

Windows NT Win32

软件开发使用详解



吴 华 岳晋生 等编



电子工业出版社

0520/0530/0540 486 系列微机
程序设计/应用开发工作手册

Windows NT Win 32 软件开发使用详解

吴 华 岳晋生 等编

刘德贵 审

电子工业出版社

(京)新登字 055 号

内 容 简 介

Win 32 是在 16 位和 32 位的 Windows 基础上,增加了多进程、多线程、异步 I/O、内存映射文件、结构化异常处理等多种功能的 32 位 Windows 的开发应用环境。

本书全面系统地介绍了 Windows NT 的基本结构,Win 32 的应用编程接口(API)的结构、编程方法和应用系统开发等有关的详细内容。

全书由四部分组成。第一部分是系统服务,共有 18 章,详细讲述了 NT 的全新结构和操作系统内核、内存、进程、线程、安全和对象等管理技术以及包括文件、设备驱动、网络驱动在内的 I/O 技术。第二部分是窗口管理,共有 7 章,详细讲述了 NT 崭新的用户窗口环境,包括 Windows、窗口、会话和输入技术等用户的工作界面。第三部分是图形设备接口,共 9 章,讲述了有关图形、图示、彩色、字型和空间定位等技术。第四部分是多媒体服务,共有 7 章,讲述媒体控制口、声音处理和多媒体文件 I/O 等技术。最后还有虚键代码和错误代码的两个附录及词汇表。

该书内容新颖、技术全面,是进行 Windows 或 Windows NT 软件开发和应用人员的一部很有价值的工具书,是软件专业人员和有关大专院校师生从事 Windows 开发的参考书。

参加本书编写工作的还有,张焕新、舒志勇、李维、李萱、赵先尚、方金华、邓芳群、张磊、徐燕虹、张蔚、张光进、李峻、李青、郑江。

Windows NT Win 32 软件开发使用详解

吴 华 岳晋生 等编

刘德贵 审

责任编辑 路 石

特约编辑 王子恢

电子工业出版社出版(北京市万寿路)

电子工业出版社发行 各地新华书店经销

河北省望都县印刷厂印刷

开本: 887×1092 毫米 1/16 印张: 37 字数: 924 千字

1995 年 7 月第一版 1995 年 7 月第一次印刷

印数: 1—5000 册 定价: 38.00 元

ISBN 7-5053-2109-9/TP·555

序 言

1993年5月Microsoft公司向世界正式公布了其最新一代的32位操作系统——Windows NT。其轰动效应在很大程度上是因为Microsoft公司在DOS和Windows上取得了巨大的成功。Windows NT是一个全新的操作系统,它采用了操作系统的最新技术,提供了现代操作系统的几乎所有功能:多CPU处理、多任务能力、虚拟资源管理、内置网络功能等,与早期Windows的应用范围、实现技术有根本差别,早期Windows只是面向个人的应用系统,而NT则面向大型的应用系统。

Windows NT与UNIX一样都是独立于硬件平台的,它是Microsoft公司第一次走出Intel硬件平台的通用型系统产品。Windows NT目前可以在Intel386/486、DEC Alpha、SGI MIPS等硬件平台上运行。

Windows NT的显著技术特点有:

- 对称多处理结构,可将应用程序或系统任务分成若干线程同时运行在多个处理器上,大大提高了运行效率。可充分发挥多处理机的优势。
- 32位寻址系统,系统能提供连续直接32位寻址。同时采用虚存技术,可支持2G字节虚存空间。
- 先占多任务管理,使操作系统对处理器有优先控制权。提供任务优先级控制,提高了系统响应速度和运行保障能力。
- 内置网络功能,包括:文件共享、打印机共享、交换电子邮件、进度计划管理,并支持远程过程调用。
- 多媒体技术,包括多媒体控制接口、多媒体文件I/O、声音处理和连接技术。
- 易于移植,采用多种措施确保了操作系统的可移植性。使用可移植的C和C++编程;采用隔离处理器(数据结构和寄存器)和硬件平台相关代码动态库的方式,以最少量重编代码,实现了整个操作系统在不同硬件平台上的跨越。

本书根据Microsoft公司为Windows系列提供的Win 32 API操作系统,对有关多线程进程、同步、安全性、I/O及对象管理等先进的操作系统能力,具体介绍系统服务、窗口管理、图形设备接口、多媒体服务等先进技术。

MS-DOS Windows和Windows NT都支持Win 32,在Win 32 API上编写的应用程序在Windows NT和MS-DOS Windows上同样运行。所有Win 32的特性都将得到MS-DOS Windows和Windows NT的支持。Win 32程序在源代码级上在x86和MIPS处理器之间兼容。

