

## <<高级.NET程序设计>>

### 图书基本信息

书名：<<高级.NET程序设计>>

13位ISBN编号：9787302067894

10位ISBN编号：7302067899

出版时间：2003-7-1

出版时间：清华大学出版社

作者：Simon Robinson

页数：467

字数：780000

译者：冉小旻

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>

## <<高级.NET程序设计>>

### 内容概要

本书详细、专业地讲述了.NET应用程序的工作原理，深入探讨了.NET的一些高级技术，其中包括中间语言、CLR工作原理、应用程序操作性能的优化和系统资源使用情况的剖析、线程同步技术、高级Windows Forms技术、如何使用WMI管理计算机中的资源、如何动态生成代码以及.NET中的代码访问安全性和密码术等内容。

本书适用于有一定编程基础并参C#有所了解的.NET开发人员。  
此外，读者还必须熟悉.NET的基本概念和主要的类库。

## &lt;&lt;高级.NET程序设计&gt;&gt;

## 书籍目录

第1章 中间语言导论1.1 IL程序集简介1.2 IL原理1.3 IL编程1.4 IL调试1.4.1 VS.NET中的调试1.4.2 调试高级语言编译后得到的IL代码1.4.3 其他调试程序:CorDbg1.4.4 IL中的编译时错误1.5 小结第2章 中间语言深度挖掘2.1 对象类型和值类型实例2.1.1 实例字段2.1.2 定义实例方法和属性2.1.3 初始化和实例构造函数2.1.4 虚拟方法2.1.5 封箱和开箱2.2 枚举2.3 数组2.4 通过P/Invoke调用非托管代码2.5 定义二进制数据2.6 异常处理2.7 属性2.8 反汇编IL和循环处理2.9 小结第3章 CLR的运行原理3.1 .NET Framework组件和ECMA标准3.1.1 ECMA标准3.1.2 Framework SDK资源3.1.3 共享源CLI3.2 值/引用类型系统3.2.1 引用类型3.2.2 值类型3.2.3 封箱类型3.2.4 System.ValueType和System.Enum3.2.5 字段调整3.2.6 使用C++直接访问托管堆内存3.3 JIT编译:验证和确认3.3.1 代码验证3.3.2 类型安全验证3.4 托管代码和非托管代码3.4.1 非托管代码的调用原理3.4.2 混合托管类型和非托管类型3.5 小结第4章 程序集4.1 内部视图:程序集的物理结构4.1.1 PE文件4.1.2 CLR的PEA扩展4.1.3 资源和资源文件4.2 外部视图:程序集的逻辑结构4.2.1 程序集的标识4.2.2 读取程序集的内容4.2.3 探讨程序集缓存4.3 查找程序集4.3.1 Microsoft编译器查找程序集的原理4.3.2 VB.NET查找程序集的原理4.3.3 CLR探查程序集的原理4.4 生成程序集4.4.1 程序集实用程序4.4.2 编译资源文件4.4.3 本地化及附属程序集4.4.4 为程序集签名4.5 综合应用4.5.1 命令行GreetMe示例4.5.2 VS.NET GreetMe示例4.6 小结第5章 无用单元收集5.1 使用无用单元收集的原因5.1.1 C/C++样式的清除5.1.2 引用计算5.1.3 无用单元收集5.2 .NET无用单元收集器的运行原理5.2.1 调用GC5.2.2 获得对程序的控制5.2.3 标识无用单元5.2.4 压缩堆5.2.5 代5.2.6 析构函数和恢复5.2.7 通过编程控制无用单元收集器5.3 实现Dispose()和Finalize()5.3.1 Finalize/Dispose()的语义5.3.2 清除非托管资源5.3.3 包含托管和非托管资源的类5.3.4 实现Dispose()和析构函数的指导原则5.4 弱引用5.5 小结第6章 改进性能6.1 托管还是非托管6.1.1 .NET及其未来6.1.2 .NET的性能优点6.2 JIT编译器优化6.3 性能建议6.4 小结第7章 剖面分析和性能计数器7.1 Windows对性能监控的支持7.2 理解内存7.2.1 通过任务管理器访问内存7.2.2 UseResources示例7.3 性能计数器7.4 PerfMon7.4.1 .NET性能计数器7.4.2 通过性能计数器编码7.4.3 MonitorUseResources示例7.4.4 注册自己的性能计数器7.5 剖面分析7.5.1 选择剖析工具7.5.2 编写自己的Profiling Timer代码7.5.3 CompuwareProfiler示例程序7.5.4 配置剖析工具7.5.5 Profiling API7.6 小结第8章 动态代码生成8.1 使用动态代码生成的理由8.1.1 开发者工具8.1.2 基于性能的原因8.2 体系结构8.3 使用Reflecion.Emit类编码8.3.1 创建一个已保存的可执行程序集8.3.2 创建并运行DLL程序集8.4 使用CodeDom类编码8.4.1 创建Dom模型8.4.2 将DOM转换为源代码8.4.3 将源代码转换为IL代码8.4.4 CodeDom类示例8.5 小结第9章 线程9.1 CLR线程支持9.1.1 托管线程的类型9.1.2 线程标识9.1.3 枚举非托管线程9.2 多线程技术9.2.1 异步委托调用9.2.2 显式地创建您自己的线程9.2.3 定时器9.2.4 内置的异步支持9.2.5 将项目显式排列到线程池中9.3 异步委托9.4 同步变量访问9.4.1 数据同步原理9.4.2 线程同步结构9.4.3 线程同步示例9.5 定时器9.6 显式地创建和终止线程9.7 小结第10章 管理设备10.1 WMI的基本概念10.1.1 一些WMI示例10.1.2 WMI结构10.1.3 WMI对象模型10.1.4 WMI查询语言10.2 使用System.Management类执行查询10.3 异步处理10.4 接收通知10.5 小结第11章 高级Windows Forms技术11.1 Windows消息的后台处理11.1.1 处理消息11.1.2 Windows窗体和消息队列11.1.3 利用消息循环11.2 消息循环示例11.2.1 直接处理消息11.2.2 BdgInInvoke()示例——初始化一个应用程序11.2.3 Abort对话框示例11.3 支持XP的控件11.4 非矩形窗口11.5 自绘形窗口11.6 图形11.6.1 GDI和GDI+的比较11.6.2 Screenshot示例11.7 小结第12章 代码访问安全性12.1 代码访问安全性概念12.1.1 针对单个程序集的CAS12.1.2 针对多个程序集的CAS12.1.3 CLR权限12.2 与Windows安全性的关系12.3 默认的安全策略12.3.1 代码组12.3.2 权限集12.4 利用CAS编写代码12.4.1 强制性安全12.4.2 声明性安全12.4.3 好的编码实践12.5 CAS的后台处理12.6 设置自定义权限12.7 确认权限12.8 小结第13章 密码术13.1 密码术的作用13.2 对称加密13.3 公钥加密13.3.1 密钥大小13.3.2 会话密钥13.4 散列法13.5 数字签名13.6 凭证13.6.1 凭证的概念13.6.2 认证机构13.6.3 Windows密码术模型13.6.4 创建凭证13.6.5 通过编程读取凭证13.7 小结

<<高级.NET程序设计>>

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>