

<<C#与.NET 3.5高级程序设计>>

图书基本信息

书名：<<C#与.NET 3.5高级程序设计>>

13位ISBN编号：9787115196910

10位ISBN编号：7115196915

出版时间：2009-3

出版时间：人民邮电出版社

作者：特罗尔森

页数：1107

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>

<<C#与.NET 3.5高级程序设计>>

前言

~应用程序开发的目标始终如一，就是在最短时间内制作出最好的软件。

当今最流行的快速开发平台是什么，毫无疑问，是微软公司的.NET Framework。

经过近8年的发展，微软的.NET已经变得非常庞大和成熟，而且.NET发展的步伐越来越快。

2002年年初，微软发布了Visual Studio.NET 2002开发工具，也带来了.NET框架的第一个版本1.0。

2003年春天，微软发布了Visual Studio.NET 2003以及.NET Framework 1.1。

尽管只是1.1，但是.NET Framework 1.1比1.0多了几项重大更新，例如对移动设备的支持（精简版.NET）和对ODBC / Oracle数据库的支持。

2005年年底，.NET Framework 2.0随着Visual Studio 2005以及SQL Server2005一起发布。

无论是Visual Studio 2005、C#2.0还是ASP.NET 2.0，在易用性、安全性等方面都比前辈优秀很多。

C#2.0中的泛型、迭代器、可空类型、匿名方法、分部类等特性确实给了我们很多方便。

2006年年底，微软随Vista操作系统推出了.NET Framework 3.0。

.NET Framework 3.0在2.0的基础上增加了几个重要组件：WCF、WPF、WF和WCS（本书后面会对WCF、WPF~~HWF进行比较详细的介绍）。

WCF：Windows Communication Foundation，最完整的集成化通信管理框架。

WCF编程模型把Web服务、.NET远程技术、分布式事务和消息队列统一到单个面向服务的编程模型中，从而实现了真正意义上的分布式计算。

WPF：Windows Presentation Foundation，最华丽的界面技术框架。

WPF提供了一种一致的方案来构建编程模型，并且支持使用更为丰富的控件和设计技术来开发Windows程序。

WF：Windows Workflow Foundation，最灵活和最强大的工作流平台。

WF是微软用来定义、执行和管理工作流的编程模型、引擎和工具的总称。

WCS：Windows CardSpace，最安全的个人数字标识解决方案。

2007年年底，微软发布了Visual Studio 2008以及.NET Framework 3.5。

在.NET Framework 3.0的基础上，3.5又新增了一些程序集，并引入了LrNQ、ASP.NET AJAX等功能（本书基于C#3.0，而且也会对LINQ技术进行简单介绍）。

c#3.0：提供了隐式类型变量、自动属性、扩展方法、对象初始化语法、Lambda表达式、匿名方法、分部方法等新特性，其实几乎所有的新特性都是因LINQ而生的。

LINQ：语言集成查询，官方组件包括LINQ to SQL、LINQ to XML、LINQ to DataSet等。

当然，还有很多第三方公司或个人推出的LINQ to XXX。

可以通过这些组件使用统一的查询语言对各种基于磁盘、网络甚至服务的数据源进行查询。

ASP.NET AJAX：其实ASP.NET AJAX最早并不是在.NET Framework 3.5中出现的，只不过.NET Framework 3.5集成了ASP.NET AJAX。

关于.NET Framework 3.0中四大技术的扩展，我们也可以认为.NET Framework 3.5提供了WCF / WPF / WF / WCS的2.0版本。

~

<<C#与.NET 3.5高级程序设计>>

内容概要

《C#与.NET 3.5高级程序设计》(第4版)是C#领域久负盛名的经典著作，深入全面地叙述了C#编程语言和.NET平台核心，并以大量示例剖析相关概念。书中介绍了C#的各种语言构造、.NET 2.0的类、核心API、公共中间语言(CIL)、动态程序集和ASP.NET扩展等内容；同时也介绍了.NET 3.0和.NET 3.5中新的编程API，包括WPF、WCF和WF的功能；另外，还介绍了最新的C# 3.0编程语言、LINQ编程技术、COM与.NET的互操作性以及平台无关的.NET开发。

<<C#与.NET 3.5高级程序设计>>

作者简介

作者：(美国)特罗尔森 (Andrew troelsen) 译者：朱晔 肖逵 张大磊 AndFew Troelsen，世界级c#专家。微软；Visual C#MVP，他是著名的微软技术咨询企业Intertech的合伙人和副总裁。