使用Win 32的API技术,开发Windows NT的软件和应用系统,是国内外软件公司和开发部门九十年代面临的重大挑战,决心在此平台工作的一切开发和应用部门都应掌握并使用新的API环境来构造大型应用系统,这就是本书的最终宗旨。

编者

一九九四年十二月

Win 32 API 介绍

概述

Windows 32 位 API 集是一个可移植的、支持 32 位连续地址空间的 API 集。这个 API 集开始是用于 Windows NT 的,过一段时间后可以用于 MS-DOS。

Win 32 严格区分 32 位和 16 位应用程序,两种方式混合是不允许的(即,32 位应用程序,只能使用 Win 32 函数而不能调用 16 位 DLL 中的函数)。这种设计目的是为了鼓励移植。Win 32 的可移植目标很简单:调用 Win 32 API 集的应用程序,要在 MIPS 的 Windows NT 上运行,只需重新编译即可。

限制和目标

Win 32 有几个明显的目标和限制:

- 它的 API 集与现在的 Windows 3.0 API 集是向上兼容的,只在绝对需要改动的地方才对函数进行了变动。
- 把已有的 16 位 Windows 3.0 应用程序移植到 Win 32 上是简单而直接的。
- x86 上的 Win 32 和 MIPS 上的 Win 32 在源代码级上绝对兼容。DOS 上的 Win 32 与 Windows NT 上的 Win 32 在二进制代码级是绝对兼容的。
- 进程分别在不同的地址空间运行。
- 在一个进程中,可以有多个并行线程执行。调度是可剥夺的。API 是本着“对称多处理 (symmetric multiprocessing)”思想设计的。
- 内存访问机制是:32 位连续地址访问。
- Win 32 API 集可在各种平台间移植。

Win 32 API 设计

Win 32 API 集是本着可移植性和安全性的思想设计的。另外,影响 Win 32 API 集的主要因素是,保持已有的 Windows API 集的功能、网络和界面。凡是有可能的,都把已有的函数扩展为 32 位,并且按 Win 32 API 进行归类。如果某函数出现了安全性和可移植性的问题,则增加一个新的 Win 32 函数,提供与已有的函数类似的功能,消除了安全性和可移植性的问题。

WJS 159/03

目 录

第 1 章 系统服务介绍	1	2.1 关于内存	19
1.1 概述	1	2.1.1 Global 和 Local 函数 ...	19
1.2 Win 32 扩展	1	2.1.2 标准 C 库	20
1.2.1 多线程进程结构	1	2.1.3 堆栈函数	20
1.2.2 连续地址空间	1	2.1.4 虚拟函数	20
1.2.3 可移值的 32 位函数	2	2.1.5 关于共享内存	20
1.3 基本功能概述	2	2.1.6 使用虚拟函数	21
1.3.1 原子管理函数	2	2.1.7 为共享内存使用文件映射	23
1.3.2 通信设备函数	2	2.1.8 使用堆栈函数	25
1.3.3 字符方式应用程序的支持	3	2.2 小结	25
1.3.4 调试工具的支持	4	2.2.1 虚拟函数	25
1.3.5 对动态连接库的支持	4	2.2.2 堆栈函数	26
1.3.6 环境	5	2.2.3 文件映射函数	26
1.3.7 事件日志	5	第 3 章 进程和线程	27
1.3.8 异常处理	5	3.1 关于多任务	27
1.3.9 文件 I/O 函数	6	3.1.1 进程和线程	27
1.3.10 句柄/对象管理	7	3.1.2 调度	28
1.3.11 映射文件	8	3.2 使用线程	29
1.3.12 初始文件函数	8	3.2.1 创建线程	29
1.3.13 安装/压缩	9	3.2.2 挂起线程的执行	30
1.3.14 内存管理函数	9	3.2.3 多线程的同步执行	31
1.3.15 模块管理函数	11	3.2.4 终止线程	31
1.3.16 管道	12	3.2.5 多线程和 GDI 对象	32
1.3.17 进程和线程	12	3.2.6 实例:多线程 MDI 应用程序	32
1.3.18 注册	13	3.3 创建子进程	38
1.3.19 资源管理函数	13	3.3.1 命令行参数	38
1.3.20 安全性	14	3.3.2 创建标志	38
1.3.21 段管理函数	16	3.3.3 继承性	39
1.3.22 声音	16	3.3.4 环境变量	40
1.3.23 同步	16	3.3.5 启动信息	41
1.3.24 时间	17	3.3.6 实例:有重定向 I/O 的 子进程	42
1.3.25 局部线程的存储空间	17	3.4 控制进程和线程的优先权	46
1.3.26 版本管理	18		
第 2 章 内存管理	19		

