

图书基本信息

书名：<<C#与.NET 4高级程序设计 (第5版)>>

13位ISBN编号：9787115250322

10位ISBN编号：7115250324

出版时间：2011-4

出版单位：人民邮电出版社

作者：[美] Andrew Troelsen

页数：1197

译者：朱晔 [等]

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>

## 内容概要

本书是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、.NET 3.5

和.NET 4中的新的编程API，包括WPF、WCF和WF的功能；另外，还介绍了最新的C# 2010编程语言、DLR、TPL、LINQ编程技术、COM与.NET的互操作性以及平台无关的.NET开发等。

本书由微软C# MVP Andrew Troelsen编写，历经多次修订，适合各层次.NET开发人员阅读。

## 作者简介

Andrew 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。

## 书籍目录

## 第一部分 C#与.NET平台

## 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 CTS
- 1.7 CLS
- 1.8 CLR
- 1.9 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 4 SDK的作用
- 2.2 用csc.exe构建C#应用程序
- 2.3 使用Notepad++构建.NET应用程序
- 2.4 使用SharpDevelop构建.NET应用程序
- 2.5 使用Visual C# 2010 Express构建.NET应用程序
- 2.6 使用Visual Studio 2010构建.NET应用程序
- 2.7 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 使用字符串数据
- 3.6 窄化和宽化数据类型转换
- 3.7 隐式类型本地变量
- 3.8 C#迭代结构
- 3.9 条件结构和关系/相等操作符
- 3.10 小结

## 第4章 C#核心编程结构

- 4.1 方法和参数修饰符
- 4.2 C#中的数组操作
- 4.3 枚举类型
- 4.4 结构类型
- 4.5 值类型和引用类型
- 4.6 C#可空类型
- 4.7 小?

## 第5章 定义封装的类类型

- 5.1 C#类类型
- 5.2 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#的封装服务
- 5.8 自动属性
- 5.9 对象初始化器语法
- 5.10 常量数据
- 5.11 分部类型
- 5.12 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 OOP的第二个支柱：继承
  - 6.4 包含/委托编程
  - 6.5 OOP的第三个支柱：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.Object
  - 6.8 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.System-Exception )
  - 7.6 应用程序级异常 ( System.Applica-tionException )
  - 7.7 处理多个异常
  - 7.8 谁在引发什么异常
  - 7.9 未处理异常的后果
  - 7.10 使用Visual Studio调试未处理的异常
  - 7.11 损坏状态异常简介
  - 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 .NET 1.0 至.NET 3.5的并发垃圾回收
  - 8.6 .NET 4下的后台垃圾回收
  - 8.7 System.GC类型
  - 8.8 构建可终结对象
  - 8.9 构建可处置对象
  - 8.10 构建可终结类型和可处置类型
  - 8.11 延迟对象实例化
  - 8.12 小结

## 第三部分 高级C#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 2010实现接口
- 9.9 通过显式接口实现解决命名冲突
- 9.10 设计接口层次结构
- 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 小结

## 第10章 泛型

- 10.1 非泛型集合
- 10.2 泛型类型参数的作用
- 10.3 System.Collections.Generic命名空间
- 10.4 创建自定义泛型方法
- 10.5 创建自定义泛型结构和类
- 10.6 类型参数的约束
- 10.7 小结

## 第11章 委托、事件和Lambda

- 11.1 .NET委托类型
- 11.2 在C#中定义委托类型
- 11.3 System.MulticastDelegate与System.Delegate基类
- 11.4 最简单的委托示例
- 11.5 使用委托发送对象状态通知
- 11.6 方法组转换语法
- 11.7 委托协变
- 11.8 泛型委托
- 11.9 C#事件
- 11.10 C#匿名方法
- 11.11 Lambda表达式
- 11.12 小结