该公司的客户包括微软、霍尼韦尔、美国宇航局等。

他曾为MSDN网站和MacTech网站撰写了有关各种操作系统平台上.NET技术的文章，并经常在业界主要技术会议上发表演讲和开设技术讲座。

除本书外，他还撰写了COM and .NET Interoperability和Visual Basic.NET and the .NET Platform：An Advanced Guide等十多部.NET技术方面的著作。

译者简介：朱晔，微软ASP.NET MVP，熟悉.NET、c#、ASP.NET、SQL Server等技术。

擅长基于.NET的分布式电子商务网站、互联网网站和网游平台的架构设计。

近期出版了畅销图书《ASP.NET第一步——基于c#和ASP.NET 2.0》和译著《SQL server 2005范例代码查询辞典》等。

个人博客：<http://www.cnblogs.com/lovecherry>。

张大磊，超过10年的软件研发实践，在网络安全、地理信息、医疗、生产制造和商业智能等多个领域积累了较为丰富的经验。

近年来主要关注微软平台技术，曾任微软公司开发技术经理。

业余时间与大中华区各地的微软技术架构师和最有价值专家进行广泛的技术交流，并多次受邀在TechED、MSDN等技术活动中发表主题演讲。

也是多个开源项目的贡献者，多个在线社区的特邀技术专家与微软金牌讲师的评委。

可以通过<http://kmsdpe.cnblogs.com>与他联系。

肖逵资深软件开发工程师，高级技术讲师，毕业后一直效力于公司核心开发部门，主持研发数个大型企业级软件产品，对.NET Framework及相关技术有深入研究，曾就职于HoneyWell、HP等公司。

王少葵，2004~2008年四届微软Visual Developer.Visual C#方面MVP，通过MCP、MCDBA、MCSA、MCAD、MCSA等多项微软认证，有10多年IT行业从业经验，现为ABB（中国）有限公司金属部高级工程师。

主要为生产制造企业提供自动控制整体解决方案，与微软、宝信、用友等软件公司有良好的合作。曾主持和参与宝钢、太钢、攀钢、酒钢、涟钢等多个企业的自动化控制、MES、ERP等若干大型系统设计与研发，涉及办公自动化、物流、数据存储、计算机通信、人机接口、模型计算等诸多领域。

活跃在CSDN社区，致力于研究.NET新技术。

范睿，（网名：fancyf）有5年.NET框架编程经验，超过6年使用C/C++的程序开发经验，对.NET框架有比较深入的研究。

经常活跃在CSDN社区，热心解答网友们的问题，致力于宣传与推广微软.NET技术的应用。

曾多次受邀进行有关.NET技术的讲座和培训。

<<C#与.NET 3.5高级程序设计>>

书籍目录

第一部分 C#和.NET平台简介第1章 NET之道1.1 NET之前的世界1.2 NET解决方案1.3 NET平台构造块(CLR、CTS和CLS)简介1.4 其他支持.NET的编程语言1.5 NET程序集概览1.6 CTS1.7 CLS1.8 CLR1.9 程序集/命名空间/类型的区别1.10 使用ildasm.exe探索程序集1.11 使用Lutz Roeder的Reflector来查看程序集1.12 部署.NET运行库1.13 NET的平台无关性1.14 小结第2章 构建C#应用程序2.1 NET Framework 3.5 SDK的作用2.2 用csc.exe构建C#应用程序2.3 使用TextPad构建.NET应用程序2.4 使用Notepad++构建.NET应用程序2.5 使用SharpDevelop构建.NET应用程序2.6 使用Visual C# 2008 Express构建.NET应用程序2.7 使用Visual Studio 2008构建.NET应用程序2.8 其他.NET开发工具2.9 小结第二部分 C#核心编程结构第3章 C#核心编程结构 3.1 一个简单的C#程序3.2 有趣的题外话: System.Environment类的其他成员3.3 System.Console类3.4 系统数据类型和C#简化符号3.5 System.String数据类型3.6 窄化和宽化数据类型转换3.7 C#迭代结构3.8 条件结构和关系/相等运算符3.9 小结第4章 C#核心编程结构 4.1 方法和参数修饰符4.2 成员重载4.3 C#中的数组操作4.4 枚举类型4.5 结构类型4.6 值类型和引用类型4.7 值类型和引用类型: 最后的细节4.8 C#可空类型4.9 小结第5章 定义封装的类类型5.1 C#类类型5.2 类构造函数5.3 this关键字的作用5.4 static关键字5.5 定义OOP的支柱5.6 C#访问修饰符5.7 第一个支柱: C#的封装支持5.8 常量数据5.9 只读字段5.10 C#的分部类型5.11 通过XML生成C#源代码的文档5.12 查看劳动成果5.13 小结第6章 继承和多态6.1 继承的基本机制6.2 回顾Visual Studio类关系图6.3 第二个支柱: 继承6.4 包含/委托编程6.5 第三个支柱: C#的多态支持6.6 基类/派生类的转换规则6.7 超级父类: System.Object6.8 小结第7章 结构化异常处理7.1 错误、bug与异常7.2 NET异常处理的作用7.3 最简单的例子7.4 配置异常的状态7.5 系统级异常(System.SystemException)7.6 应用程序级异常(System.ApplicationException) 7.7 处理多个异常7.8 finally块7.9 谁在引发什么异常7.10 未处理异常的后果7.11 使用Visual Studio调试未处理的异常7.12 小结第8章 对象的生命周期8.1 类、对象和引用8.2 对象生命周期的基础8.3 应用程序根的作用8.4 对象的代8.5 System.GC类型8.6 构建可终结对象8.7 构建可处置对象8.8 构建可终结类型和可处置类型8.9 小结第三部分 C#高级编程结构第9章 接口9.1 接口类型9.2 定义自定义接口9.3 实现接口9.4 在对象级别调用接口成员9.5 接口作为参数9.6 接口作为返回值9.7 接口类型数组9.8 使用Visual Studio 2008实现接口9.9 通过显式接口实现解决命名冲突9.10 定义接口层次结构9.11 构建可枚举类型(IEnumerable和IEnumerator)9.12 构建可克隆的对象(ICloneable)9.13 构建可比较的对象(IComparable)9.14 回调接口9.15 小结第10章 集合与泛型10.1 System.Collections命名空间的接口10.2 System.Collections命名空间的类类型10.3 System.Collections.Specialized命名空间10.4 装箱、拆箱以及和System.Object的关系10.5 类型安全和强类型集合问题10.6 System.Collections.Generic命名空间10.7 创建自定义泛型方法10.8 创建泛型结构和类10.9 创建自定义泛型集合10.10 创建泛型基类10.11 创建泛型接口10.12 小结第11章 委托、事件和Lambda11.1 NET委托类型11.2 使用C#定义委托11.3 System.MulticastDelegate与System.Delegate基类11.4 最简单的委托示例11.5 使用委托改造Car类型11.6 更复杂的委托示例11.7 委托协变11.8 创建泛型委托11.9 C#事件11.10 泛型EventHandler委托11.11 C#匿名方法11.12 方法组转换11.13 C# 3.0 Lambda运算符11.14 小结第12章 索引器、运算符和指针12.1 索引器方法12.2 运算符重载12.3 自定义类型转换12.4 指针类型12.5 C#预处理指令12.6 小结第13章 C# 3.0的语言功能13.1 隐式类型局部变量13.2 自动属性13.3 扩展方法13.4 分部方法13.5 对象初始化器13.6 匿名类型13.7 小结第14章 LINQ14.1 LINQ的作用14.2 LINQ查询表达式初览14.3 LINQ和泛型集合14.4 LINQ和非泛型集合14.5 查询运算符的内部表示14.6 LINQ查询运算符14.7 LINQ查询14.8 小结第四部分 使用.NET程序集编程第15章 .NET程序集入门15.1 定义自定义命名空间15.2 .NET程序集的作用15.3 .NET程序集的格式15.4 构建和使用单文件程序集15.5 构建和使用多文件程序集15.6 私有程序集15.7 共享程序集15.8 使用共享程序集15.9 配置共享程序集15.10 GAC的内部结构15.11 发行者策略程序集15.12 元素15.13 System.Configuration命名空间15.14 机器配置文件15.15 小结第16章 类型反射、晚期绑定和基于特性的编程16.1 类型元数据的必要性16.2 反射16.3 构建自定义的元数据查看器16.4 动态加载程序集16.5 反射共享程序集16.6 晚期绑定16.7