3.4.1	进程函数	47	7.3.4	创建 DDE 存取	81
3.4.2	线程函数	48	7.3.5	创建 DDE 对话	82
3.4.3	调试函数	48	7.4	打开对象	82
3.4.4	STARTUPINFO 标志	48	7.5	存取对象	82
3.4.5	创建标志	48	7.6	关闭对象	83
3.4.6	线程优先级常数	49	7.7	窗口类	83
3.4.7	进程优先级常数	49	7.8	SM_SECURE 系统变量	83
3.4.8	结构	49	7.9	ES_PASSWORD 风格	
第 4 章	同步	50		编辑控制	83
4.1	介绍	50	7.10	枚举窗口	83
4.1.1	关于同步对象	50	7.11	存取屏幕内容	84
4.1.2	进程间同步	50	7.12	存取剪贴板	84
4.1.3	单个进程的同步线程	51	7.13	DDE 安全性	84
第 5 章	文件 I/O	63	7.13.1	注册	84
5.1	关于文件	63	7.13.2	对话	84
5.1.1	系统组织	63	7.14	审核	85
5.1.2	文件操作	64	7.15	服务器初始化	85
5.2	使用文件	67	7.16	客户机初始化	85
5.2.1	实例:建立和打开文件	67	7.16.1	服务器连接	85
5.2.2	实例:文件读、写和加锁	68	7.16.2	窗口工作站存取	86
5.2.3	实例:建立和使用临时性文件	69	7.16.3	线程初始化	86
5.2.4	实例:查找文件和改变文件属性	71	7.17	注册处理	86
5.3	函数	72	7.17.1	注册	86
第 6 章	文件映射	74	7.17.2	注销	86
6.1	关于文件映射	74	7.17.3	窗口工作站加锁	87
6.1.1	共享文件	74	7.18	关闭系统	87
6.1.2	共享内存块	75	第 8 章	动态连接库	88
6.1.3	关闭文件视图	75	8.1	关于动态连接库	88
6.2	使用文件映射	75	8.1.1	动态连接的优点	88
6.3	函数	76	8.1.2	动态连接中使用的文件	89
第 7 章	安全性	77	8.1.3	调用 DLL 函数	90
7.1	目标	77	8.1.4	DLL 数据作用范围:多进程、进程和线程	91
7.2	USER 32 安全对象	77	8.2	使用动态连接库	92
7.2.1	对象存取类型	78	8.2.1	建立一个简单的动态连接库	92
7.3	对象句柄	79	8.2.2	使用装入时动态连接	93
7.3.1	创建桌面	79	8.2.3	使用运行时动态连接	93
7.3.2	创建窗口	80			
7.3.3	创建菜单	81			

8.2.4	函数 LibMain	94
8.3	函数	95
第9章	资源	96
9.1	关于资源	96
9.1.1	寻找和装入资源	96
9.1.2	增加、删除和替换资源	97
9.1.3	枚举资源	97
9.2	使用资源	98
9.2.1	更新资源	98
9.2.2	建立资源列表	99
9.3	函数	103
第10章	管道	104
10.1	关于管道	104
10.1.1	无名管道	104
10.1.2	命名管道	104
10.2	使用管道	108
10.2.1	使用无名管道	108
10.2.2	使用命名管道	109
10.2.3	命名管道的交互操作	120
10.3	函数	120
第11章	邮件槽	122
11.1	关于邮件槽	122
11.1.1	邮件槽的名字	122
11.1.2	邮件槽与 LAN Manager	123
11.1.3	服务器与客户函数	123
11.2	使用邮件槽	124
11.2.1	创建邮件槽	124
11.2.2	写邮件槽	125
11.2.3	读邮件槽	125
11.3	函数	127
第12章	网络	128
12.1	关于网络	128
12.1.1	不依赖于网络的函数	128
12.1.2	Netbios 函数	129
12.2	网络的使用	129
12.2.1	使用连接对话框	130
12.2.2	枚举网络资源	130
12.2.3	增加网络连接	132
12.2.4	恢复连接名	132

12.2.5	恢复用户名	133
12.2.6	取消网络连接	133
12.2.7	恢复网络错误信息	134
12.3	函数	135
第13章	字符模式支持	136
13.1	引言	136
13.2	关于控制台	136
13.2.1	File 函数的输入输出	137
13.2.2	使用 Console 函数进行 输入输出	137
13.3	控制台的使用	137
13.3.1	获取控制台输入输出 句柄	138
13.3.2	控制台模式	139
13.3.3	File 函数的模式	139
13.3.4	Console 函数的模式	140
13.3.5	通过 File 函数使用 控制台	141
13.3.6	移动与隐去光标	143
13.3.7	用 Console 函数进行 输入	143
13.3.8	输入缓冲区	143
13.3.9	键盘事件	144
13.3.10	鼠标事件	144
13.3.11	缓冲区大小改变事件	145
13.3.12	使用 Console 函数的 输出	146
13.3.13	窗口和显示缓冲区的 大小	147
13.3.14	在显示缓存区中写 字符串	147
13.3.15	在显示缓存区中写 颜色属性	148
13.3.16	写字符和属性块	148
13.3.17	读显示缓冲区	149
13.3.18	滚动显示缓冲区	150
13.3.19	控制句柄函数	152
13.4	函数	153
第14章	服务控制管理程序	155
14.1	目标与局限	155