## 第12章 高级C#语言特性

- 12.1 索引器方法
- 12.2 操作符重载
- 12.3 自定义类型转换
- 12.4 扩展方法
- 12.5 分部方法
- 12.6 匿名类型
- 12.7 指针类型
- 12.8 小结

## 第13章 LINQ to Object

- 13.1 LINQ特有的编程结构
- 13.2 LINQ的作用
- 13.3 将LINQ查询应用于原始数组
- 13.4 返回LINQ查询的结果
- 13.5 将LINQ查询应用到集合对象
- 13.6 C#LINQ查询操作符
- 13.7 LINQ查询语句的内部表示
- 13.8 小结
- 第四部分 用.NET程序集编程
- 第14章 .NET程序集入门
  - 14.1 定义自定义命名空间
  - 14.2 .NET程序集的作用
  - 14.3 .NET程序集的格式
  - 14.4 构建和使用单文件程序集
  - 14.5 构建和使用多文件程序集
  - 14.6 私有程序集
  - 14.7 共享程序集
  - 14.8 使用共享程序集
  - 14.9 配置共享程序集
  - 14.10 发行者策略程序集
  - 14.11 元素
  - 14.12 System.Configuration命名空间
  - 14.13 小结
- 第15章 类型反射、晚期绑定和基于特性的编程
  - 15.1 类型元数据的必要性
  - 15.2 反射
  - 15.3 构建自定义的元数据查看器
  - 15.4 动态加载程序集
  - 15.5 反射共享程序集
  - 15.6 晚期绑定
  - 15.7 .NET特性的作用
  - 15.8 构建自定义特性
  - 15.9 程序集级别 (和模块级别) 特性
  - 15.10 使用早期绑定反射特性
  - 15.11 使用晚期绑定反射特性
  - 15.12 反射、晚期绑定和自定义特性的使用背景
  - 15.13 构建可扩展的应用程序
  - 15.14 小结
- 第16章 进程、应用程序域和对象上下文
  - 16.1 Windows进程的作用
  - 16.2 .NET平台下与进程进行交互
  - 16.3 .NET应用程序域
  - 16.4 与默认应用程序域进行交互?
  - 16.5 创建新的应用程序域
  - 16.6 对象上下文边界
  - 16.7 进程、应用程序域和上下文小结
  - 16.8 小结

- 第17章 CIL和动态程序集的作用
  - 17.1 学习CIL语法的原因
  - 17.2 CIL指令、特性和操作码
  - 17.3 入栈和出栈：CIL基于栈的本质
  - 17.4 正反向工程
  - 17.5 CIL指令和特性
  - 17.6 .NET基础类库、C#和CIL数据类型的映射
  - 17.7 在CIL中定义类型成员
  - 17.8 剖析CIL操作码
  - 17.9 使用CIL构建.NET程序集
  - 17.10 动态程序集
  - 17.11 小结
- 第18章 动态类型和动态语言运行时
  - 18.1 dynamic关键字的作用
  - 18.2 DLR的作用
  - 18.3 使用动态类型简化后期绑定调用
  - 18.4 使用动态数据简化COM互操作
  - 18.5 使用C# 2010的特性进行COM互操作
  - 18.6 小结
- 第五部分 .NET基础类库
  - 第19章 构建多线程应用程序
    - 19.1 进程、应用程序域、上下文及线程之间的关系
    - 19.2 .NET委托的简短回顾
    - 19.3 委托的异步性
    - 19.4 异步调用方法
    - 19.5 System.Threading命名空间
    - 19.6 System.Threading.Thread类
    - 19.7 以编程方式创建次线程
    - 19.8 并发问题
    - 19.9 使用Timer Callback编程
    - 19.10 CLR线程池
    - 19.11 .NET平台下的并行编程
    - 19.12 并行LINQ查询 (PLINQ)
    - 19.13 小结
  - 第20章 文件输入输出和对象序列化
    - 20.1 研究System.IO命名空间
    - 20.2 Directory(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StringReader类型
    - 20.11 使用BinaryWriter和Binary-Reader
    - 20.12 以编程方式“观察”文件



- 20.13 对象序列化
- 20.14 为序列化配置对象
- 20.15 选择序列化格式化程序
- 20.16 使用BinaryFormatter序列化对象
- 20.17 使用SoapFormatter序列化对象
- 20.18 使用XmlSerializer序列化对象
- 20.19 序列化对象集合
- 20.20 自定义Soap/Binary序列化过程
- 20.21 小结
- 第21章 ADO.NET之一：连接层
  - 21.1 ADO.NET的宏观定义
  - 21.2 ADO.NET数据提供程序
  - 21.3 其他的ADO.NET命名空间
  - 21.4 System.Data命名空间的类型
  - 21.5 使用接口的抽象数据提供程序
  - 21.6 创建AutoLot数据库
  - 21.7 ADO.NET数据提供程序工厂模型
  - 21.8 ADO.NET的连接层
  - 21.9 使用数据读取器
  - 21.10 构建可重用的数据访问库
  - 21.11 创建控制台UI前端
  - 21.12 数据库事务
  - 21.13 小结
- 第22章 ADO.NET之二：断开连接层
  - 22.1 ADO.NET断开连接层
  - 22.2 DataSet的作用
  - 22.3 使用DataColumn
  - 22.4 使用DataRow
  - 22.5 使用DataTable
  - 22.6 将DataTable对象绑定到用户界面
  - 22.7 使用数据适配器
  - 22.8 向AutoLotDAL.dll添加断开连接功能
  - 22.9 多表DataSet对象和数据关系
  - 22.10 Windows Forms数据库设计器工具
  - 22.11 将强类型的数据库代码隔离到类库中
  - 22.12 LINQ to DataSet
  - 22.13 小结
- 第23章 ADO.NET之三：Entity Framework
  - 23.1 Entity Framework的作用
  - 23.2 创建和分析EDM
  - 23.3 对概念模型进行编程
  - 23.4 AutoLotDAL 4.0版，加入实体
  - 23.5 将数据实体绑定到Windows Forms GUI
  - 23.6 小结
- 第24章 LINQ to XML简介
  - 24.1 两个XML API的故事
  - 24.2 System.Xml.Linq命名空间的成员

- 24.3 使用XElement和XDocument
- 24.4 在内存中操作XML文档
- 24.5 小结
- 第25章 WCF
  - 25.1 各种分布式计算API
  - 25.2 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ABC
  - 25.7 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 用WCF 4.0简化配置设置
  - 25.11 使用WCF服务库项目模板
  - 25.12 以Windows服务承载WCF服务
  - 25.13 从客户端异步调用服务
  - 25.14 定义WCF数据契约
  - 25.15 小结
- 第26章 Windows Workflow Foundation
  - 4.0简介
  - 26.1 定义业务流程
  - 26.2 构建简单的工作流
  - 26.3 WF 4.0运行时
  - 26.4 检查WF 4.0中的活动
  - 26.5 构建流程图工作流
  - 26.6 用专门的库隔离工作流
  - 26.7 使用工作流库
  - 26.8 小结
- 第六部分 使用WPF构建桌面用户界面
- 第27章 WPF和XAML
  - 27.1 WPF背后的动机
  - 27.2 各种形式的WPF应用程序
  - 27.3 WPF程序集
  - 27.4 创建不使用XAML的WPF应用程序
  - 27.5 仅使用XAML构建WPF应用程序
  - 27.6 将标记转换为.NET程序集
  - 27.7 WPF XAML语法
  - 27.8 使用代码隐藏文件构建WPF应用程序
  - 27.9 使用Visual Studio 2010构建WPF应用程序
  - 27.10 小结
- 第28章 使用WPF控件编程
  - 28.1 WPF核心控件概述
  - 28.2 使用面板控制内容布局
  - 28.3 使用嵌套面板构建窗口框架
  - 28.4 WPF控件命令
  - 28.5 使用Expression Blend构建WPF用户界面

