

<<Windows核心编程(第5版)>>

图书基本信息

书名：<<Windows核心编程(第5版)>>

13位ISBN编号：9787302184003

10位ISBN编号：7302184003

出版时间：2008年9月

出版时间：清华大学出版社

作者：Jeffrey Richter, Christophe Nasarre

页数：770

译者：葛子昂, 周靖, 廖敏

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<Windows核心编程(第5版)>>

前言

Windows世界已经发生了许多变化，WindowsXP、Windows2003、WindowsVista以及WindowsServer2008相继推出。

与之对应的，《Windows核心编程》（第5版）也与时俱进，针对最新的操作系统进行更新，为我们了解并发挥最新的平台特性提供了宝贵的指引。

这是一本Windows系统编程的权威专著，许多年之前我就已经读过它的第4版；这是一本生动的教科书，它帮助我更深入地了解Windows系统；同时它又是一本参考书，在我开发Windows应用程序的过程中遇到问题时，我会到本书中寻找答案。

希望它同样能够为你答疑解惑，并解决你的实际问题。

本书由葛子昂、周靖、廖敏共同翻译：第8~22章由葛子昂翻译，第1~6章由周靖翻译，第23~26章及附录由廖敏翻译，第7章由刘江友情客串翻译，全书由葛子昂最终审阅和统稿。

鉴于时间和精力有限，一些翻译难免存在不足甚至错误之处，为此我建立了一份网上勘误表。

如果读者发现任何错误，都可以通过该网页与我联系，一旦确认，我会立即将其更新到勘误表中。

勘误表的网址为www.gesoftfactory.com/ge/WindowsViaCpp。

最后，感谢Jeffrey和Christophe在本书翻译过程中不厌其烦地解答我的问题，核实并纠正英文版中发现的一些错误。

感谢李劲松、王渊峰、张羿、孙展波、谭映辉以及孙祺对中文版的全部或部分章节进行了审阅并提出宝贵的意见。

感谢潘爱民在百忙中抽空阅读部分译稿，并给以反馈。

感谢妻儿和家人对我的理解和支持，本书的翻译工作占用了本应属于你们的时间，现在终于可以有更多时间来陪伴你们了。

<<Windows核心编程(第5版)>>

内容概要

这是一本经典的Windows核心编程指南，从第1版到第5版，引领着数十万程序员走入Windows开发阵营，培养了大批精英。

作为Windows开发人员的必备参考，本书是为打算理解Windows的C和C++程序员精心设计的。

第5版全面覆盖Windows

XP，Windows Vista和Windows Server

2008中的170个新增函数和Windows特性。

书中还讲解了Windows系统如何使用这些特性，我们开发的应用程序又如何充分使用这些特性，如何自行创建新的特性。

作者简介

Jeffrey

Richter是Wintellect公司的创始人之一，该公司从事培训、调试和咨询，致力于帮助其他公司以更快的速度开发出更优秀的软件。

他著作颇丰，代表作有畅销书CLR

viaC #。

他还是MSDN Magazine的特约编辑和专栏作家。

<<Windows核心编程(第5版)>>

书籍目录

第 部分 必备知识

第1章 错误处理

- 1.1 定义自己的错误代码
- 1.2 ErrorShow示例程序

第2章 字符和字符串处理

- 2.1 字符编码
- 2.2 ANSI字符和Unicode字符与字符串数据类型
- 2.3 Windows中的Unicode函数和ANSI函数
- 2.4 C运行库中的Unicode函数和ANSI函数
- 2.5 C运行库中的安全字符串函数
 - 2.5.1 初识新的安全字符串函数
 - 2.5.2 在处理字符串时如何获得更多控制
 - 2.5.3 Windows字符串函数
- 2.6 为何要用Unicode
- 2.7 推荐的字符和字符串处理方式
- 2.8 Unicode与ANSI字符串转换
 - 2.8.1 导出ANSI和Unicode DLL函数
 - 2.8.2 判断文本是ANSI还是Unicode

第3章 内核对象

- 3.1 何为内核对象
 - 3.1.1 使用计数
 - 3.1.2 内核对象的安全性
- 3.2 进程内核对象句柄表
 - 3.2.1 创建一个内核对象
 - 3.2.2 关闭内核对象
- 3.3 跨进程边界共享内核对象
 - 3.3.1 使用对象句柄继承
 - 3.3.2 改变句柄的标志
 - 3.3.3 为对象命名
 - 3.3.4 终端服务命名空间
 - 3.3.5 专有命名空间
 - 3.3.5 复制对象句柄