14.2	服务的类型	155	17.1.2	异常的结构	174
14.2.1	Win 32 服务	155	17.1.3	异常处理的语法	176
14.2.2	Drvrvr 服务	156	第 18 章 通信		179
14.3	Win 32 服务的用户帐号	156	18.1	关于通信	179
14.4	配置登记簿	157	18.2	Comm 函数的使用	179
14.5	Service Control Manager 的客户	157	18.2.1	获取通信资源的句柄	179
14.6	Service Control Manager 函数	158	18.2.2	初始化通信资源	179
14.7	SCManager 对象	159	18.2.3	配置通信资源	179
14.7.1	SCManager 访问类型	159	18.2.4	读通信资源	181
14.8	服务对象	160	18.2.5	写通信资源	182
14.8.1	服务访问类型	160	18.2.6	其它通信函数	182
14.8.2	服务控制程序的数据类型	162	18.3	函数	184
第 15 章 时间		163	第 19 章 窗口管理介绍		185
15.1	关于日期和时间函数	163	19.1	概述	185
15.1.1	MS-DOS 时间	163	19.1.1	Win 3 和 PM 的输入处理问题	185
15.1.2	Windows 时间	163	19.1.2	Win 32 输入处理的目标	185
15.2	日期和时间函数的使用	164	19.1.3	多线程运行	185
15.2.1	文件时间	164	19.1.4	输入管道	186
15.2.2	实例:显示文件创建时间的文件目录	164	19.1.5	局部输入状态	186
15.2.3	实例:改文件时间为当前时间	165	19.1.6	受局部输入状态影响的函数	187
15.2.4	系统时间	165	19.1.7	Z-序的相关活动	187
15.2.5	实例:设置系统时间	165	19.1.8	擦掉和重画挂起应用程序的窗口框架或图标	188
15.2.6	函数	167	19.1.9	图标语言支持(NLS)	188
第 16 章 调试		168	19.1.10	单字编码	188
16.1	关于调试支撑及性能监视	168	19.1.11	通知消息	189
16.1.1	其它 Win32 函数的调试支撑	168	19.1.12	热键	190
16.1.2	调试函数	169	19.1.13	窗口挂钩变化	190
16.1.3	性能监视函数	171	19.1.14	各种 USER 32 函数的变化	191
16.2	调试支持功能的使用	172	19.2	USER 功能概述	196
16.3	函数	173	19.2.1	加速键	196
第 17 章 结构化的异常处理		174	19.2.2	记号	196
17.1	关于结构化的异常处理	174	19.2.3	裁剪板	196
17.1.1	目标	174	19.2.4	普通对话框	197

19.2.5	光标	197
19.2.6	DDE 管理库	198
19.2.7	对话框	199
19.2.8	拖拉/放下	200
19.2.9	错误	200
19.2.10	挂钩	200
19.2.11	图标	201
19.2.12	键盘	201
19.2.13	多文档界面	201
19.2.14	菜单	202
19.2.15	消息	203
19.2.16	鼠标	204
19.2.17	对象连接和嵌入	204
19.2.18	刷新	206
19.2.19	特征	206
19.2.20	矩形	206
19.2.21	滚动	207
19.2.22	字符串	207
19.2.23	窗口管理	208
第 20 章	消息和消息队列	211
20.1	关于消息和消息队列	211
20.1.1	消息	211
20.1.2	排队和非排队的消息	211
20.1.3	消息处理	212
20.1.4	邮寄和发送消息	214
20.1.5	消息类型	214
20.1.6	消息过滤	215
20.1.7	消息死锁	215
20.2	使用消息和消息队列	216
20.2.1	创建消息循环	216
20.2.2	检查消息队列	218
20.2.3	邮寄消息	219
20.2.4	发送消息	219
20.3	函数	221
第 21 章	键盘输入	222
21.1	关于键盘输入	222
21.1.1	键盘输入模式	222
21.1.2	键盘焦点和激活	222
21.1.3	击键消息	223
21.1.4	字符消息	224
21.1.5	键状态	225
21.1.6	击键和字符翻译	226
21.1.7	热键支持	226
21.2	使用键盘输入	226
21.2.1	处理击键消息	226
21.2.2	翻译字符消息	228
21.2.3	处理字符消息	228
21.2.4	使用标记	229
21.2.5	实例:显示键盘输入	229
21.3	函数和消息	235
第 22 章	鼠标输入	237
22.1	关于鼠标输入	237
22.1.1	鼠标光标	237
22.1.2	鼠标捕获	237
22.1.3	鼠标配置	237
22.1.4	鼠标消息	238
22.1.5	WM_NCHITTEST 消息	240
22.1.6	窗口活动	241
22.2	使用鼠标输入	241
22.2.1	跟踪鼠标光标	241
22.2.2	实例:用鼠标画线	241
22.2.3	处理连续两次按下鼠标的消息	244
22.2.4	实例:选择一行文本	245
22.3	鼠标输入函数和消息	247
第 23 章	滚动条	249
23.1	有关滚动条的描述	249
23.1.1	滚动条的组成	249
23.1.2	标准滚动条和滚动条控制	250
23.1.3	滚动框的位置和滚动范围	251
23.1.4	滚动条请求	251
23.1.5	滚动条的键盘界面	253
23.1.6	滚动客户区域	254
23.1.7	滚动条的可见性	254
23.1.8	滚动条的颜色和度量	255
23.2	使用滚动条	255

