪荷君、孙明、林雪梅、黄惠英、刘雯等。

由于时间仓促，加之水平有限，书中不足之处在所难免，敬请读者批评指正。本书责任编辑的联系方式是 huangyan@ptpress.com.cn，欢迎来信交流。

编者
2009年1月

目 录

第 1 章 Windows 应用程序开发入门..... 1	2.2 Windows API 的功能分类..... 15
1.1 第一个实例程序..... 1	2.2.1 系统基本服务..... 15
1.1.1 start.exe 1	2.2.2 系统管理..... 17
1.1.2 Windows API..... 2	2.2.3 用户界面..... 17
1.1.3 程序入口函数..... 2	2.2.4 图像和多媒体..... 20
1.1.4 start.c 代码分析..... 2	2.2.5 网络..... 20
1.2 编译代码..... 3	2.2.6 系统安全..... 20
1.2.1 安装 Visual Studio 3	2.2.7 其他功能..... 21
1.2.2 安装 Microsoft Platform SDK 4	2.3 Windows API 核心 DLL 21
1.2.3 集成 Microsoft Platform SDK 与 Visual C++速成版..... 5	2.3.1 Kernel32.dll 21
1.2.4 Vista SDK 与 Visual Studio 2008 6	2.3.2 User32.dll..... 21
1.2.5 Visual Studio 专业版或团队系统版..... 7	2.3.3 Gdi32.dll 22
1.2.6 使用图形化 IDE 建立工程、进行编译 7	2.3.4 标准 C 函数..... 22
1.2.7 “解决方案”与“工程”..... 8	2.3.5 其他 DLL 22
1.2.8 使用命令行工具编译..... 8	2.4 Unicode 和多字节..... 22
第 2 章 Windows API 概要..... 10	2.4.1 W 版本和 A 版本的 API..... 24
2.1 Windows 数据类型..... 10	2.4.2 Unicode 与 ASCII 的转换..... 24
2.1.1 Windows 数据类型示例..... 10	2.5 对 Windows 程序设计规范的建议..... 25
2.1.2 Windows 数据类型与标准 C 数据 类型的关系..... 14	第 3 章 开发工具配置与使用..... 26
2.1.3 Windows 数据类型与 Windows API..... 14	3.1 使用 Visual C/C++编译链接工具..... 26
2.1.4 Windows 中的数据结构..... 15	3.1.1 编译器 cl.exe 27
	3.1.2 资源编译器 rc.exe..... 31
	3.1.3 链接器 link.exe..... 32
	3.1.4 其他工具..... 38
	3.1.5 编译链接工具依赖的环境变量..... 39

3.1.6 示例：使用/D选项进行条件编译	42	4.3.1 删除、复制、重命名、移动文件	87
3.2 使用 Platform SDK	43	4.3.2 创建、打开、读写文件，获取文件大小	90
3.2.1 Platform SDK 的目录结构与功能	43	4.3.3 创建目录	96
3.2.2 为编译链接工具设置环境变量	45	4.3.4 获取程序所在的目录、程序模块路径，获取和设置当前目录	97
3.2.3 Platform SDK 工具集	46	4.3.5 查找文件、遍历指定目录下的文件和子目录	100
3.2.4 Windows Vista SDK	48	4.3.6 递归遍历目录树	103
3.3 编写 Makefile	48	4.3.7 获取、设置文件属性和时间	105
3.3.1 使用 nmake.exe 构建工程	48	4.4 内存映射文件	110
3.3.2 Makefile 实例	50	4.4.1 使用 Mapping File 提高文件读写的效率	110
3.3.3 注释	50	4.4.2 通过 Mapping File 在进程间传递和共享数据	115
3.3.4 宏	50	4.4.3 通过文件句柄获得文件路径	118
3.3.5 描述块：目标、依赖项和命令	53	4.5 总结	121
3.3.6 makefile 预处理	55	第5章 内存管理	122
3.3.7 在 Platform SDK 的基础上使用 nmake	56	5.1 Windows 内存管理原理	122
3.4 使用 WinDbg 调试	57	5.1.1 基本概念	122
3.4.1 安装 WinDbg	57	5.1.2 分页与分段内存管理、内存映射与地址转换	123
3.4.2 编译可调试的程序	58	5.1.3 进程的内存空间	125
3.4.3 WinDbg 命令	59	5.1.4 虚拟内存布局、内存的分工、堆与栈	127
3.4.4 调试过程演示	59	5.1.5 内存的保护属性和存取权限	127
3.5 集成开发环境 Visual Studio	62	5.1.6 本章 API 列表	127
3.5.1 工程类型选择与配置	62	5.2 堆管理	129
3.5.2 Visual Studio 快捷方式	64	5.2.1 获取堆句柄、分配与再分配堆	129
3.5.3 生成项目	64	5.2.2 获取堆中内存块的大小信息	133
3.5.4 调试	65	5.2.3 释放内存、销毁堆	134
3.5.5 选项与设置	65	5.3 全局 (Global) 和局部 (Local) 内存管理	136
3.6 开发环境配置总结	66	5.3.1 Global 函数	136
第4章 文件系统	67	5.3.2 Local 函数	137
4.1 概述	67	5.3.3 使用全局和局部函数分配和释放内存、改变内存块属性	137
4.1.1 文件系统的基本概念	67	5.4 虚拟内存管理	138
4.1.2 文件系统主要 API	68		
4.2 磁盘和驱动器管理	70		
4.2.1 遍历卷并获取属性	70		
4.2.2 操作驱动器挂载点	76		
4.2.3 判断光驱中是否有光盘	81		
4.2.4 获取磁盘分区的总容量、空闲容量、簇、扇区信息	83		
4.3 文件和目录管理	86		