<<C#与.NET 3.5高级程序设计>>

特性编程16.8 构建自定义特性16.9 程序集级别(和模块级别)特性16.10 使用早期绑定反射特性16.11 使用晚期绑定反射特性16.12 反射、晚期绑定和自定义特性的使用背景16.13 构建可扩展的应用程序16.14 小结第17章 进程、应用程序域和对象上下文17.1 回顾传统的Win32进程17.2 .NET平台下与进程进行交互17.3 .NET应用程序域17.4 对象上下文边界17.5 进程、应用程序域和上下文小结17.6 小结第18章 构建多线程应用程序18.1 进程、应用程序域、上下文及线程之间的关系18.2 .NET委托的简短回顾18.3 委托的异步性18.4 异步调用方法18.5 System.Threading命名空间18.6 System.Threading.Thread类18.7 以编程方式创建次线程18.8 并发问题18.9 使用Timer Callback编程18.10 CLR线程池18.11 BackgroundWorker组件的作用18.12 小结第19章 CIL和动态程序集的作用19.1 CIL编程的本质19.2 CIL指令、特性和操作码19.3 入栈和出栈：CIL基于栈的本质19.4 正向工程19.5 CIL指令和特性19.6 .NET基类库、C#和CIL数据类型的映射19.7 在CIL中定义成员19.8 剖析CIL操作码19.9 使用CIL构建.NET程序集19.10 动态程序集19.11 小结第五部分 .NET基类库简介第20章 文件输入输出和隔离存储20.1 研究System.IO命名空间20.2 DirectoryInfo(Info)和File(Info)类型20.3 使用DirectoryInfo类型20.4 使用Directory类型20.5 使用DriveInfo类类型20.6 使用FileInfo类20.7 使用File类型20.8 Stream抽象类20.9 使用StreamWriter和StreamReader类型20.10 使用StringWriter和StringReader20.11 使用BinaryWriter和BinaryReader20.12 以编程方式“观察”文件20.13 实现异步文件I/O操作20.14 隔离存储的作用20.15 代码访问安全入门20.16 隔离存储概览20.17 使用IsolatedStorageFile获取存储20.18 实战隔离存储：ClickOnce部署20.19 小结第21章 对象序列化21.1 对象序列化21.2 为序列化配置对象21.3 选择序列化格式化程序21.4 使用BinaryFormatter序列化对象21.5 使用SoapFormatter序列化对象21.6 使用XmlSerializer序列化对象21.7 序列化对象集合21.8 自定义序列化过程21.9 小结第22章 ADO.NET之一：连接层22.1 ADO.NET高层次定义22.2 ADO.NET的数据提供程序22.3 其他的ADO.NET命名空间22.4 System.Data命名空间的类型22.5 使用接口抽象数据提供程序22.6 创建AutoLot数据库22.7 ADO.NET 数据提供程序工厂模型22.8 ADO.NET的连接式访问22.9 使用数据读取器22.10 构建可重用的数据访问库22.11 创建控制台UI前端22.12 使用SqlCommand进行异步数据访问22.13 数据库事务22.14 小结第23章 ADO.NET之二：断开连接层23.1 ADO.NET断开连接层23.2 DataSet的作用23.3 使用DataColumn23.4 使用DataRow23.5 使用DataTable23.6 将DataTable对象绑定到用户界面23.7 使用数据适配器填充DataSet/ DataTable23.8 重访AutoLotDAL.dll23.9 切换多表DataSet对象23.10 Visual Studio 2008的数据访问工具23.11 从UI层解耦自动生成的代码23.12 小结第24章 LINQ API编程24.1 LINQ to ADO.NET的作用24.2 使用LINQ to DataSet编程24.3 使用LINQ to SQL编程24.4 使用sqlmetal.exe生成实体类24.5 使用Visual Studio 2008建立实体类24.6 使用LINQ to XML操作XML文档24.7 在内存文档中导航24.8 小结第25章 WCF25.1 各种分布式计算API25.2 WCF的作用25.3 WCF核心程序集25.4 Visual Studio WCF项目模板25.5 WCF应用程序的基本构成25.6 WCF的ABC25.7 构建WCF服务25.8 承载WCF服务25.9 构建WCF客户端应用程序25.10 使用WCF服务库项目模板25.11 以Windows服务承载WCF服务25.12 异步调用服务25.13 定义WCF数据契约25.14 小结第26章 WF26.1 定义业务流程26.2 WF的构建块26.3 WF程序集、命名空间和项目26.4 构建一个启用工作流的简单应用26.5 WF引擎承载代码26.6 在工作流中调用Web服务26.7 构建可重用的WF代码库26.8 关于自定义活动的简要说明26.9 小结第六部分 桌面用户界面第27章 Windows Forms编程27.1 Windows Forms 命名空间27.2 创建一个简单的Windows Forms 程序(不用IDE)27.3 Visual Studio Windows Forms项目模板27.4 剖析Form27.5 响应鼠标活动27.6 响应键盘活动27.7 设计对话框27.8 通过GDI+呈现图形数据27.9 创建一个完整的Windows Forms应用程序27.10 小结第28章 WPF和XAML28.1 WPF背后的动机28.2 各种形式的WPF应用程序28.3 WPF程序集28.4 创建(不使用XAML的)WPF应用程序28.5 Application类型的其他细节28.6 Window类型的其他细节28.7 构建(XAML相关的)WPF应用程序28.8 将标记转换为.NET程序集28.9 使用代码隐藏文件实现的关注点的分离28.10 XAML语法28.11 使用Visual Studio 2008构建WPF应用程序28.12 在运行时处理XAML：SimpleXaml- Pad.exe28.13 微软Expression Blend的作用28.14 小结第29章 使用WPF控件编程29.1 WPF控件库概述29.2 使用XAML声明控件29.3 依赖属性的作用29.4 路由事件29.5 使用Button类型29.6 使用CheckBox和RadioButton29.7 使用ListBox和ComboBox类型29.8 使