23.2.1	建立滚动条	255
23.2.2	滚动文本	256
23.2.3	滚动位图	261
23.2.4	建立标准滚动条的 键盘界面	269
23.3	函数	270
23.4	消息	271
第24章	菜单	272
24.1	有关菜单的描述	272
24.1.1	菜单条和弹出式菜单	272
24.1.2	菜单句柄	273
24.1.3	类菜单	273
24.1.4	菜单项	273
24.1.5	用键盘访问菜单	277
24.1.6	菜单的建立	278
24.1.7	菜单的显示	278
24.1.8	撤消菜单	279
24.1.9	菜单消息	279
24.1.10	菜单的修改	279
24.2	怎样使用菜单	280
24.2.1	使用菜单样板资源	280
24.2.2	建立浮动弹出式菜单	284
24.2.3	使用菜单项位图	286
24.2.4	建立用户自己绘制的 菜单项	291
24.2.5	使用自己的选中标志 位图	296
24.3	函数和消息	304
第25章	键盘加速键	305
25.1	关于键盘加速表	305
25.1.1	加速键表	305
25.1.2	加速键的创建	306
25.1.3	加速键的分配	306
25.1.4	加速键和菜单	307
25.2	使用键盘加速键	308
25.2.1	使用加速键表资源	308
25.2.2	使用运行时创建的 加速键表	313
25.3	函数	320
25.4	消息	320

第26章	图形设备接口	321
26.1	简介	321
26.2	GDI的增强	321
26.3	GDI的改进	321
26.3.1	调用的改进	322
26.3.2	函数改进	322
26.3.3	改进的详细描述	322
26.3.4	新的函数调用	324
26.4	GDI函数概述	325
26.4.1	位图	326
26.4.2	绘图工具	326
26.4.3	剪裁	327
26.4.4	颜色	327
26.4.5	曲线输出	328
26.4.6	设备环境关系函数	328
26.4.7	字模	328
26.4.8	线输出	329
26.4.9	映射	329
26.4.10	元文件	330
26.4.11	GDI对象	330
26.4.12	调色板的管理	331
26.4.13	路径	332
26.4.14	定位	332
26.4.15	打印	332
26.4.16	区域	333
26.4.17	字符文本输出	334
第27章	位图	335
27.1	位图	335
27.1.1	位图、绘图平面和设备 环境	337
27.1.2	两种类型的位图	337
27.1.3	位图旋转	338
27.1.4	位图缩放	339
27.1.5	位图作为画刷	339
27.1.6	位图存储	340
27.2	位图的使用	342
27.2.1	利用位图来捕捉图象	342
27.2.2	利用位图来缩放图象	343
27.2.3	利用位图来存储图象	344

27.3	函数	347
第 28 章	画刷	349
28.1	画刷	349
28.1.1	实心画刷	350
28.1.2	备用画刷	350
28.1.3	阴影画刷	350
28.1.4	图案画刷	350
28.1.5	画刷起点	351
28.2	画刷的使用	351
28.3	函数	357
第 29 章	画笔	358
29.1	关于画笔	358
29.1.1	装饰画笔	358
29.1.2	几何画笔	359
29.2	如何使用画笔	360
29.2.1	修改选择颜色公共对话框	360
29.3	函数	363
第 30 章	区域	364
30.1	关于区域	364
30.1.1	区域的创建和选择	364
30.1.2	区域的操作	364
30.2	如何使用区域	365
30.2.1	使用区域来剪裁输出	365
30.2.2	使用区域来执行探测	370
30.3	函数	370
第 31 章	直线和曲线	371
31.1	关于直线和曲线	371
31.1.1	直线	371
31.1.2	曲线	372
31.1.3	直线与曲线的组合	373
31.1.4	线的属性	373
31.2	如何使用直线和曲线	374
31.2.1	使用线函数来绘制浮标	374
31.2.2	使用直线及曲线画饼图	376
31.3	有关函数	379
第 32 章	颜色和调色板	380