- 28.6 构建Ink API选项卡
- 28.7 Documents API
- 28.8 构建Documents选项卡
- 28.9 WPF数据绑定模型
- 28.10 小结
- 第29章 WPF图形呈现服务
  - 29.1 理解WPF的图形呈现服务
  - 29.2 使用形状呈现图形数据
  - 29.3 WPF画刷和画笔
  - 29.4 图形变换
  - 29.5 使用Expression Blend处理形状
  - 29.6 使用绘图和几何图形呈现图形数据
  - 29.7 使用Expression Design生成复杂的向量图形
  - 29.8 使用可视化层呈现图形数据
  - 29.9 小结
- 第30章 WPF资源、动画和样式
  - 30.1 理解WPF资源系统
  - 30.2 使用对象（逻辑）资源
  - 30.3 理解WPF动画服务
  - 30.4 用XAML创建动画
  - 30.5 WPF样式的作用
  - 30.6 使用Expression Blend生成样式
  - 30.7 小结
- 第31章 WPF控件模板和用户控件
  - 31.1 依赖属性的作用
  - 31.2 构建自定义依赖属性
  - 31.3 路由事件
  - 31.4 逻辑树、可视树和默认模板
  - 31.5 在Visual Studio 2010中构建自定义控件模板
  - 31.6 使用Blend构建自定义UserControl
  - 31.7 创建WPF应用程序Jackpot Deluxe
  - 31.8 小结
- 第七部分 使用ASP.NET构建Web应用程序
- 第32章 构建ASP.NET网页
  - 32.1 HTTP的作用
  - 32.2 Web应用程序和Web服务
  - 32.3 HTML的作用
  - 32.4 客户端脚本的作用
  - 32.5 回发到Web服务器
  - 32.6 ASP.NET API的特性
  - 32.7 构建单个文件的ASP.NET网页
  - 32.8 使用代码文件构建ASP.NET Web页面
  - 32.9 ASP.NET Web Site和ASP.NET Web Application 0
  - 32.10 ASP.NET网站目录结构
  - 32.11 页面类型的继承链
  - 32.12 与传入的HTTP请求交互
  - 32.13 与输出HTTP响应交互

- 32.14 ASP.NET网页的生命周期
- 32.15 Web.config文件的作用
- 32.16 小结
- 第33章 ASP.NET Web控件、母版页和主题
  - 33.1 Web控件的本质
  - 33.2 Control和WebControl基类
  - 33.3 ASP.NET Web控件的类别
  - 33.4 构建ASP.NET汽车网站
  - 33.5 验证控件的作用
  - 33.6 使用主题
  - 33.7 小结
- 第34章 ASP.NET状态管理技术
  - 34.1 状态问题
  - 34.2 ASP.NET状态管理技术
  - 34.3 ASP.NET视图状态的作用
  - 34.4 Global.asax文件的作用
  - 34.5 应用程序状态与会话状态的差别
  - 34.6 使用应用程序缓存
  - 34.7 维护会话数据
  - 34.8 cookie
  - 34.9 元素的作用
  - 34.10 ASP.NET用户配置API
  - 34.11 小结
- 第八部分 附 录
  - 附录A Windows Forms编程
    - A.1 Windows Forms命名空间
    - A.2 创建一个简单的Windows Forms程序
    - A.3 Visual Studio Windows Forms项目模板
    - A.4 剖析Form
    - A.5 响应鼠标活动
    - A.6 设计对话框
    - A.7 通过GDI+呈现图形数据
    - A.8 创建一个完整的Windows Forms应用程序
    - A.9 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Mono
    - B.3 Mono开发语言
    - B.4 兼容Microsoft的Mono开发工具
    - B.5 使用Mono创建.NET 应用程序
    - B.6 谁在使用Mono
    - B.7 推荐学习
    - B.8 小结

#### 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>