第 部分 工作机制

第4章 进程

- 4.1 编写第一个Windows应用程序
 - 4.1.1 进程实例句柄
 - 4.1.2 进程前一个实例的句柄
 - 4.1.3 进程的命令行
 - 4.1.4 进程的环境变量
 - 4.1.5 进程的关联性
 - 4.1.6 进程的错误模式
 - 4.1.7 进程当前所在的驱动器和目录
 - 4.1.8 进程的当前目录
 - 4.1.9 系统版本
- 4.2 CreateProcess函数

<<Windows核心编程(第5版)>>

4.2.1 pszApplicationName和pszCommandLine参数

4.2.2 psaProcess, psaThread和bInheritHandles参数

4.2.3 fdwCreate参数

4.2.4 pvEnvironment参数

4.2.5 pszCurDir参数

4.2.6 psiStartInfo参数

4.2.7 ppiProcInfo参数

4.3 终止进程

4.3.1 主线程的入口点函数返回

4.3.2 ExitProcess函数

4.3.3 TerminateProcess函数

4.3.4 当进程中的所有线程终止时

4.3.5 当进程终止运行时

4.4 子进程

4.5 管理员以标准用户权限运行时

4.5.1 自动提升进程的权限

4.5.2 手动提升进程的权限

4.5.3 何为当前权限上下文

4.5.4 枚举系统中正在运行的进程

4.5.5 Process Information示例程序

第5章 作业

5.1 对作业中的进程施加限制

5.2 将进程放入作业中

5.3 终止作业中的所有线程查询作业统计信息

5.4 作业通知

5.6 Job Lab示例程序

第6章 线程基础

6.1 何时创建线程

6.2 何时不应该创建线程

6.3 编写第一个线程函数

6.4 CreateThread函数

6.4.1 psa参数

6.4.2 cbStackSize参数

6.4.3 pfnStartAddr和pvParam参数

6.4.4 dwCreateFlags

6.4.5 pdwThreadId

6.5 终止运行线程

6.5.1 线程函数返回

6.5.2 ExitThread函数

6.5.3 TerminateThread函数

6.5.4 进程终止运行时

6.5.5 线程终止运行时

6.6 线程内幕

6.7 C/C++运行库注意事项

6.7.1 用_beginthreadex而不要用CreateThread创建线程

6.7.2 绝对不应该调用的C/C++运行库函数

6.8 了解自己的身份

<<Windows核心编程(第5版)>>

6.8.1 将伪句柄转换为真正的句柄

第7章 线程调度、优先级和关联性

7.1 线程的挂起和恢复

7.2 进程的挂起和恢复

7.3 睡眠

7.4 切换到另一个线程

7.5 在超线程CPU上切换到另一个线程

7.6 线程的执行时间

7.7 在实际上下文中谈CONTEXT结构

7.8 线程优先级

7.9 从抽象角度看优先级

7.10 优先级编程

7.10.1 动态提升线程优先级

7.10.2 为前台进程微调调度程序

7.10.3 调度I/O请求优先级

7.10.4 Scheduling Lab 示例程序

7.11 关联性

第8章 用户模式下的线程同步

8.1 原子访问：Interlocked系列函数

8.2 高速缓存行

8.3 高级线程同步需要避免使用的一种方法

8.4 关键段

8.4.1 关键段：细节

8.4.2 关键段和旋转锁

8.4.3 关键段和错误处理

8.5 Slim读/写锁

8.6 条件变量

8.6.1 Queue示例程序

8.6.2 在停止线程时的死锁问题

8.6.3 一些有用的窍门和技巧

第9章 用内核对象进行线程同步

9.1 等待函数

9.2 等待成功所引起的副作用

9.3 事件内核对象

9.4 可等待的计时器内核对象

9.4.1 让可等待的计时器添加APC调用

9.4.2 计时器的剩余问题

9.5 信号量内核对象

9.6 互斥量内核对象

9.6.1 遗弃问题

9.6.2 互斥量与关键段的比较

9.6.3 Queue示例程序

9.7 线程同步对象速查表

9.8 其他的线程同步函数

9.8.1 异步设备I/O

9.8.2 WaitForInputIdle函数

9.8.3 MsgWaitForMultipleObjects (Ex) 函数

<<Windows核心编程(第5版)>>

9.8.4 WaitForDebugEvent函数

9.8.5 SignalObjectAndWait函数

9.8.6 使用等待链遍历API来检测死锁

第10章 同步设备I/O与异步设备I/O

10.1 打开和关闭设备细看CreateFile函数

10.2 使用文件设备

10.2.1 取得文件的大小

10.2.2 设置文件指针的位置

10.2.3 设置文件尾

10.3 执行同步设备I/O

10.3.1 将数据刷新至设备