32.1	关于颜色	380
32.1.1	颜色的物理特性	380
32.1.2	彩色视频显示技术	380
32.1.3	颜色的操作	382
32.2	如何使用颜色	381
32.2.1	创建一个彩色画笔	384
32.3	有关函数	385
32.4	有关消息	385
第 33 章	坐标位置及变换	386
33.1	关于坐标位置及其变换	386
33.1.1	坐标位置	387
33.1.2	变换	387
33.1.3	缺省变换	393
33.2	如何使用坐标位置及其变换	393
33.2.1	使用预定义的单位绘制图形	394
33.2.2	在应用程序使用区内使图形居中	394
33.2.3	图形缩放	394
33.2.4	图形平移	395
33.2.5	图形旋转	395
33.2.6	图形剪切	395
33.2.7	图形镜像	396
33.3	有关函数	396
第 34 章	元文件	397
34.1	关于元文件	397
34.2	增强型格式元文件	397
34.2.1	增强型元文件的记录	397
34.2.2	增强型元文件的操作	398
34.3	Windows 格式元文件	399
34.4	从 Windows 格式到增强型格式的转换	400
34.5	如何使用元文件	400
34.5.1	创建一个增强型的元文件	400
34.5.2	显示图象并将其存储到增强型元文件中	402
34.5.3	打开增强型元文件并显示其内窗	406

34.6	有关函数	408
第 35 章	多媒体服务器	409
35.1	Windows 的多媒体服务器	409
35.2	Windows 多媒体服务器的体系结构	410
35.3	Windows 多媒体设计的要点	410
35.4	构造多媒体应用程序	410
35.5	调试多媒体应用程序	410
第 36 章	媒体控制接口	411
36.1	MCI 概述	411
36.1.1	MCI 的体系结构	411
36.1.2	MCI 的编程接口	411
36.1.3	MCI 命令集	412
36.1.4	MCI 设备	412
36.1.5	打开 MCI 设备	413
36.2	使用命令消息接口	414
36.2.1	关于命令消息	414
36.2.2	MCI 指令消息小结	415
36.2.3	发送指令消息	417
36.2.4	打开设备	417
36.2.5	关闭设备	420
36.2.6	使用等待和通知标志	421
36.2.7	获得 MCI 系统信息	422
36.3	关于指令串接口	423
36.3.1	使用 MeisendString 函数发送指令字符串	423
36.3.2	关于指令串的附加信息	423
第 37 章	音响系统	424
37.1	音响服务器	424
37.1.1	音响服务器的类型	424
37.1.2	音响服务器的级别	424
37.2	Windows 音响结构	425
37.2.1	高级音响函数	425
37.2.2	低级音响函数	425
37.2.3	MIDI 映射器	425
37.3	音响文件格式	425
37.3.1	使用 RIFF 文件	426

37.4	参考文献	426
第 38 章	高级音响	427
38.1	函数前辍	427
38.2	播放波形音响	427
38.2.1	播放波形音响中的限制	427
38.2.2	使用 SndPlaySound 函数	428
38.2.3	播放系统警告音响	430
38.3	使用 MCI 播放和记录音响	430
38.3.1	MCI 音响数据类型	431
38.3.2	MCI 音响命令	431
38.3.3	打开 MCI 音响设备	432
38.3.4	MCI 错误处理	435
38.3.5	启动重播	436
38.3.6	改变当前位置	439
38.3.7	设置时间格式	440
38.3.8	关闭 MCI 音响设备	441
38.3.9	获取设备和媒体的信息	441
38.3.10	波形音响设备的记录	444
38.3.11	使用 MCI MIDI 序列发生器	447
38.4	MIDI 绘图仪	448
38.4.1	MIDI 符号规定	448
38.4.2	MIDI 绘图仪和 Windows	449
38.4.3	MIDI 绘图仪的体系结构	449
38.4.4	通道映射	449
38.4.5	插补映射	449
38.4.6	键映射	450
38.4.7	映射和 MIDI 消息小结	450
38.5	授权的 MIDI 文件	450
38.5.1	关于基础级和扩展级的	

