Windows NT 高级编程技术

[美] JEFFREY RICHTER 著
郑全战、王毅、洛水、忻宏杰、孙静宇 译
郑全战、洛水 审校

清 华 大 学 出 版 社

(京)新登字 158 号

内 容 简 介

本书讲述了 Windows NT 编程最常用的特性:进程和线程管理、存储器管理、动态链接库、文件系统和文件 I/O、UNICODE,以及 Windows NT 和 16 位 Windows 的特性比较。

本书是 Win 32 应用程序开发者的编程指南,也是 Windows NT 程序开发者最好的参考书。

Advanced Windows NT: The Developer's Guide to the Win32 Application Programming Interface

Copyright © 1994 by Jeffrey M. Richter

本书中文版版权由 Microsoft Press 授予清华大学出版社独家出版发行。未经出版者书面许可,本书的任何部分都不得以任何方式或任何手段复制和传播。本书封面贴有激光防伪标签,无标签者不得进入销售。

图书在版编目(CIP)数据

Windows NT 高级编程技术/(美)里克特(Ricter, J.)著;郑全战等译. —北京:清华大学出版社, 1994.11

ISBN 7-302-01627-5

Ⅰ.W… Ⅱ.①里…②郑… Ⅲ.①编辑程序,Window②NT-程序设计③窗口软件-程序设计-操作系统 Ⅳ.TP316

中国版本图书馆 CIP 数据核字(94)第 11051 号

出版者:清华大学出版社(北京清华大学校内,邮编 100084)
印刷者:清华大学印刷厂
发行者:新华书店总店北京科技发行所
开 本:787× 1092 1/16 印张:32.5 字数:762 千字
版 次:1994 年 12 月第 1 版 1994 年 12 月第 1 次印刷
书 号:ISBN 7-302-01627-5/TP·689
印 数:0001—5000
定 价:47.00 元

目　　录

此为试读，需要完整PDF请访问：www.ertongbook.com

此为试读,需要完整PDF请访问: www.ertongbook.com

译　者　序

自从 1990 年 Microsoft 推出 Windows 3.0 以来, Windows 已经成为基于 Intel X86 微处理器计算机新的操作系统环境。Microsoft Windows 不仅为基于 MS-DOS 的计算机增加了图形用户接口, 而且扩展了 MS-DOS 的功能: 提供多任务, 改善了内存管理。

Windows NT 推进了这种发展和变革, 在 Windows 易于使用的基础上增加了高档操作系统的功能和特性。Windows NT 不需要 MS-DOS 就可以运行在 Intel 80X86 系列微处理器的计算机上, 同时还可以运行在非 Intel 80X86 系列微处理器的计算机上, 象 MIPS、DEL Alpha, 甚至 PowerPC 也开始支持 Windows NT。

Windows NT 还具备如下特性:

- 运行现存的基于 MS-DOS 和基于 Microsoft Windows 的软件。
- 在单处理器计算机和多处理器计算机上运行
- 支持 32 位编程
- 达到 C2 级安全性

Windows NT 是 Microsoft Windows 操作系统系列中最新、功能最强的成员。尽管由于 Windows NT 对计算机硬件要求较高而暂时难以普及, 但随着计算机硬件技术的发展, Windows NT 的目前版本 Windows NT 3.1、Windows NT Advanced Server 3.1, 与 Microsoft 即将推出的 Chicago(Windows 4.0), 以及 Microsoft 计划推出的 Daytona、Cairo 必将走上统一。到那时, 在所有操作系统中, Windows NT 必将傲视群雄, 独领风骚。

最大的收获在于及早的投资。可以预见, 在不久的将来, 谁熟悉 Windows NT 编程, 谁将获益匪浅。正如今日, 随处可见招聘广告中千篇一律的"会 Windows 编程者优先"。