10.3.2 同步I/O的取消

10.4 异步设备I/O基础

10.4.1 OVERLAPPED结构

10.4.2 异步设备I/O的注意事项

10.4.3 取消队列中的设备I/O请求

10.5 接收I/O请求完成通知

10.5.1 触发设备内核对象

10.5.2 触发事件内核对象

10.5.3 可提醒I/O

10.5.4 I/O完成端口

10.5.5 模拟已完成的I/O请求

第11章 Windows线程池

11.1 情形1：以异步方式调用函数

11.1.1 显式地控制工作项

11.1.2 Batch示例程序

11.2 情形2：每隔一段时间调用一个函数

11.3 情形3：在内核对象触发时调用一个函数

11.4 情形4：在异步I/O请求完成时调用一个函数

11.5 回调函数的终止操作

11.5.1 对线程池进行定制

11.5.2 得体地销毁线程池：清理组

第12章 纤程

第 部分 内存管理

第13章 Windows内存体系结构

13.1 进程的虚拟地址空间

13.2 虚拟地址空间的分区

13.2.1 空指针赋值分区

13.2.2 用户模式分区

13.3 地址空间中的区域

13.4 给区域调拨物理存储器

13.5 物理存储器和页交换文件

13.6 页面保护属性

13.6.1 写时复制

13.6.2 一些特殊的访问保护属性标志

13.7 实例分析

13.8 数据对齐的重要性

<<Windows核心编程(第5版)>>

第14章 探索虚拟内存

- 14.1 系统信息
- 14.2 虚拟内存状态
- 14.3 NUMA机器中的内存管理
- 14.4 确定地址空间的状态
 - 14.4.1 VMQuery函数
 - 14.4.2 示例程序：虚拟内存映射

第15章 在应用程序中使用虚拟内存

- 15.1 预订地址空间区域
- 15.2 给区域调拨物理存储器
- 15.3 同时预订和调拨物理存储器
- 15.4 何时调拨物理存储器
- 15.5 撤销调拨物理存储器及释放区
 - 15.5.1 何时撤销调拨物理存储器
 - 15.5.2 虚拟内存分配示例程序
- 15.6 改变保护属性
- 15.7 重置物理存储器的内容
- 15.8 地址窗口扩展

第16章 线程栈

- 16.1 C/C++运行库的栈检查函数
- 16.2 Summation示例程序

第17章 内存映射文件

- 17.1 映射到内存的可执行文件和DLL
 - 17.1.1 同一个可执行文件或DLL的多个实例不会共享静态数据
 - 17.1.2 在同一个可执行文件或DLL的多个实例间共享静态数据
 - 17.1.3 Application Instances示例程序
- 17.2 映射到内存的数据文件
 - 17.2.1 方法1：一个文件，一块缓存
 - 17.2.2 方法2：两个文件，一块缓存
 - 17.2.3 方法3：一个文件，两块缓存
 - 17.2.4 方法4：一个文件，零个缓存
- 17.3 使用内存映射文件
 - 17.3.1 第1步：创建或打开文件内核对象
 - 17.3.2 第2步：创建文件映射内核对象
 - 17.3.3 第3步：将文件的数据映射到进程的地址空间
 - 17.3.4 第4步：从进程的地址空间撤销对文件数据的映射
 - 17.3.5 第5步和第6步：关闭文件映射对象和文件对象
- 17.6 File Reverse示例程序
- 17.7 用内存映射文件来处理大文件
- 17.8 内存映射文件和一致性
- 17.9 给内存映射文件指定基地址
- 17.10 内存映射文件的实现细节

第18章 堆

- 18.1 进程的默认堆
- 18.2 为什么要创建额外的堆
 - 18.2.1 对组件进行保护
 - 18.2.2 更有效的内存管理

<<Windows核心编程(第5版)>>

- 18.2.3 使内存访问局部化
- 18.2.4 避免线程同步的开销
- 18.2.5 快速释放

18.3 如何创建额外的堆

- 18.3.1 从堆中分配内存块
- 18.3.2 调整内存块的大小
- 18.3.3 获得内存块的大小
- 18.3.4 释放内存块
- 18.3.5 销毁堆
- 18.3.6 在C++中使用堆

18.4 其他堆函数

第 部分 动态链接库

第19章 DLL基础

19.1 DLL和进程的地址空间

19.2 纵观全局

- 19.2.1 构建DLL模块
- 19.2.2 构建可执行模块
- 19.2.3 运行可执行模块

第20章 DLL高级技术

20.1 DLL模块的显式载入和符号链接

- 20.1.1 显式地载入DLL模块
- 20.1.2 显式地卸载DLL模块
- 20.1.3 显式地链接到导出符号

20.2 DLL的入口点函数

- 20.2.1 DLL_PROCESS_ATTACH通知
- 20.2.2 DLL_PROCESS_DETACH通知
- 20.2.3 DLL_THREAD_ATTACH通知
- 20.2.4 DLL_THREAD_DETACH通知
- 20.2.5DllMain的序列化调用
- 20.2.6 DllMain和C/C++运行库

20.3 延迟载入DLL

20.4 函数转发器

20.5 已知的DLL

20.6 DLL重定向

20.7 模块的基地址重定位

20.8 模块的绑定

第21章 线程局部存储区

21.1 动态TLS

21.2 静态TLS0

第22章 DLL注入和API拦截

22.1 DLL注入的一个例子

22.2 使用注册表来注入DLL

22.3 使用Windows挂钩来注入DLL

22.4 使用远程线程来注入DLL

22.4.1 Inject Library示例程序

22.4.2 Image Walk DLL

22.5 使用木马DLL来注入DLL

<<Windows核心编程(第5版)>>

- 22.6 把DLL作为调试器来注入
- 22.7 使用CreateProcess来注入代码
- 22.8 API拦截的一个例子9
 - 22.8.1 通过覆盖代码来拦截API0
 - 22.8.2 通过修改模块的导入段来拦截API
 - 22.8.3 Last MessageBox Info示例程序

第 部分 结构化异常处理

第23章 终止处理程序

第24章 异常处理程序与软件异常

- 24.1 通过实例理解异常过滤程序和异常处理程序
 - 24.1.1 Funcmeister1函数
 - 24.1.2 Funcmeister2函数
- 24.2 EXCEPTION_EXECUTE_HANDLER1
 - 24.2.1 一些有用的例子
 - 24.2.2 全局展开
 - 24.2.3 停止全局展开
- 24.3 EXCEPTION_CONTINUE_EXECUTION
- 24.4 EXCEPTION_CONTINUE_SEARCH0
- 24.5 GetExceptionCode2
- 24.6 GetExceptionInformation6
- 24.7 软件异常

第25章 未处理异常、向量化异常处理与C++异常

- 25.1 UnhandledExceptionFilter函数详解
- 25.2 即时调试
- 25.3 电子表格示例程序
- 25.4 向量化异常和继续处理程序
- 25.5 C++异常与结构化异常的比较
- 25.6 异常与调试器

第26章 错误报告与应用程序恢复

- 26.1 Windows错误报告控制台
- 26.2 可编程的Windows错误报告
- 26.3 对进程中所有的问题报告进行定制
- 26.4 问题报告的创建与定制
 - 26.4.1 创建一个自定义的问题报告
 - 26.4.2 设置报告参数：WerReportSetParameter
 - 26.4.3 将小型转储文件放入报告：WerReportAddDump8
 - 26.4.4 将任意文件放入报告：WerReportAddFile9
 - 26.4.5 修改对话框文本：WerReportSetUIOption0
 - 26.4.6 提交错误报告：WerReportSubmit0
 - 26.4.7 关闭问题报告：WerReportCloseHandle
 - 26.4.8 Customized WER示例程序
- 26.5 应用程序的自动重启与恢复
 - 26.5.1 应用程序的自动重启
 - 26.5.2 对应用程序恢复的支持

第 部分

附录A 构建环境

附录B 消息处理宏、子控件宏和API宏

索引

章节摘录

插图：第 部分 必备知识第1章错误处理1.1定义自己的错误代码1.2ErrorShow示例程序在深入讨论MicrosoftWindows提供的诸多特性之前，应该先理解各个Windows函数如何进行错误处理的。调用Windows函数时，它会先验证我们传给它们的参数，然后再开始执行任务。如果传入的参数无效，或者由于其他原因导致操作无法执行，则函数的返回值将指出函数因为某些原因失败了。

<<Windows核心编程(第5版)>>

媒体关注与评论

无论是Windows编程新手，还是完全用本机代码来编程或通过P/Invoke来调用，NET Framework未提供的API的老手，都会发现《Windows核心编程》的价值。

——Mark Russinovich，《深入解析Windows操作系统》作者 要想在Windows编程方面更上一层楼，迟早都需要好阅读并领会《Windows核心编程》的内容。

——Francis Glassborow，C/C++用户协会前主席 搞Windows程序设计有两方面的资源是不可或缺的：一是MSDN，另一个便是《Windows核心编程》。

——侯捷，著名技术作家

<<Windows核心编程(第5版)>>

编辑推荐

《Windows核心编程》是一本经典的Windows核心编程指南，从第1版到第5版，引领着数十万程序员走入Windows开发阵营，培养了大批精英。

Windows核心编程（第5版）针对WindowsXP，WindowsVista和WindowsServer2008全面修订。

主题广泛，内容丰富，讲解深入而精辟。

透过这本Windows编程经典，我们可以在专家的悉心指导下，洞悉Windows编程精华，深入了解高级编程技巧，编写出高性能的Windows应用程序。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>