合成器.....	451	39.5.1 MIDI 输出数据类型	475
38.5.2 MIDI 文件的偏写说明	451	39.5.2 查询 MIDI 输出设备 ...	475
.....	451	39.5.3 打开 MIDI 输出设备 ...	476
38.5.3 标准的 MIDI 插补分配	452	39.5.4 发送 MIDI 消息	477
.....	452	39.5.5 发送缓冲区消息	478
38.5.4 标准的 MIDI 键分配 ...	452	39.5.6 用运行状态来发送 MIDI	
38.5.5 使用 MARKMIDI		消息	479
实用程序.....	452	39.5.7 重新设置 MIDI 输出 ...	479
第 39 章 低级音响	453	39.5.8 改变内部 MIDI 合成器	
39.1 函数前缀.....	453	音量	479
39.2 使用低级音响服务.....	453	39.5.9 预装入带内部 MIDI	
39.2.1 询问音响设备	453	合成器的助音器	480
39.2.2 打开及关闭音响设备		39.5.10 使用带低级 MIDI 函数	
.....	455	的 MIDI 绘图仪	480
39.2.3 分配及配备音响数据块		39.6 记录 MIDI 音响	481
.....	455	39.6.1 MIDI 输入数据类型.....	481
39.2.4 管理音响数据块	456	39.6.2 询问 MIDI 输入设备 ...	481
39.2.5 使用 MMTIME 结构		39.6.3 打开 MIDI 输入设备 ...	481
.....	458	39.6.4 管理 MIDI 记录	482
39.2.6 用音响函数来处理错误		39.6.5 接收带时间标记的	
.....	459	MIDI 消息	484
39.3 播放波形音响.....	459	39.6.6 接收运行状态消息 ...	484
39.3.1 波形输出数据类型 ...	459	39.7 辅助音响设备.....	484
39.3.2 查询波形输出设备 ...	460	39.7.1 查询辅助音响设备 ...	484
39.3.3 打开波形输出设备 ...	462	39.7.2 改变辅助音响设备	
39.3.4 指定波形数据格式 ...	462	的音量.....	485
39.3.5 写波形数据	466	39.8 音响裁剪板格式.....	486
39.3.6 获取当前重放位置 ...	470	第 40 章 多媒体计时器	487
39.3.7 停止、暂停及重新启动		40.1 函数前缀.....	487
重放	470	40.2 计时器服务.....	487
39.3.8 关闭波形输出设备 ...	471	40.2.1 计时器数据类型	487
39.3.9 改变波形重放音量 ...	471	40.2.2 使用计时器服务	487
39.3.10 改变音调及重放速率		40.2.3 获取系统时间	488
.....	472	40.2.4 确定最大和最小事件	
39.4 记录波形音响.....	472	周期	488
39.4.1 波形输入数据类型 ...	472	40.2.5 建立最小计时器分辨率	
39.4.2 查询波形输入设备 ...	472	488
39.4.3 打开波形输入设备 ...	373	40.2.6 启动计时器事件	489
39.4.4 管理波形记录	373	40.2.7 中断计时器事件	490
39.5 播放 MIDI 音响	475	40.2.8 使用计时器反调函数	

.....	490
第 41 章 多媒体文件 I/O	492
41.1 多媒体文件 I/O 服务介绍	492
41.1.1 与 MS-DOS、C 运行时库 及 Windows 文件 I/O 进行比较	492
41.1.2 函数前缀	493
41.1.3 数据类型	493
41.2 完成基本文件 I/O	493
41.2.1 打开文件	493
41.2.2 建立及删除文件	495
41.2.3 读写文件	495
41.2.4 在文件中定位新位置	485
41.3 完成带缓冲区的文件 I/O	496
41.3.1 确定何时使用带缓冲区的 文件 I/O	496
41.3.2 为带缓冲区的文件 I/O 打开一个文件	496
41.3.3 I/O 缓冲区控制函数	497
41.4 使用 RIFF 文件	498
41.4.1 RIFF 文件介绍	498
41.4.2 MMCKINFO 结构	499
41.4.3 产生四字符码	499
41.4.4 建立 RIFF Chunk	499
41.4.5 导向 RIFF 文件	500
41.4.6 RIFF 文件 I/O 举例	502
41.5 MMIOINFO 结构	504
41.6 直接访问文件 I/O 缓冲区	504
41.6.1 获取文件 I/O 缓冲区 信息	505
41.6.2 读写文件 I/O 缓冲区	505
41.6.3 改进文件 I/O 缓冲区	505
41.6.4 访问文件 I/O 缓冲区 举例	506