驰骋天下, 好马伴君行。然而书籍繁多, 何处识"好马"? 《Windows NT 高级编程技术》对于那些想驰骋于 Windows NT 天地的程序员, 实在是一本难得的好书。

《Windows NT 高级编程技术》分章讲述了对大多数 Windows NT 应用程序开发者来说最为有用的 Windows NT 特性: 进程和线程管理、存储器管理、动态链接库、文件系统和文件 I/O、UNICODE, 以及 Windows NT 与 16 位 Windows 特性的比较。细细读来, 相信会获益匪浅。

全书共分十一章并有一附录。各章的主题分别是：进程和线程、内存管理、虚拟内存管理、内存映射文件、线程同步、Win32 子系统环境、动态链接库、局部线程存储、文件系统和文件 I/O、结构异常处理、UNICODE。附录讲解消息拆析器。

本书由郑全战、王毅、洛水、忻宏杰、孙静宇共同翻译而成。

由于水平所限，书中难免有错，欢迎批评指正。

译　者

于北京大学计算机系

1994 年 7 月

鸣　　谢

虽然这部书的封面上只出现了我一个人的名字,但许多朋友都以这样或那样的方式为本书的编写出了力。如果没有以下这些人的帮助和支持,仅凭我个人的努力和专业知识,我是不能完成这部书的编写工作的——感谢大家!

在整个编书过程中,Susan“Q”Ramee给予了我爱和支持。Sue还校对了各章节并针对程序范例想出了许多构想。当然,我还要感谢她的两只猫咪Nalt和Cat。当我在深夜无法入睡而决定写作时,Nalt和Cat经常陪伴着我。当我打字时,她们经常在键盘上跳来跳去而且扔掉我的笔记。我敢保证,读者所发现的这本书中任何打字错误都是Nalt和Cat所为。

Jim Harkins是我的一位最好的朋友。他对本书的I/O文件及文件变换的程序范例贡献很大。Jim还帮我考虑了许多线程、同步的问题。

Paul Yao也是我的一位好友,他引导我在Peovia如何获得成功。就这本书而论,Paul首次向我介绍了Windows NT存储器管理及有关章节的大量内容。Paul会说:“要想看看在MIPS机上有多少介绍,只需进入debugger并持续按F8键直到你进到finally模块!”当然我还要感谢Becky允许我把Pogacha工具——YUM带回家,他们真是太友好了。

Scott Ludnig,一位Microsoft公司Windows的领先开发者,总是极其耐心地回答我所有的问题。虽然我们有过争论,但Scott的确是令我非常尊敬的一位学者。

Lou Perazzoli, Steve Wood及Marc Lucovsky是Windows NT开发组的成员,他们审阅了部分章节并回答了有关线程及存储器管理的许多问题。

Chuck Mitchell, Sthe Salisbury和Jonathan Mark这几位是Visual C++ forWindows NT开发组的成员,他们回答了我有关结构化异常处理、本地线程存储,C运行时间库及连接方面的问题。

我还要提到并感谢Visual C++ for Windows NT开发小组的其他几位成员。他们是:Byron Dazey、Ericlang、Dan Spalding、Matthew Tabbs Bruce Johnson、Jon Jorstad、Dave Henderson、T. K. Brackman。

Bernie McIlvoy帮助我在DEC Alpha机上测试了这些应用程序。

Mark Cliggett, Cameron Ferroni, Eric Fogelin, Randy Kath和Steve Sinofsky在我不熟悉的主题上予我以帮助。

我的Microsoft Press的编辑Nancy Siadek希望因她为这部书作出的努力和贡献而获得奖励。我敢肯定她都不知道她为这部书付出了多少心血。Nancy在我和她相处写书的这短暂的时间内教我的比我这一生学的都要多。

Jeff Carey是我在Microsoft Press时的技术编辑。Nancy提出的许多问题使我要重写书中的一些材料,而Jeff帮助我解决了这一难题。

此为试读,需要完整PDF请访问: www.ertongbook.com

我也要感谢 Microsoft Press 组的其他成员。虽然他们中的许多人我从未见过，但我很欣赏并感谢他们的努力，他们是：Erin O'connor，Laura Sackerman，Peggy Herman，Lisa Iversen 和 Barb Runyan。

还要感谢：

Borland International 的 Dan Horn，因为有一些章节中有他的建议和他编写的材料

Jim Lane，Tom Van Back 和 Rich Peterson，他们一直在 DEC Alpha 编辑器上修正错误。

Dean Holmes 这位 Microsoft Press 的先导者，为我签了名并且在我忙于搬家时帮我赶上了进度。

Gretchen Bilson 和 Microsoft System Journal 的每一位成员都鼓励我继续写作。

Charles Petzold 向我介绍了 Microsoft Press，并给我提供热乎乎的酸汤。

Carlos Richardson，他帮我安装了 TJ2NET(我的家庭网络系统)，使之能在我的新家开始运行。

Premia 公司的每一个人都向我提供了早期版本的 CodeWright for Windows NT，这样我可以利用强大的文本编辑器编制书中的范例程序。

Shapeware，Inc. 的每一位成员都向我提供了 Viso，这样我就能创造原始技术图形。

John Socha 激励我一路由 Philadephia，PA 迁到 Bellevue，WA。

Michele Leialoha 甚至将书橱也弄得挺滑稽。

Donna Murray，她在这几年中给了我爱护、支持和友谊。

我的兄弟 Ron 一直在帮我寻找 Patrick Moraz 的“Salamander”的复印件。虽然你从未找到过它，但我知道你尽了力。我将在 Peter Grobriel 下次到城里来时请他在你的高尔夫球棍上签名。

还有我的母亲和父亲，我要感谢他们许多年来给我的爱和支持。

前　言

我很高兴能够编写这部书。与其它事情相比我更喜欢处于科技的前沿并能学到新东西。Windows NT 就显然处于技术的前沿,这里有很多新知识可学。如果你已经是一位 Windows 编程员,当你仅学到一点修正你已有程序的简单技术之后,你会发现你已经能编写 Windows NT 的应用程序了。但这些被修正的程序不会利用 Windows NT 提供的崭新及强大的功能。

当你利用 Windows NT 工作时,一开始并不能在你的应用程序中综合地利用这些功能。Windows NT 的许多功能使编程变得极为简单。而且,正如我很快发现的那样当我修正我以前的程序时,我可以删去我原有程序中的很大一部分代码,并可通过调用 Windows NT 提供的独特功能代替它们。

利用这些崭新的功能是如此的方便,以致于我希望在未来我能集中我的精力来编写 Windows NT 的应用程序。事实上,最近我已经帮一些人调试过 16 位 Windows 应用程序,这些程序在系统中不能正常工作。我们通宵工作却没能发现问题。因为利用 Debugger 这些问题不能显示出来,所以我们不能使用 Debugger。在修正了错误以后,我说:"在 Windows NT 中不可能出现这种错误。Windows NT 真是太好了。"

我一直尽力跟上 Windows NT 开发者的步伐。纵观整个 Windows NT 的发展过程,Microsoft 不断增加新的功能及新的标志。这意味着我不得不一直不断地学习并编写有关这些新特点的书。但是后来发生了一件令人高兴的事。在我的生日,1993 年 7 月 27 日这一天,Windows NT 编写组给了我一个作者所能要求的最好的礼物:他们放弃了 Windows NT 的进一步的开发。

就是说,他们再也不会给这一系统增加新的功能了,我也长出了一口气。

这部书是我在利用 Windows NT 工作而取得的经验的结晶。在过去的这些年里,我特地将精力集中在我认为对大多数 Windows NT 应用开发者最有用的一些方面,即:过程和线程控制、存储器管理、动态链接库、系统文件和 I/O 文件、以及 Windows NT 与 16 位 Windows 的特性的比较。

我确信 Windows NT 将会成为一种标准的操作系统。首先,它将替代大型机和微型计算机上的操作系统,然后在若干年以后(也许很多年),当存储器价格开始下降时,Windows NT 也将成为个人计算机的操作系统。这本书将帮助你掌握这一重要的环境的开发应用。

0.1　术语解释

当我在一些有关的场合讲话时,人们经常问的一个问题是:Win32 和 Windows NT

有什么不同? 我已回答了这个问题,并且为了指出这些术语的确切意义而很费了一些周折。因此,为了以后不再引起混乱,我将尽力把它们解释清楚。

0.1.1 Win32 API 及其支持的工作台

Win32 是一种 API 的名称,这个解释就足够了。在源程序中用于调用的那些功能的设置包含在 Win32 API 中。因为你调用了 Win32 API 中的函数,因此在编写程序时,实际上你只是在编写一个 Win32 程序。

Win32 API 可在一些工作平台上运行,其中之一就是 Windows NT。Windows NT 是一种可运行多种类型应用程序的操作系统。例如, Windows NT 可运行 OS/ 2. 1X 应用程序、MS-DOS 应用程序及 16 位 Windows 应用程序,另外, Windows NT 也可运行 Win32 应用程序(调用 Win32 API 中函数的程序)。

其它一些已有的工作平台也可运行 Win32 应用程序。事实上,这些工作平台中的一个在 Windows NT 之前已出现。这一工作平台被称为 Win32s。Win32s 的名称中包含 Win32,这容易引起混淆。当编写一 Win32 程序并编译它后,它在 Win32s 和 Windows NT 工作平台上都可运行。

现在我们解释一个有关 CPU 的内容。Windows NT 可在具有不同 CPU 的计算机上运行。如果利用一个具有 X86 CPU 的编译器编译 Win32 源程序,那么在任何运行于 X86 计算机上的 Win32 工作平台上都可运行这一 Win32 应用程序。但是,如果针对 MIPS CPU 编译了一个 Win32 应用程序,则只能在 MIPS 计算机中的 Win32 工作台上运行这一应用程序。当前, MIPS 计算机上唯一的 Win32 工作台就是 Windows NT。Microsoft 使 Win32s 工作平台只能用于 X86 计算机。

所有这些工作平台都包含所有 Win32 函数的实现。这意味着除了当前运行的工作平台,其它 Win32 APIh 的函数都可调用。但是有些实现是受到限制的。例如, Win32 API 包含一被称为 Create Thread(建立线程)的函数,它允许应用程序建立一新的执行线程。在 Win32 API 的 Windows NT 实现中,这一函数确实存在——它建立了一个新的执行线程。但是, 在 Wind32s API 的 Win32 实现中, 这一函数只简单地返回到空闲状态(NULL),它指出不能建立新的执行线程。

具有这一限制的原因是 Win32s 实际上只是 16 位 Windows 的扩展。Win32s 通过形实转换调用 16 位 Windows 函数运行 Win32 API 的绝大部分应用程序。因为 16 位 Windows 不支持新的执行线程的建立,因此 Win32s 不支持这一函数。但请记住,虽然有些实现是受到限制的,但是 Win32s 可实现 Win32 的所有函数。由以上讨论可看出此书中的所有程序范例都可进行编译并在 Win32 工作平台上运行。但另一方面,此书中讨论的大部分函数(例如: 多线程编程, 虚拟内存, 内存映射文件)在 Windows NT 工作平台上可完全运行,但在 Win32 工作台上运行要受到一些限制。因为程序中调用的许多函数在 Win32s 工作台上是受限的,所以读者不可能实际运行这些程序范例。只有在 Windows NT 工作平台上才能运行这些程序范例,且可看到它们的全貌。

0.1.2 Windows NT 及其子单元

Windows NT 是一完备操作系统的名称。当你购买了 Windows NT 时,实际上是购买了一个能运行 MS-DOS 应用程序, OS/21. X 应用程序, POSIX 应用程序、16 位 Windows 应用程序和 Win32 应用程序的完备系统。

这一完备系统包括一些子单元。例如, POSIX 应用程序实际上是调用包含在 POSIX 子系统中的功能, OS/2 应用程序调用 OS/2 子系统中的功能, Win32 应用程序调用 Win32 子系统中的功能, 等等。

Windows NT Executive(执行体)管理内存分配、线程文件及其它有关这些子系统的所有资源。Windows NT Executive 是 Windows NT 操作系统的核心。没有它, 就不称其为操作系统。但是, 如果放弃 POSIX 或者 OS/2 子系统, 那么此操作系统仍可运行 Win32 应用程序。

在 Windows NT 操作系统中, Win32 系统对 Win32 API 的应用程序产生响应。这一子系统通过利用从 Windows NT Executive 得到的优越之处运行 API。因此在本书中, 读者会见到诸如这样的语句:

Win32 分配内存并返回。

Windows NT 分配内存并返回。

Win32 子系统分配内存并返回。

Windows NT Executive 分配内存并返回。

当我使用一条语句时, 我是在力图说明有关操作系统的某一部分负责所需操作的细节。在某些情况下, 我的目的不明确。例如, 通过调用 GlobolAlloc 功能来分配内存有可能需要 Win32 子系统及 Windows NT Executive 的共同合作才能实现。但是既然我也不太确定哪一子单元是最终负责的, 我会因 Win32 子系统决定如何处理针对 GlobalAlloc 的调用而称“Win32 分配内存”。

记住当我提供某一叙述的更多的细节时, 我要明确我是在讨论 Win32 API 的 Windows NT 实现程序。Wind32 API 的其它实现是利用 Win32 另外一些函数。

0.2 我对大家的期望

这部书是针对那些已具有编写有关 16 位 Windows 程序经验的 Windows 开发者而编写的。不过读者并不需要有关 16 位 Windows 深入的知识, 而只要具备 Windows 编程的基本知识就够了, 这些基本知识包括窗口过程、窗口消息、对话框和内存管理。这本书的内容涵盖了运行在 Windows NT 操作系统下的已被引入 Win32 API 中的新功能。本书并不是力图讲解 Windows 编程的入门知识。这本书也包含了当你将 16 位 Windows 应用程序修正到 32 位时所需解答的问题。

0.3 关于应用程序范例

编写应用程序范例的目的是以实际程序证明如何利用 Windows 的先进功能。读者还不能通过阅读足够的书籍来更新已通过编写应用程序而获得的知识和经验。我在学习 Windows NT 时也有同样的经历。本书所出现的许多应用范例是我在理解了 Windows NT 如何工作的情况下经过努力而编写的。

0.3.1 用 C 语言编写程序

当在决定用哪一种语言编写应用程序范例时,我在选择 C 还是 C++ 时产生了困扰。对大项目而言,我总是使用 C++ ,但实际上大多数 Windows 的编程人员目前还未使用 C++ ,而且我也不想因选择了 C++ 而疏远了我的很大部分且很有潜力的读者。

0.3.2 Message Cracker Macros(消息拆析宏)

如果读者不利用 C++ 和 Windows 类库(类似 Microsoft Foundcotion Classes)编写程序,那么我建议读者利用定义在 WINDOWS X.H 头文件内的消息拆析宏,这些宏定义使你的程序更便于编写、阅读和保存。我认为信息拆析宏的功能非常强大,因此我在本书附录中描述了信息拆析宏存在的必要性及如何有效地利用它们。

0.3.3 有关 16 位 Windows 编程的知识

虽然很需要有 16 位 Windows 编程的经验,但是书中所有的程序都不依赖于有关 16 位 Windows 编程的深入知识。程序范例是在假设读者已熟悉对话框及其子控制的建立和管理的基础上编写的,读者仅需具有极少的有关 GDI 及 KERNEL 功能的知识即可。

在给本书起名时,我确实对 16 位 Windows 和 Windows NT 进行了大量比较。如果读者已理解了 16 位 Windows 是如何运行的,那就不难理解在 Windows NT 中又是如何变化的。

0.3.4 不相关代码

我曾想从程序范例中删去那些与我所要阐明的技术不直接相关的代码。不幸的是,无论编写什么样的 Windows 程序这都是不可能的。例如,大多数 Windows 编程书重复了本书中每一个例程中出现的注册 Windows 类的代码。我已尽了最大努力来减少这一类不相干的代码。

我减少不相干代码的方法之一是使用对 Windows 编程者来说不总是显而易见的那些技术。例如,大多数应用程序范例的用户界面是对话框。事实上,大多数程序范例在 Windows 中有一行简单地调用 DialogBox(对话框)功能的代码。其结果是,没有一个程序范例初始化 WNDCLASS 结构或者是调用 RegisterClass 函数。另外,在第 9 章仅有一个应用程序(FileChng),在其中有一组重复循环指令。

0.3.5 应用程序的独立性

我已尽力保持应用程序互相独立。例如,内存映射文件这一章是包含内存映射文件应用程序的唯一的一章。因为我在设计构造程序时已使它们相互独立,因此读者阅读前后各章时不必互相对照查找。

读者偶尔会发现在一应用程序中使用了前面几章已出现的技术及信息。比如,SEHEXCPT 应用程序出现在第 10 章"Structured Exception Handling",它论证了如何控制虚拟内存。因为 SEH 是控制虚拟内存的一个非常有用的工具,所以我曾决定将这两个专题写在一个应用程序中。为了完全理解这一应用程序,读者在验证 SEHEXCPT 应用程序之前应阅读第三章。

0.3.6 STRICT 一致性

所有的应用程序范例都已用已定义的能发现常见代码错误的 STRICT 标志进行了编译。例如,传递给函数的不正确的句柄类型是在编译期间,而不是在运行期间被发现。

有关利用 STRICT 标志符的更多的信息,参见包含在 Windows NT SDK 中的编程技术这一部分。

0.3.7 错误检查

检查错误在软件工程中占很大部分。不幸的是,即使是适当的错误检查也会使软件工程的规模及复杂程度呈指数上升。为了使程序范例更简单明了,我未增加太多的错误检查代码。如果读者利用我的一些代码片断并将它们组合到自己设计的程序中,最好能详细地检查这些代码并增加适当的错误检查项。

0.3.8 没有错误

我很希望所有的应用程序都没有错误。但是正如所有的软件一样,在找到一个错误之前,还是可以这么说的。当然,为了发现问题我已将我的代码调试了若干次。如果读者发现了错误,我希望能通过我的 CompuServe 地址: 70444, 24 告诉我。

0.3.9 测试平台及环境

我为这本书而作的大部分研究和开发工作是在仅具有 Intel 486 CPU 的计算机上完成的。我在 MIPS 计算机和 DEC Alpha 计算机上也已对应用程序进行了重新编译和测试实验,这时我使用了在这些工作平台上与 Windows NT 配套的编译器及连接器。

另外通过利用 Visual C++ for Windows NT(仅对 X86 而言),我已编译并连接了这些应用程序范例并且通过利用 Win32 的 Borland 32 位 C++ 编译器测试了许多程序(同样,也是仅对 X86 而言)。在利用 Win32 的 Borland 32 位 C++ 编译器时,有些程序需要进行调整。

对于大多数程序范例,我没有使用软件特定的扩展编译器。

无论用什么样的计算机运行 Windows NT,也不管是利用什么工具编译和连接应用

此为试读,需要完整PDF请访问: www.ertongbook.com