5.4.1 虚拟地址空间与内存分页	139	6.4 进程状态信息	176
5.4.2 分配和释放可读可写的虚拟内存 页面	139	6.4.1 PS API 与 Tool help API	176
5.4.3 修改内存页面状态和保护属性、 将页面锁定在物理内存中	142	6.4.2 遍历系统中的进程	178
5.4.4 管理其他进程的虚拟内存	143	6.4.3 列举进程的模块、线程	182
5.5 内存操作与内存信息管理	144	6.4.4 进程的堆使用、内存占用、虚拟 内存大小, 页面错误情况	184
5.5.1 复制、填充、移动、清零内存块、 防止缓冲区溢出	144	6.5 动态链接库	185
5.5.2 获得当前系统内存使用情况	146	6.5.1 加载、释放 DLL、通过句柄获取 DLL 相关信息	186
5.5.3 判断内存指针的可用性	147	6.5.2 编写动态链接库、导出函数	186
5.6 各种内存分配方式的关系与比较	148	6.5.3 创建动态链接库工程, 配置 DLL 编译链接选项	188
5.6.1 标准 C 内存管理函数与 Windows 内存管理 API 的关系	149	6.5.4 运行时动态获取 DLL 导出函数 地址并调用	189
5.6.2 功能性区别	149	6.5.5 声明导出函数、创建 lib 库, 为其他 模块提供导入表调用接口	190
5.6.3 效率的区别	149	6.5.6 通过构建导入表调用 DLL 导出 函数	191
第 6 章 进程、线程和模块	150	第 7 章 线程同步	192
6.1 基本概念	150	7.1 基本原理	192
6.1.1 应用程序与进程	150	7.1.1 线程同步的过程	193
6.1.2 控制台应用程序与图形用户界面 应用程序	151	7.1.2 同步对象	193
6.1.3 动态链接库、模块	151	7.1.3 等待函数	193
6.1.4 线程、纤程与作业	152	7.2 同步对象示例	194
6.1.5 权限与优先级	153	7.2.1 使用事件对象 (Event)	194
6.2 进程管理	153	7.2.2 使用互斥对象 (Mutex)	199
6.2.1 创建进程、获取进程相关信息、 获取启动参数	153	7.2.3 使用信号量控制访问共享数据的 线程数量	202
6.2.2 编写控制台程序和图形用户界面 应用程序	158	7.2.4 使用可等待计时器 (Timer)	206
6.2.3 获取和设置环境变量	158	7.3 等待进程和线程的执行完成	209
6.3 线程、纤程	162	第 8 章 服务	210
6.3.1 创建线程、退出线程、获取线程 信息	162	8.1 基本概念	210
6.3.2 挂起、恢复、切换、终止线程	164	8.1.1 服务控制器 (SCM)	211
6.3.3 创建远程线程、将代码注入其他 进程中执行	167	8.1.2 服务程序	211
6.3.4 创建纤程、删除纤程、调度纤程	170	8.1.3 服务控制管理程序	211
6.3.5 纤程与线程的互相转换	171	8.1.4 系统服务管理工具	211
		8.1.5 服务的属性	211

8.2 编写服务程序.....	212	9.4.7 文本框控件.....	267
8.2.1 入口函数.....	212	9.4.8 为文本框控件设置文字.....	268
8.2.2 服务主函数.....	212	9.5 界面资源.....	269
8.2.3 控制处理函数.....	213	9.5.1 资源脚本 (.rc).....	269
8.3 实现对服务的控制和管理.....	216	9.5.2 资源 ID 定义和头文件.....	272
8.3.1 创建、删除服务.....	216	9.5.3 在程序中使用资源.....	273
8.3.2 启动、停止服务, 向服务发送控制 请求.....	219	9.6 菜单.....	273
8.3.3 管理服务状态、配置服务、服务的 依赖关系.....	222	9.6.1 菜单资源和菜单句柄.....	273
第 9 章 图形用户界面.....	229	9.6.2 动态增加、删除、设置菜单及 菜单项.....	274
9.1 字符界面程序.....	229	9.6.3 菜单消息处理.....	274
9.1.1 基本概念.....	230	9.7 对话框.....	275
9.1.2 控制台读写.....	231	9.7.1 创建对话框.....	275
9.1.3 控制台字体、颜色等属性, 操作 屏幕缓存.....	234	9.7.2 对话框消息处理函数.....	276
9.1.4 控制台事件.....	244	第 10 章 系统信息的管理.....	277
9.2 图形用户界面: 基本概念.....	246	10.1 Windows 系统信息.....	277
9.2.1 窗口.....	246	10.1.1 获取系统版本.....	277
9.2.2 窗口类.....	246	10.1.2 获取计算机硬件信息.....	279
9.2.3 消息和消息处理函数.....	247	10.1.3 获取系统目录等信息.....	281
9.2.4 控件.....	247	10.1.4 用户名、计算机名、域名.....	282
9.2.5 资源.....	248	10.1.5 处理系统颜色信息、尺度 信息等.....	284
9.2.6 对话框.....	248	10.1.6 鼠标、键盘等外设信息.....	285
9.3 图形用户界面: 窗口.....	248	10.2 时间信息.....	286
9.3.1 注册窗口类.....	249	10.2.1 设置、获取系统时间.....	286
9.3.2 创建窗口.....	251	10.2.2 获取开机至现在持续的时间.....	287
9.3.3 窗口消息处理函数.....	253	10.2.3 文件时间与系统时间的转换.....	287
9.3.4 窗口属性、位置和大小.....	256	10.3 注册表.....	288
9.3.5 窗口显示方式.....	257	10.3.1 注册表的作用及组织形式.....	288
9.3.6 线程消息队列和消息循环.....	258	10.3.2 键、子键、键属性及键值的 相关操作.....	289
9.4 图形用户界面: 控件.....	258	10.3.3 列举注册表项及键值.....	292
9.4.1 Tree View 控件.....	258	10.3.4 通过注册表设置一个自启动的 程序.....	293
9.4.2 为 Tree View 控件增加节点.....	260	10.3.5 设置随程序启动而启动的调试器 (任何程序).....	294
9.4.3 Tree View 右键菜单.....	262	10.3.6 指定程序崩溃实时调试器.....	294
9.4.4 List View 控件.....	263		
9.4.5 为 List View 控件增加分栏.....	265		
9.4.6 为 List View 控件增加项.....	266		

第 11 章 进程间通信	295	12.3.3 注册 Shell 扩展.....	345
11.1 邮槽 (MailSlot)	295	12.3.4 COM 程序开发基础.....	346
11.1.1 创建邮槽、从邮槽中读取消息.....	296	12.3.5 编写 Handler 程序.....	346
11.1.2 通过邮槽发送消息.....	299	12.3.6 Shell 扩展程序的调试.....	362
11.2 管道 (Pipe)	300	12.3.7 总结.....	363
11.2.1 创建命名管道.....	300	12.4 任务栏通知区域 (Tray) 图标.....	363
11.2.2 管道监听.....	302	12.4.1 创建图标窗口.....	364
11.2.3 使用异步 I/O 进行读写.....	303	12.4.2 创建图标和图标菜单.....	367
11.2.4 关闭管道实例.....	307	12.4.3 最小化主窗口到通知区域.....	370
11.2.5 客户端.....	307	12.4.4 弹出气泡通知.....	372
11.3 剪贴板.....	310	12.4.5 动态图标.....	374
11.3.1 获取、设置剪贴板数据.....	310	12.4.6 其他功能.....	376
11.3.2 监视剪贴板.....	317	第 13 章 Windows GDI	379
11.3.3 剪贴板数据格式.....	325	13.1 GDI 编程接口概述.....	379
11.4 数据复制消息 (WM_COPYDATA)	327	13.1.1 Windows GDI 的功能.....	379
11.4.1 数据发送端.....	327	13.1.2 链接库与头文件.....	380
11.4.2 数据接收端.....	330	13.2 设备上下文 (DC)、输出操作与图形对象.....	380
11.5 其他进程间通信方式.....	332	13.2.1 设备上下文类型与关联设备.....	380
11.5.1 动态数据交换 (DDE) 和网络动态数据交换 (NDDE)	332	13.2.2 图形对象的作用及与 DC 的关系.....	380
11.5.2 通过 File Mapping 在进程间共享数据.....	333	13.2.3 各类图形对象的具体属性与作用.....	383
11.5.3 Windows Socket.....	333	13.2.4 绘制、填充、写入等图形输出操作.....	384
第 12 章 Windows Shell 程序设计	334	13.2.5 修剪与坐标变换.....	385
12.1 Windows Shell 目录管理.....	335	13.2.6 设备上下文的图形模式.....	385
12.1.1 Shell 对目录和文件的管理形式.....	335	13.3 一个最简单的 GDI 程序.....	386
12.1.2 “我的文档”等特殊目录相关操作.....	335	13.3.1 示例.....	386
12.1.3 绑定、遍历、属性获取.....	337	13.3.2 DC 的操作.....	387
12.1.4 浏览文件对话框.....	339	13.3.3 颜色的表示.....	388
12.2 文件协助 (File Associations)	340	13.3.4 图形对象: 画刷和画笔.....	389
12.2.1 文件类型相关注册表键值.....	340	13.3.5 输出操作: 绘制图形和线条.....	390
12.2.2 为文件指定默认打开程序.....	341	13.4 文字和字体.....	391
12.2.3 定制文件类型的图标.....	342	13.4.1 选择、设置字体.....	393
12.3 Shell 扩展.....	343	13.4.2 选择字体图形对象.....	394
12.3.1 对象及概念.....	343	13.4.3 文字的颜色.....	394
12.3.2 CLSID, 处理例程的 GUID.....	344	13.4.4 输出文字.....	395
		13.4.5 DC 图形模式设置.....	395
		13.4.6 遍历字体.....	396

13.4.7 为系统安装、删除字体文件	398	14.2 IP Helper	456
13.5 绘制线条	398	第 15 章 程序安装与设置	463
13.5.1 选择画笔对象	399	15.1 创建 cab 文件	463
13.5.2 直线	399	15.1.1 makecab.exe	463
13.5.3 绘制任意曲线	399	15.1.2 压缩多个文件	464
13.5.4 跟踪鼠标轨迹	399	15.1.3 Cabinet 软件开发工具包 (CABSDK)	466
13.5.5 弧线	405	15.2 编写 INF 文件	466
13.6 绘制图形	405	15.2.1 INF 文件格式	466
13.6.1 填充颜色与边缘勾勒	406	15.2.2 Install 节	468
13.6.2 绘制矩形、椭圆、圆角矩形	406	15.2.3 CopyFiles 和 AddReg 等安装 过程	468
13.6.3 椭圆弓形和椭圆扇形	411	15.2.4 源路径和目的路径	469
13.6.4 多边形	411	15.2.5 字符串表	469
13.6.5 RECT 结构及对 RECT 的操作	412	15.3 安装程序 setup.exe 的编号	469
13.7 位图操作	414	15.4 使用 msi 文件进行安装	472
13.7.1 截取屏幕、保存位图文件	414	15.4.1 Windows Installer Service	472
13.7.2 将位图显示在界面上	419	15.4.2 msi 文件的创建与修改工具 orca.exe	474
13.8 区域 (Regions)、路径 (Paths) 与修剪 (Clip) 操作	422	15.4.3 准备工作	475
13.8.1 区域的创建及形状、位置等 属性	422	15.4.4 编辑表组	475
13.8.2 区域边沿、区域填充、反转与 勾勒操作	423	第 16 章 设备驱动管理与内核通信	476
13.8.3 组合、比较、移动等操作	426	16.1 设备管理	476
13.8.4 点击测试 (Hit Testing)	427	16.1.1 列举设备接口	477
13.8.5 路径的创建与操作	431	16.1.2 监控设备的加载和卸载	483
13.8.6 路径转换为区域	432	16.2 I/O 控制、内核通信	488
13.8.7 使用区域和路径进行修剪操作, 限制输出	432	16.2.1 加载驱动程序	488
13.9 坐标变换	438	16.2.2 控制驱动程序、与驱动程序进行 通信	495
13.9.1 缩放	439	16.3 编写设备驱动程序	498
13.9.2 旋转	440	16.3.1 驱动程序开发包: DDK	499
13.10 调色板	440	16.3.2 开发驱动程序	499
第 14 章 网络通信与配置	443	16.4 I/O 模式, 同步与异步	504
14.1 Socket 通信	444	第 17 章 用户、认证和对象安全	506
14.1.1 客户端	444	17.1 基本概念	506
14.1.2 服务端	449	17.1.1 访问令牌、权限和用户标识	506
14.1.3 处理并发的客户端连接	455	17.1.2 进程的系统操作权限	507
14.1.4 网络通信的异步 I/O 模式	456		

17.1.3 安全对象	508	样例代码	534
17.1.4 访问控制列表 (ACL)	508	18.3.1 SDK 文档和 MSDN	534
17.2 安全机制程序示例	509	18.3.2 SDK 示例代码	535
17.2.1 列举进程访问令牌内容和权限	509	18.4 x86 平台程序函数调用原理	535
17.2.2 修改进程的权限	514	18.4.1 函数调用的真实过程	535
17.2.3 列举安全对象的安全描述符	515	18.4.2 函数调用约定	539
17.2.4 修改安全描述符	521	18.4.3 为什么通过参数返回数据时只能 使用指针	540
17.3 用户	522	18.4.4 缓冲区溢出	540
17.3.1 创建用户	522	18.4.5 程序运行错误的调试技巧	540
17.3.2 用户组	523	18.5 可执行程序结构与 API 函数接口内部 机理	541
17.3.3 删除用户	525	18.5.1 Windows 可执行程序结构	541
17.3.4 列举用户和用户组、获取用户 信息	525	18.5.2 导入表、导出表、动态链接	543
第 18 章 Windows API 的内部原理	532	18.5.3 NTDLL.DLL、NATIVE API 和 SSDT	544
18.1 关于 API 的补充说明	532	18.5.4 API HOOK	546
18.1.1 Windows API 的版本演进和 Vista 新增 API	532	18.6 发布程序	546
18.1.2 64 位操作系统的接口	533	18.6.1 合理选择编译链接选项	546
18.2 Windows 系统中的对象封装	533	18.6.2 构建到指定路径	546
18.2.1 什么是对象	534	18.7 模块化, 向 Windows API 学习接口 定义	547
18.2.2 面向对象的思想	534	18.7.1 lib 文件	547
18.2.3 Windows 系统中的对象: 内核对象、 GDI 对象等	534	18.7.2 头文件	547
18.3 Windows 程序设计参考: 文档资源与		18.7.3 为第三方应用软件提供 SDK	547

第 1 章

Windows 应用程序开发入门

如何开始 Windows 应用程序开发的学习呢？本书将从一个最简单的 Windows 应用程序入手，讲解 Windows 应用程序设计中的基本概念、编程工具的使用安装与使用方法。通过本章的学习，读者将会对 Windows 应用程序的设计有一个初步的、直观的认识。

1.1 第一个实例程序

本节将给出一个完整的实例程序，并对该程序源代码进行分析。通过本节的学习，读者可以了解 Windows 应用程序源代码的必要组成部分以及它们的组织结构。

1.1.1 start.exe

start.exe 是本书的第一个实例，创建一个简单的消息对话框。

实例 1-1 第一个 Windows 应用程序 start.exe

打开本书配套光盘，在第一章实例代码目录中找到可执行文件 start.exe，运行 start.exe，会弹出如图 1-1 所示的消息对话框。

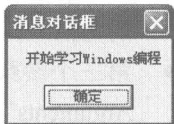


图 1-1 第一个实例程序

这就是一个最简单的 Windows 应用程序实例运行的效果，实现代码如下：

```
/* 头文件 */
#include <windows.h>
//连接时使用 User32.lib
#pragma comment (lib, "User32.lib")
/* *****
 * WinMain
 * 功能: Windows 应用程序示例
 * *****/
int WinMain(
    HINSTANCE hInstance,
    HINSTANCE hPrevInstance,
    LPSTR lpCmdLine,
    int nCmdShow
)
```

```
{  
    // 调用 API 函数 MessageBox  
    MessageBox(NULL,  
        TEXT("开始学习 Windows 编程"),  
        TEXT("消息对话框"),  
        MB_OK);  
    return 0;  
}
```

1.1.2 Windows API

实例 1-1 中，程序最核心的功能实现使用了 Windows API 的 MessageBox 函数。

就像 C 语言有一系列库函数一样，任何一种程序开发平台都会提供众多的接口函数供开发人员使用。Windows 应用程序的开发核心问题是如何使用 Windows SDK 所提供的 API。

所谓 API 是“Application Program Interface”的简写，中文含义为“应用程序接口”，它是一系列函数、宏、数据类型、数据结构的集合。运行于 Windows 系统的应用程序可以使用这些操作系统提供接口来完成应用程序需要的功能。

MessageBox 函数是 Windows 众多 API 中的一个，其的功能是弹出一个对话框。

1.1.3 程序入口函数

WinMain 函数是程序入口点，相当于 C 语言的 main 函数，其定义如下：

```
int WinMain(  
    HINSTANCE hInstance,  
    HINSTANCE hPrevInstance,  
    LPSTR lpCmdLine,  
    int nCmdShow  
);
```

WinMain 函数有 4 个参数：hInstance、hPrevInstance、lpCmdLine、nCmdShow。

- 第一个参数是 hInstance，数据类型是 HINSTANCE，此参数表示应用程序本次运行实例的句柄。
- 第二个参数是 hPrevInstance，数据类型也是 HINSTANCE，表示应用程序之前运行实例的句柄，但是在实际应用中，此参数始终为 NULL。
- 第三个参数 lpCmdLine 是运行时参数。如在 cmd 命令行中运行“start.exe Command Arguments”，那么 lpCmdLine 就指向字符串“Command Arguments”，作用与 C 语言中 main 函数的参数作用类似。数据类型 LPSTR 是 Windows API 中常用的字符串类型。
- 最后一个参数是 nCmdShow，数据类型是 int，表示应用程序窗口（如果有）的显示状态。

WinMain 函数的返回类型是 int 型的。在实例 1-1 中，使用了如下语句来返回。

```
return 0;
```

1.1.4 start.c 代码分析

现在分析 1.1.1 小节给出的实例 1-1 程序的源代码 start.c。

在代码的最开始是源代码文件信息的注释。

之后是预编译声明，在本实例中共有两条预编译声明。

```
#include <windows.h>
```

这一行代码指明了包括 Windows.h。Windows.h 是 Windows 应用程序开发中常用的头文件，在 Windows 应用程序开发中所使用的很多的数据类型、结构、API 接口函数都在 Windows.h 或 Windows.h 所包含的其他头文件中进行了声明，比如实例 1-1 所使用的 MessageBox 函数。如果不

声明包括 `Windows.h`，那么编译器将不能识别 `MessageBox`。

```
#pragma comment (lib, "User32.lib")
```

这行代码指明将源文件编译生成目标文件（.obj）后，将目标文件链接成可执行文件的过程中需要使用到 `User32.lib`，因为 `User32.lib` 中包含了 `MessageBox` 的链接信息。如果在链接程序时，调用链接器的参数中指明了链接时需要使用到 `user32.lib`，这一句程序代码可以省略（参见第 3 章）。

代码接下来是 `WinMain` 函数的定义。在 `WinMain` 函数中调用了 `MessageBox` 函数。

```
MessageBox(NULL,  
            TEXT("开始学习 Windows 编程"),  
            TEXT("消息对话框"),  
            MB_OK);
```

`MessageBox` 函数的原型如下：

```
int MessageBox(  
    HWND hWnd,  
    LPCTSTR lpText,  
    LPCTSTR lpCaption,  
    UINT uType  
);
```

第一个参数 `hWnd` 是消息框所属的窗口的句柄，可以设置为 `NULL`。

第二个参数 `lpText` 是字符串，表示消息框所显示的消息。

第三个参数 `lpCaption` 是字符串，表示消息框的标题。

第四个参数 `uType` 是消息框的类型，在实例 1-1 中设置为“`MB_OK`”，意思是消息框包含一个“确定”按钮。

在 `MessageBox` 函数调用之后，程序返回 0，结束 `WinMain` 函数，退出进程。

1.2 编译代码

本书使用 `Visual Studio` 和 `Platform SDK` 作为程序编译和连接的工具。各版本的 `Visual Studio`，包括 `Visual Studio 6.0`、`Visual Studio 2003`、`Visual Studio 2005`、`Visual Studio 2008` 都可以用于编译本书中的实例程序。

过去已经发行了的所有 `Windows` 版本，每一个版本都有一个 `Platform SDK`。`Windows` 系统为应用程序提供了很多的调用接口，如果要使用这些调用接口就需要用于 `Platform SDK`。`Platform SDK` 提供了开发 `Windows` 应用程序所必须的头文件、库文件等。`Windows` 应用程序的开发接口是 `C/C++` 语言形式的。读者可以使用 `Platform SDK for Windows XP SP2`、`Platform SDK for Windows Server 2003`、`Platform SDK for Windows Vista`。

`Visual C++ 2005 Express Edition`（`Visual C++` 速成版）是 `Visual Studio 2005` 的一个子版本。包括了编译和连接的全部工具 `VC\VC++8.0`，对编译本书的示例程序已经足够使用。

微软公司提供官方下载，下载地址为 <http://msdn2.microsoft.com/zh-cn/express/default.aspx>。各个版本的 `Platform SDK` 都可以从微软公司的网站上下载到。

在本书中，将综合使用 `Visual C++ Express Edition 2005`、`Visual C++ Express Editon 2008` 以及 `Microsoft Platform SDK XP SP2` 版本、`2003SP1` 版本和 `Vista` 版本。

1.2.1 安装 Visual Studio

`Visual Studio` 包括速成版、专业版和团队系统版本（`Visual Studio Team System`）。速成版是免费的版本，专业版和团队系统版本是商业版。但是速成版的功能与专业版的功能并没有太大的区别。

如果读者使用 Visual C++速成版,需单独安装 Microsoft Platform SDK,并将其与 Visual Studio 集成在一起,才可以在本机进行 Windows 应用程序的开发。

Visual C++速成版的安装步骤如下。

(1) 从微软公司的站点上下载中文版的 Visual C++速成版,也可以从本书配套光盘中获得。运行 vcsetup.exe。

(2) 单击“下一步”按钮。

在这一步,读者需要选择需要安装的组件。推荐安装图形化的 IDE,IDE 将大大方便代码的编辑、编译等工作。不安装图形化 IDE,读者也可以使用命令行工具来编译程序。

(3) 在安装的过程中,需要连接上 Internet。安装程序将下载 Windows Installer 3.1、NET Framework 和 Visual C++ 2005 速成版进行安装。

安装完成后,进行注册就可以使用了。在 Visual Studio 安装目录下的 bin 文件夹中,读者可以看到包括 cl.exe、nmake.exe、link.exe、lib.exe、rc.exe 等编译和链接工具。

安装了 Visual C++速成版后仅仅可以进行标准 C 程序和 C++程序的开发,但是还不能调用 Windows API,无法完成一个 Windows 应用程序所需的大部分功能,因为缺少 Platform SDK。

1.2.2 安装 Microsoft Platform SDK

SDK 是 Software Development Kit 的缩写,即“软件开发工具包”。Microsoft Platform SDK 是进行 Windows 应用程序的开发包。

每一个 Windows 发行版本都会有对应的 Platform SDK。SDK 中会使用与这个版本相一致的的头文件和库文件等。每一个版本 Windows 中新增加的 API、数据结构中也会在其中体现。

下面以 Platform SDK for Windows Server 2003 R2 为例说明,其他版本的安装方法类似。如图 1-2 所示。

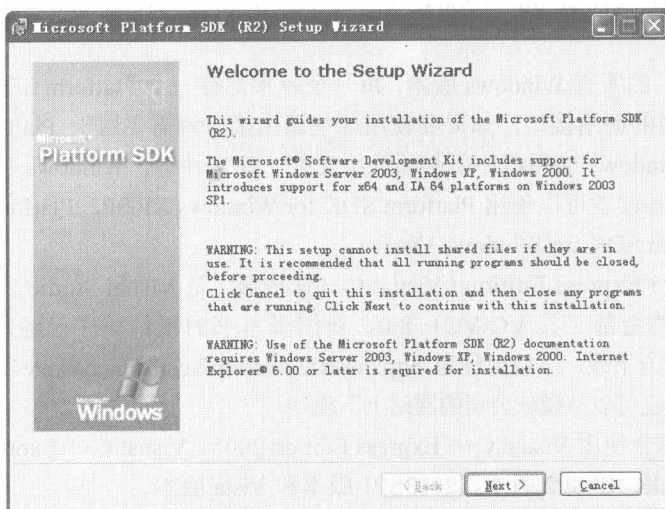


图 1-2 Platform SDK 安装界面

单击“Next”按钮,选择安装选项。

在一般情况下,选择典型安装可以满足使用需要。这里需要进行一些设置,所以选择

“Custom”，如图 1-3 所示。

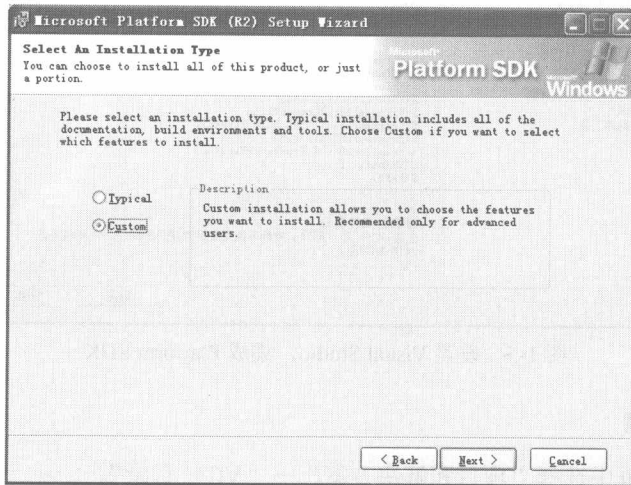


图 1-3 选择安装类型

在出现如图 1-4 所示的安装选择时，选择“Configuration Options”→“Register Environment Variables”选项，并选择完全安装。如果不选择此项，Visual Studio 工具将无法找到 SDK。

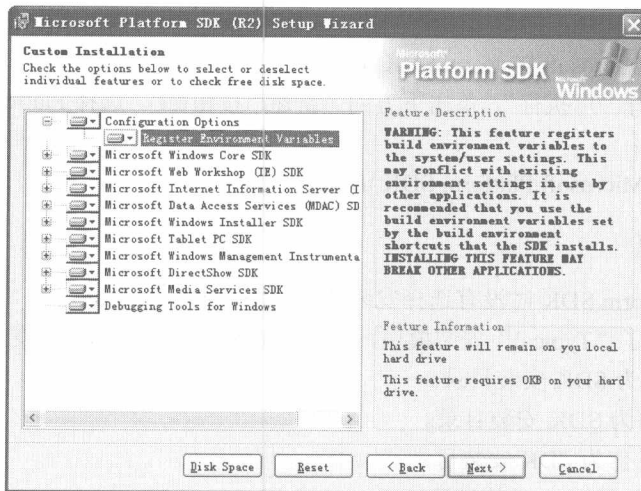


图 1-4 选择安装 Platform SDK 的功能

在安装完成后，可以从 Platform SDK 安装目录中找到若干头文件和库文件，如 Windows.h、Kernel32.lib 等文件，这些就是我们进行 Windows 应用程序开发所必不可少的支持。

1.2.3 集成 Microsoft Platform SDK 与 Visual C++速成版

在安装完成后，还需要对 Visual C++速成版进行设置，使编译链接工具可以找到 SDK。如果编译链接工具找不到 SDK，那么在源代码中的 API 调用将会产生编译链接错误。

在 Visual C++速成版中进行如下操作。在菜单中选择“工具”→“选项”，出现“选项”对话框，如图 1-5 所示。