<<C#与.NET 3.5高级程序设计>>

用文本区29.9 使用面板进行内容布局29.10 使用嵌套面板创建窗口框架29.11 WPF控件命令29.12 WPF数据绑定模型29.13 使用IValueConverter进行数据转换29.14 绑定到自定义对象29.15 将UI元素绑定到XML文档29.16 小结第30章 WPF 2D图形呈现、资源和主题30.1 WPF图形呈现服务30.2 探究Shape派生类30.3 使用WPF画刷30.4 使用WPF画笔30.5 探究Drawing派生类30.6 UI变换的功能30.7 WPF动画服务30.8 WPF的资源体系30.9 为WPF控件定义应用样式30.10 使用模板改变控件的UI30.11 小结第七部分 使用ASP.NET构建Web应用程序第31章 构建ASP.NET网页31.1 HTTP的作用31.2 Web应用程序和Web服务31.3 HTML的作用31.4 客户端脚本的作用31.5 提交表单数据(GET和POST)31.6 构建传统的ASP页面31.7 传统ASP相关问题31.8 ASP.NET命名空间31.9 ASP.NET网页代码模型31.10 ASP.NET站点目录结构细节31.11 ASP.NET页面编译周期31.12 页面类型的继承链31.13 与传入的HTTP请求交互31.14 与输出HTTP响应交互31.15 ASP.NET网页的生命周期31.16 Web.config文件的作用31.17 小结第32章 ASP.NET Web控件、主题和母版页32.1 Web控件的本质32.2 System.Web.UI.Control类型32.3 System.Web.UI.WebControls.WebControl类型32.4 ASP.NET Web控件的类别32.5 构建功能丰富的ASP.NET站点32.6 验证控件的作用32.7 使用主题32.8 使用HTML表格布局控件32.9 小结第33章 ASP.NET状态管理技术33.1 状态问题33.2 ASP.NET状态管理技术33.3 ASP.NET视图状态的作用33.4 Global.asax文件的作用33.5 应用程序状态与会话状态差别33.6 使用应用程序缓存33.7 维护会话数据33.8 cookie33.9 元素的作用33.10 ASP.NET用户配置API33.11 小结第八部分 附录附录A COM与.NET的互操作性A.1 NET互操作的研究范围A.2 NET调用COM互操作的简单示例A.3 NET互操作程序集A.4 运行库可调用包装A.5 COM IDL的作用A.6 使用类型库创建互操作程序集A.7 创建一个更复杂的COM服务A.8 互操作程序集A.9 在COM中使用.NETA.10 CCW的作用A.11 NET类接口的作用A.12 构建.NET类型A.13 生成类型库并注册.NET类型A.14 导出类型信息A.15 创建一个Visual Basic 6.0的测试客户端A.16 小结附录B 使用Mono进行平台无关的.NET开发B.1 NET的平台无关性B.2 获取和安装MonoB.3 Mono开发工具B.4 使用Mono创建.NET应用程序B.5 推荐学习B.6 小结索引

<<C#与.NET 3.5高级程序设计>>

章节摘录

当初步了解.NET的语言无关性后，开发者会提出许多问题。

其中最普遍的问题可能就是：“如果所有的.NET语言都会编译成‘托管代码’，为什么我们还需要多种编译器呢？”

”这个问题有多个答案。

首先，程序员在选择编程语言时有各自不同的喜好（包括我自己）。

一些人喜欢充满分号和圆括号而且关键字相当少的语言；另一些人喜欢更具有“可读性”语法标记的语言（如EIVB）；还有一些人在开始转向.NET平台时还希望可以使用他们已掌握的技能（通过COBOL.NET）。

现在，平心而论，如果微软推出一门派生自BASIC语言系列的“官方”.NET语言，你认为所有的程序员会喜欢这样的选择吗？

或者，如果这个唯一的“官方”.NET语言是基于Fortran语法的，那么可以想象所有人都会对.NET置之不理。

因为.NET运行库并不在意一段托管代码是由哪种语言生成的，所以.NET程序员可以继续使用他们熟悉的语法，且与组员、部门甚至其他公司共享编译的程序集（不管他们用的是哪种.NET语言）。

将各种.NET语言集成为一个统一软件方案的另一个好处，就是能够取长补短。

所有的编程语言都有各自的优点和缺点。

例如，一些编程语言对高级的数学处理有相当完美的内在支持能力。

另一些则精于支持财务计算、逻辑计算和与大型机交互等。

当你学习到某种编程语言的优点并将其融合于.NET平台时，大家就都能受益。

当然，实际上我们大部分时间还是在用自己习惯的.NET语言来编写程序。

但是，一旦学会了一种.NET语言的语法，就很容易掌握其他的了。

这是非常有益的，对软件技术顾问而言尤其如此。

如果你熟悉C#，在为只使用Visual Basic.NET的客户做咨询时，你仍然能够使用.NET Framework的功能，并且可以毫不费力地掌握代码的整体结构。

够棒的吧。

1.5.NET程序集概览不管选择了哪种.NET语言编程，需要明白的是，尽管.NET二进制文件COM服务器和非托管Win32二进制文件（*.dll或*.exe）具有相同的文件扩展名，但它们的内部却是完全不同的。

例如，*.dll的.NET二进制文件不会导出与COM运行库进行通信的方法（因为.NET不是COM）。

此外，.NET二进制文件不使用COM类型库文件描述而且不用在系统注册表中注册。

也许更重要的是，.NET二进制文件不包含特定于平台的指令，它包含的是平台无关的IL（中间语言）和类型元数据。

图1.3清楚显示了这个流程。

插图：

<<C#与.NET 3.5高级程序设计>>

编辑推荐

《C#与.NET 3.5高级程序设计》(第4版)由微软C# MVP Andrew Troelsen编写, 历经多次修订, 适合各层次.NET开发人员阅读。

Amazon超级畅销书, 全面涵盖.c#3.0, 用IL深入揭示各语言特性, 深度剖析.NET 3.5平台。

让你知其然, 更知其所以然, 国内多位微软MVP联手翻译。

C#语言作为.NET平台上的第一语言, 已经成为目前功能最强大的通用语言之一。

《C#与.NET 3.5高级程序设计(第4版)》是被誉为“C样圣经”的经典巨著, 因语言生动流畅、剖析深入、涵盖全面而广受推崇。

畅销不衰。

曾经获得Referenceware编程图书大奖。

并入选Jolt大奖提名。

书中探讨了C#语言和.NET平台的各种特性。

包括重载运算符、指针、泛型等高级功能和Clt、远程处理、Windows Forms、ASP.NET、ADO.NET等技术。

不少概念都通过IL代码透视其背后的本质, 使你知其然。

更知其所以然。

新版更透彻阐述了C#3.0新功能(包括自动属性、扩展方法、匿名类型等)和.NET 3.5的最新特性(包括LINQ、WPF、WCF和WF等相关技术)。

附录中包括了COM与.NET的互操作和Mono发等主题。

与同类图书不同。

全书由世界级C#专家Andrew Troelsen以一人之力完成, 因此写作思路和布局谋篇都独具匠心。

中文版由国内多位微软MVP联手译出。

强大的译者阵容有力地保证了这部权威原著原汁原味的重现。

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>