<<精通.NET互操作>>

图书基本信息

书名：<<精通.NET互操作>>

13位ISBN编号：9787115204349

10位ISBN编号：7115204349

出版时间：2009-5

出版时间：人民邮电出版社

作者：黄际洲,崔晓源

页数：419

前言

自从2000年微软公司．NET平台问世以来，全球已经有超过400万开发人员使用．NET平台进行软件开发。
对于．NET来说，这无疑是一个巨大的成功。
这不仅仅体现在商业上的成功，其核心价值在于．NET为基于微软Windows平台的软件开发过程提供了一种新颖、高效的编程模型。
在该模型下，开发人员能够更容易地将精力集中在其特定的开发情景中，而不用过多地关注消息循环、窗口过程等操作系统底层的处理。
目前，基于-NET平台的技术和开发环境正处于飞速发展的时期。
在本书即将出版之际，微软公司已经正式发布了NET Framework 4．O所支持的新特性以及预览版。
另一方面，由于历史的原因，在．NET出现之前，开发人员已经编写了大量经过严格测试且可复用的非托管代码。
它们以C库函数、C++类库以及COM组件的形式存在于诸多应用程序和框架之中，并承担着非常重要的角色。
但由于在托管和非托管对象模型之间，数据类型、方法签名和错误处理机制都存在很大差异，从而使两种编程模型之间的代码互用和移植更加复杂。
因此，在很长一段时期内，开发人员必须面对．NET与久经考验的“遗留代码（1egacy code）”长期并存的局面。
当然，开发人员可以选择．NET平台，使用托管代码重写这些已有的非托管代码。
但这个重写的过程势必会枯燥无味，而且项目经理也不会在项目进度中安排大量的时间以进行重写代码的工作。
更让开发人员感到尴尬的是，很多时候，即使花了很大代价对非托管代码进行了重写，但还是不能保证重写后的托管代码像那些久经考验的非托管代码一样正常或高效地工作。
因此在很多情况下，重用已有非托管代码就成了最经济、可行的解决方案。
以下是这些情况中的一些典型案例。
开发人员所在的部门一直使用第三方提供的COM组件为产品的核心功能提供支持。
而新业务要求使用．NET平台。
这就出现了一个问题。
一方面公司已经为这些COM组件投入了大量的资金，不会轻易放弃这些组件。
另一方面开发部门使用．NET平台进行开发，无法直接使用这些COM组件。
因此，有效地在．NET平台中重用这些COM组件就成为产品成功的关键要素。

<<精通.NET互操作>>

内容概要

本书介绍Windows平台上的托管代码与非托管代码之间进行互操作的各种技术，包括由.NET提供的各种互操作方法、属性以及各种工具的用法及其工作原理。
本书包括3部分，平台调用——主要用于解决在托管代码中调用非托管程序设计语言编写的flat API（如Win32 API、C/C++风格的API等）的问题；C++ Interop——技术专门用于解决托管代码与C++编写的非托管代码之间的互操作问题；COM Interop——介绍了使用COM Interop解决在托管代码中调用COM组件，以及在COM中调用托管类型的问题。

本书适合所有在开发过程中需要涉及到托管代码与非托管代码进行交互操作的.NET开发人员阅读使用。
不论是开始学习.NET编程的开发人员，还是刚刚接触互操作的资深.NET开发人员，都能从本书中获益。

作者简介

黄际洲，2004-2007年连续四年微软最有价值专家（MVP）。
感兴趣的研究方向主要包括自然语言处理、信息检索、聊天机器人等。
他曾翻译了三本游戏编程方面的书籍：《Direct 3D中的2D编程》、《游戏编程All in One》及《Directx角色扮演游戏编程》。

<<精通.NET互操作>>

书籍目录

章节摘录

插图：本节主要讲述了3种优化平台调用性能的方法。
（1）将。
Dlllmport属性的ExactSpelling字段设置为true，显式地指定要调用的非托管函数的名称，以优化平台调用在非托管DLL中搜索函数的方式。
（2）尽可能地使用blittable类型。
由于采用blittable类型能够减少平台调用过程中耗费在数据封送处理上的时间，从而能极大地提升平台调用的性能。
（3）尽可能避免从LJnieode到ANSI的转换。
由于．NET采用的是unicode编码，如果要调用的非托管函数接收的是ANSI字符（串），那么封送拆收器必须先将字符（串）从LJnieode转换为ANSI，再从托管内存中将字符（串）复制到非托管内存中。这样不仅编码转换会牺牲性能，而且复制操作也会耗费时间。
除了上面介绍的这3种经过对比测试一一验证了的、优化平台调用性能的方法外，还有其他一些可以优化平台调用性能的方法。
比如采用IntPtr作为复杂类型的封送数据类型，或尽可能减少平台调用次数，在非托管代码中尽可能多地完成操作，而只进行获取最终计算结果的平台调用等，这些方法都能够提升平台调用的性能。
在实际进行平台调用时，需要对造成性能损耗的原因进行深入分析，只有弄清了造成损耗的原因，才能更有针对性地进行性能优化。

媒体关注与评论

托管代码与非托管代码之间的交互是许多程序员在.NET开发平台上不得不面对的任务。
《精通.NET互操作：P/Invoke，C++Interop和COM Interop》这本书深入而透彻地解析了.NET支持的三种与原生代码互操作的技术，作者以自己的经验讲述了原生代码与托管代码之间互操作所涉及到的编程要点，以及背后的一些实现原理。
书中提供的实例有助于程序员快速领会并掌握.NET与原生代码互操作技术的用法。
我建议在.NFT平台上工作的程序员读一读这本书。
——潘爱民，著名技术作家，著有《COM原理与应用》等多部畅销书，并翻译了多部经典名作

<<精通.NET互操作>>

编辑推荐

《精通.NET互操作P/Invoke,C++Interop和COM Interop》涵盖了：使用P/Invoke调用C库函数及windows API；使用C++Interop与C++类库及核心算法库进行交互；使用COM Interop实现托管代码与COM之间的交互。

版权说明