41.6.5 终止文件 I/O 缓冲区的 直接访问	507
41.7 完成内存文件上的文件 I/O	507
41.7.1 打开内存文件	507
41.8 使用常规 I/O 过程	507
41.8.1 用常规 I/O 过程打开 文件	508
41.8.2 写一个 I/O 过程	508
41.8.3 安装一个 I/O 过程	509
41.8.4 与其它应用程序共享 I/O 过程	409
第 42 章 Unicode	510
42.1 总则	510
42.2 Windows 32 API 中的 Unicode 支持	510
42.3 数据类型	510
42.4 函数原型	511
42.5 基本步骤	511
42.6 窗口类	512
42.7 消息	514
42.8 子类产生和自动消息传送	515
42.9 资源	515
42.10 字符串函数	516
42.11 C 运行时库	516
42.12 文件名	517
42.13 特殊字符	518
42.14 Unicode 的普通文本格式	519
42.15 特殊标题	520
42.16 工作环境	520
附录 A 虚拟键码	522
附录 B 错误代码	526
B.1 错误代码的字母顺序表	526
B.2 错误代码值顺序表	539
词汇表	554

第 1 章 系统服务介绍

1.1 概述

本章讲述 Windows 32 位 API 的基本组成部分。

1.2 Win 32 扩展

要成为一个全功能的 32 位操作系统,需要对 Windows 3.0 所提供的基本 API 集进行扩展。扩展的目的是很明确的,它是为了:

- 对于在不同地址空间运行的每个进程,支持其内部的多线程处理。
- 支持连续的 32 位地址空间。
- 对所有可为进程间所共享的对象,解决其安全性问题。
- 支持可移植的 32 位 API 界面。

注意,Win 32 的重点是要扩展现在的 Windows 3.0 技术,以使 Windows 3.0 应用程序很容易移植到 x86 或 MIPS 平台上的 Win 32。

1.2.1 多线程进程结构

Win 32 的一个突出特点是引入了多线程进程(multi-threaded process)结构。一个 Win 32 进程包括:

- 一个受保护的地址空间,可以访问最大达 2GB 的虚拟内存。
- 在进程内部可执行多个线程。每个线程有它自己的堆栈和处理机状态。在运行 Windows NT 的 MP 系统上,多个线程可同时执行,但分别使用不同的处理器。

为了支持 Win 32 的进程结构,必须在 Win 32 集上加上一些小小的扩展。这些扩展可分类如下:

- 对进程以及线程创建、操纵的支持。
- 对一个进程内线程之间的同步和同步对象的支持,同步对象可被多个进程共享,如果它们的进程可访问同一个同步对象,它们就可以同步。
- 一个统一的共享机制,提供安全特性,即对进程间共享对象提出的限制/控制。

1.2.2 连续地址空间

Win 32 的一个最大扩展是引入了连续(flat)32 位内存管理。为弥补 16 位基于用选择器进行内存管理(Local 和 Global 函数)的不足而设计的新函数,可更容易地使用和编写文档,原有的功能也尽可能地保存下来。16 位内存管理的某些特性,在使用 32 位模式时不可移植或不需时,没有保留到 32 位环境中来。

1.2.3 可移植的 32 位函数

扩展的最终一个主要方面是着眼于应用程序的可移植性。这方面有两个主要目标：

- 限制应用程序中面向 x86 结构而在 MIPS 平台上没有的内容(EMS 内存函数…);
- 去掉应用程序对 DOS 函数(int 21h)的依赖。

在所有可能的地方,面向 x86 结构的函数都被去掉或替换了,这类例子包括:

- SwitchStackBack 和 SwitchStackTo
- Catch 和 Throw
- LimitEmsPages

当前的 Windows 版本的最起码要求是,应用程序调用下层的 DOS 服务来执行文件操作。Win 32 API 集经过扩展提供了全部的服务。这有利于应用程序在 DOS 上的 Win 32 和 MIPS 上的 Win 32 之间进行移植。

1.3 基本功能概述

下面给出 Win 32 的基本组成部分,给读者提供关于函数的总览。

1.3.1 原子管理函数

原子表,即哈希(Hash)表,是把字符串表示为 16 位整数值的一个有效方法,其中的全局原子表用于存放 DDE(动态数据交换)原子值。每个应用程序进程能够创建一个或多个只对它们自己可见的自用原子表。

原子函数包括:

函 数	描 述
AddAtom	为字符串创建一个原子。
DeleteAtom	如果引用计数值为 0,就删除原子。
FindAtom	检索与一个字符串相对应的原子。
GetAtomHandle	获得对应于某原子的字符串的句柄(相对于局部堆栈)。
GetAtomName	拷贝与一个原子相对应的字符串。
GlobalAddAtom	为字符串创建全局原子。
GlobalDeleteAtom	如果引用计数值为 0,则删除一个全局原子。
GlobalFindAtom	检索与一个字符串相对应的全局原子。
GlobalGetAtomName	拷贝与一个全局原子相对应的字符串。
InitAtomTable	初始化原子哈希表。

1.3.2 通信设备函数

根据应用程序开发者反馈回来的消息,强化了通信函数。怎样使用通信函数,参见第 18 章“通信”。

通信函数包括: