

Visual C# 2008

开发技术实例详解

知程序 行天下

行天下

知程序

李继攀 黄国平等编著

程序天下



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

400多个实例程序源代码 + 5个典型系统
+ 10个学习视频 + 600多页学习电子书



Visual C# 2008

开发技术实例详解

知程序 行天下 行天下 知程序

李继攀 黄国平等编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING



程序天下

内 容 简 介

本书是一本讲解 C# 编程语言实例程序的书。本书以“技术要点”、“实现步骤”和“源程序解读”的方式，向读者提供大量独立的、完整的 C# 语言示例程序。

本书在巩固 C# 编程语言示例程序的基础上，遍及 C# 编程语言的各种实用技术，为各个主题的 C# 应用程序提供直接的实战指导和实现方式，内容涵盖 C# 编程的各个方面，包括了现今高级的编程技术应用。这些实例具有极强的针对性和实用性，直指 C# 编程技术的核心内容，揭示其中关键。通过这些生动的实例指导，读者能够按图索骥，由点到面地将学习的编程知识有效地用于实际开发，充分体会到 C# 编程的乐趣与奥妙。

本书的示例程序包含编者多年的研究和实践心得，可使读者全面地、透彻地理解 C# 编程的核心技术和技巧，也可以直接修改书中的实例用于实际开发，对提高 C# 编程实践能力有极大的帮助。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

Visual C# 2008 开发技术实例详解 / 李继攀等编著. —北京: 电子工业出版社, 2008.8
(程序天下)
ISBN 978-7-121-06855-3

I. V… II. 李… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2008) 第 081701 号

责任编辑: 王鹤扬

印 刷: 北京智力达印刷有限公司

装 订: 三河市皇庄路通装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 51.75 字数: 1166 千字

印 次: 2008 年 8 月第 1 次印刷

印 数: 5000 册 定价: 89.00 元 (含光盘 1 张)

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlt@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前 言

C#编程语言是.NET 语言的主打产品，Microsoft 提供该语言来开发各种各样的应用程序和组件。C#语言是真正的面向对象的编程语言，在.NET Framework 强有力的支持下，开发人员能够使用 C#语言便捷地编写各类应用程序及组件。从最简单的“Hello World”程序，到最复杂的企业应用程序，都可以用 C#编程语言实现。

编写本书的目的，是为了向读者全面介绍 C#编程实战技巧。C#编程语言的应用非常广泛，在软件行业的各个应用领域中，C#编程语言已经成为基于.NET Framework 解决方案的首选语言。本书精心地组织了 C#编程语言在各个应用领域中的典型示例，演示 C#编程语言所涉及的各种示例程序，揭示 C#编程语言的核心技术，展示 C#编程语言思想的实际应用技巧，力图使读者得以窥一斑而见全豹，举一反三，借此领悟 C#编程语言的奥妙。

本书准确地向读者演示实现示例的每个步骤，读者能够按部就班地逐步构建 C#示例程序。随着每个示例程序的编译运行，辅以本书简明精练的说明和解读，使读者把握实际开发所需要的技术要点，并能立即应用到实际开发中。

本书的内容安排

本书共分为七篇，全面涉及 C#编程的各种实际应用技巧，以下是各篇具体内容的简单介绍。

第一篇（第 1 章～第 7 章）的示例程序旨在点明 C#编程必需但仅此就足够的语法基础。C#语言在设计时借鉴了多种编程语言的设计，比起其他编程语言，虽然 C#语言的语法相对简单，比较容易掌握，但是也会使一些重要的语法容易被开发人员忽略，在实际开发过程中，往往存在着许多因为对语法理解的偏差而造成的错误。本篇摒弃枯燥的说教方式，代之以生动的示例程序，深入说明 C#语法的重点内容，使读者能够迅速掌握和强化 C#编程开发的基础语法。

第二篇（第 8 章～第 11 章）的示例程序展现了 C#编程语言面向对象的特征。C#编程语言不仅继承了传统面向对象的概念，还在此基础上做了大量扩展，这些扩展使 C#语言的面向对象编程功能独树一帜，成为 C#编程中极为重要且广泛应用的技术。本篇对 C#编程语言的面向对象概念及其实现方法作系统介绍，并凸显了 C#编程语言在面向对象编程时的独特之处。读者通过这些示例能够把握面向对象的各種概念及其在 C#编程语言中的实现方法。

第三篇（第 12 章～第 18 章）的示例程序涉及了 C#编程语言在各个应用领域广泛应用的高级应用技术。为了满足开发各种应用程序的需要，C#编程语言基于.NET Framework 提供的框架支持，借助语法设计和面向对象特征的优势，囊括了当今应用程序开发最流行、也最重要的技术，提供了各种功能强大的开发模型、模式和方法。这些开发模型、模式和

方法,对于开发人员而言是必须全面深入掌握的。本篇紧密结合实践,通过精挑细选的示例程序,简明扼要地介绍开发 C#应用程序必备的高级技术,针对实际开发时容易造成困惑的难点做深入剖析,直击各种应用技术的核心要害,并列举多种途径的实现方式及甄选场合所应考虑的因素。读者将通过这些示例程序精通 C#编程实践的多方面技术,在解决所遇到的具体技术问题时有胸有成竹,游刃有余。

第四篇(第 19 章~第 26 章)的示例程序涵盖了 ASP.NET 开发的各个方面。随着 Internet 的快速发展,大量的应用程序从传统的 Win32 平台移植到 Internet 平台,Web 开发已不可抗拒地成为当今应用程序开发的主流。在多种需求相继提出、各种 Web 开发技术应运而生的情况下,C#编程语言凭借得天独厚的技术优势迅速成为开发 Web 应用的首选。Web 开发最显著的特点是一个 Web 应用往往集成了许多形形色色的技术手段,从页面交互实现到后台的数据服务,都包含有大量各自领域的多项技术。本篇针对 Web 开发的这些特点,基于 ASP.NET 框架,通过大量的 Web 应用程序示例,全方位、多视角地切入 ASP.NET 的开发实践中,从一脉相承的传统 ASP 六大对象开始,一直到最新的 AJAX 编程技术,本篇的示例程序都提供了全面的实践指导和精准的阐述,不仅面面俱到地列举示例程序,还在演示过程中力求突出重点难点,以指点迷津,帮助读者摆脱 Web 开发扑朔迷离的困境,走上 ASP.NET 开发的康庄大道。

第五篇(第 27 章~第 30 章)的示例程序列举了 Windows 应用程序开发的所有界面要素的使用方法。Windows 应用程序具有即时响应、操作方便的交互性能,因为 Windows 应用程序能够方便地实现优秀的交互性能,长期拥有大量用户群,所以在未来较长的一段时期内 Windows 应用程序在许多应用场合的地位仍然无可替代。.NET Framework 中提供了大量的可视化控件和丰富的界面要素,以支持 Windows 应用程序的快速开发。本章针对 Windows 应用程序开发的特点,介绍窗口、控件、对话框和图形等 Windows 编程的界面要素,在程序中实现灵活的控制方式和友好的交互界面,这些示例程序蕴涵丰富的编程经验,包含了许多巧妙的编排、控制技术,使 Windows 应用程序的各项技术得到了合理应用。读者能够通过这些示例程序深刻理解 Windows 程序编程的要领,构建灵活、丰富的用户交互界面,实现复杂、强大的程序功能。

第六篇(第 31 章~第 35 章)的示例程序集成了 .NET 开发的重要特性。这些特性为 .NET Framework 所独有,为了实现应用程序之间的交互,.NET Framework 提供一系列的功能支持,包括托管代码与托管代码之间、托管代码与非托管代码之间以及托管代码和系统对象之间的交互功能,这些功能实现了在应用程序之间进行消息传递和消息处理,还提供了对注册表、COM 组件和本地化等系统对象的操作功能。编写良好的和各种对象交互的托管代码的关键,在于了解 .NET Framework 与这些程序和对象交互的机制和基本原理。本篇在介绍这些 .NET Framework 框架的交互特性时,透彻精到地阐述了托管代码交互的原理和机制。读者能够通过这些示例程序见微知著,掌握 .NET Framework 的重要特征,在编写托管代码访问其他外部对象的时候也能够触类旁通,顺利解决难题。

第七篇(第 36 章~第 40 章)的示例程序演示了 .NET 的其他应用。包括 C#编程语言构建的各种应用程序,在 .NET Framework 中除了提供对 Windows 应用程序和 ASP.NET 应用程序的支持,还提供了其他应用程序的项目模板。使用这些模板,C#编程语言可以构建

丰富多彩的各种应用程序。本篇的示例程序讲解控制台应用、Web 服务、Windows 服务和 Remoting 分布式应用以及 Office 应用的构建方法，这些应用使用的技术都是实际编程中需要熟悉和掌握的，有些应用程序已经成为 C# 程序中不可或缺的部分（例如 Web 服务应用）和未来的发展方向（例如 Office 应用）。读者从这些示例程序中能够充分领略到 C# 编程语言在各种应用中的强大功能，并能依据本篇所介绍的示例编写出自己的各种应用程序，成为 C# 编程的多面手。

本书以 C# 编程语言的示例程序为主线，按照相应主题组织各个章节，但每个示例程序均可独立成篇，每个示例程序均可独立编译运行。读者可以从任意的章节开始阅读，按照本书说明的步骤，迅速掌握需要学习的知识。

本书的特点

本书以实例讲解的方式提供 C# 编程的实践指导。内容涵盖 C# 编程的各个方面，不仅涵盖了全面的基础知识应用，还包括了现今高级的编程技术应用，这些实例具有极强的针对性和实用性，直指 C# 编程技术的核心内容，揭示其中关键。每个实例的重点在“技术要点”部分加以提纲挈领地说明，难点在“源程序解读”部分加以详细阐述，通过这些生动的实例指导，读者能够按图索骥，由点到面地将学习的编程知识有效地用于实际开发，达到窥一斑而见全豹的效果。本书主要的特点如下。

- 为编程人员提供实用的编程指导。本书以 C# 编程的实战技术为线索，精心组织 400 多个编程人员需要学、看得懂、用得上的实例，辅以精辟的要点、难点说明，对于复杂的程序以流程图表示程序的结构，使读者不仅能真正地解决实际编程的问题，类似的问题也能迎刃而解。
- 具有独立的可操作性。本书的每个实例都独立成篇，章节之间不存在纠缠不清的瓜葛。读者仅需要了解某个专题的知识时，没有必要从别的章节开始输入代码。本书的结构采用各个例子独立的形式，以专题为章节组织起来，读者只需要把精力放在一个小节中，按照实例的步骤操作，就能够立即编译运行，学习起来轻松方便。
- 具有极高的使用价值。本书将抽象的理论与实际应用紧密结合，通过应用在各个实际项目中的实例，充分体现 C# 编程技术的特点。所有实例均由资深的编程人员编写，其中缜密的源代码组织、简练的功能实现、深入独到的见解都极具参考价值。
- 本书的学习曲线极为平缓。不使用传统的平面的学习方式，而使用立体的学习方式，直接从实例的学习来把握理论知识，使读者获得强烈的成就感和趣味性。

本书的学习目标简要

- 如何熟练掌握 C# 编程语言的基础语法。
- 如何自如地使用 Visual Studio 2008 开发环境。

- 如何使用 C#编程语言开发完全面向对象（OO）的应用程序。
- 如何使用 C#编程语言结合 XML 实现轻量数据的读取。
- 如何使用 ADO.NET 实现程序与数据的轻松查询。
- 如何连接数据库并在网页中浏览和维护数据。
- 如何使用功能强大的 GridView 控件。
- 如何使用验证控件实现数据验证功能。
- 如何使用 ASP.NET AJAX 实现页面局部刷新和异步更新。
- 如何创建和使用外观、主题、母版页以及 Web 部件。
- 如何使用导航控件为网站添加导航功能。
- 如何使用登录控件实现用户权限管理。
- 如何创建和使用 Web Service。
- 如何维护状态和缓存。
- 如何使用 C#编程语言开发多线程、COM 等高级技术。
- 如何与 Office 轻松交互，实现对 Word、Excel 的轻松操作。

适合阅读本书的读者

- C#编程语言的初学者。
- 从其他编程语言转型到 C#编程语言的程序员。
- 使用 C#编程语言开发应用程序的程序员。
- C#应用程序测试人员。
- ASP.NET 网页设计人员。
- 数据库设计人员和管理人员。
- 项目技术选型人员。
- 其他需要了解 C#编程技术的相关人员。
- C#编程语言的底层开发人员。

光盘内容介绍

为了方便读者学习本书，本书附带超值光盘 1 张，内容为本书源程序和作者免费赠送的学习资料（659 页电子书、10 个学习视频和 5 个应用系统）。这些免费赠送的学习资料如下：

- 182 页 ASP.NET 网络开发技术电子书；
- 199 页 Visual C#开发技术电子书；
- 154 页 .NET 数据库开发技术电子书；
- 124 页 ASP.NET AJAX 开发电子书；

- 4个.NET 2.0 入门视频;
- 6个.NET 3.5 入门视频;
- 5个常见的典型系统。

一、源程序

主要包括书中涉及的所有源程序文件，全部按照章节整理。

二、免费赠送的学习资料

1. Visual C# 2005 开发技术电子书（199 页）

- 第 1 章 C#与.NET Framework 简介
- 第 2 章 C#程序设计基础
- 第 3 章 与用户交互
- 第 4 章 常用控件
- 第 5 章 类
- 第 6 章 抽象类、多态及接口
- 第 7 章 处理异常和错误
- 第 8 章 XML 编程
- 第 9 章 应用程序部署
- 第 10 章 时间跟踪系统

2. .NET 数据库开发技术电子书（154 页）

- 第 1 章 操作数据库
- 第 2 章 使用数据库
- 第 3 章 使用数据绑定和 DataSet
- 第 4 章 SQL Server XML 的功能
- 第 5 章 XML 编程
- 第 6 章 数据映射方案
- 第 7 章 NHibernate 事务、并发和缓存
- 第 8 章 NHibernate 高级数据映射和查询

3. ASP.NET AJAX 开发电子书（124 页）

- 第 1 章 初识 ASP.NET AJAX
- 第 2 章 AJAX Control Toolkit 中的文本输入处理
- 第 3 章 基于 AJAX 的电子邮件处理

4. ASP.NET 2.0 网络开发技术

- 第 1 章 Web 开发基础知识
- 第 2 章 走进 ASP.NET 2.0

- 第3章 类、对象和命名空间
- 第4章 ASP.NET 2.0 核心对象
- 第5章 标准服务器控件
- 第6章 HTML 服务器控件
- 第7章 数据验证控件

5. .NET 2.0 入门视频 (4 个)

- Visual Studio 的安装和配置
- ASP.NET Web 程序开发入门
- .NET 程序调试方法
- 运行 eBookShop 实例

6. .NET 3.5 入门视频 (6 个)

- 创建 ASP.NET 程序
- 创建窗体应用程序
- 创建控制台应用程序
- 打开现有程序
- 读取 SQLExpress 数据
- 使用 SQLExpress 建立数据库

7. 常见的典型系统 (5 个)

- **博客引擎:** 是一个结构精炼但功能强大的博客系统, 它充分利用了 ASP.NET 2.0 的新特性, 比如母版页、主题、国际化等界面特性, 还使用了提供者模型实现了一个可扩展的数据访问层。
- **时间跟踪系统:** 在工作团队内部, 经常会列出项目计划、项目人员等信息。使用时间跟踪系统将这些计划信息录入系统, 在项目完成时录入实际完成信息, 这样可以跟踪项目进度。通过自定义的报表生成, 可以将预计与实际完成的时间进行比较。
- **图书管理系统:** 该管理信息系统具有一个典型的 Windows 窗体应用程序界面, 并完成图书馆日常工作中大部分常见的流程。本系统具备的功能有图书入库、借书、还书、图书管理、借阅人员管理、系统使用人员管理和惩罚管理等。
- **在线求职系统:** 本系统可以为企业和求职者提供一个互动的供求平台。企业需要查看网站人才库中感兴趣的求职者信息, 发布公司用人需求, 并建立与求职者的互动。求职者需要查看企业所发布的招聘信息, 并能通过求职网站了解公司的基本资料, 求职者与企业都要方便地收藏感兴趣的信息, 以便于相互进一步的沟通与交流。
- **中小型企业网站:** 本系统为企业提供了一个展示的平台, 向外界提供企业产品资料、企业专业人员、企业的客户信誉以及企业的相关新闻和联系方式。

编 者
2008 年 4 月

第一篇 C#语法篇

第 1 章 C#初步	2
1.1 创建 Hello World 控制台程序	3
它是学习一门新语言的好起点，我们就从这里开始吧！本示例创建一个控制台程序，并在屏幕上输出“Hello World”。	
1.2 创建 Hello World Web 应用程序	4
本示例创建一个 Web 应用程序，并在 Web 窗口中输出“Hello World”。	
1.3 创建 Hello World Windows 应用程序	5
本示例创建一个 Windows 应用程序，并在 Windows 窗口中输出“Hello World”。	
第 2 章 数据类型	7
2.1 C#数据类型与比较	8
C#语言是强类型语言，因此每个变量和对象都必须具有声明类型。C#语言的数据类型主要分为两类：值类型和引用类型。	
2.2 decimal 类型的格式化	9
decimal 关键字表示 128 位数据类型。同浮点型相比，decimal 类型具有更高的精度和更小的范围，这使它适合于财务和货币计算。	
2.3 Object 类型操作	11
在 C#语言的统一类型系统中，所有类型（预定义类型、用户定义类型、引用类型和值类型）都是直接或间接从 Object 类型继承的。可以将任何类型的值赋给 object 类型的变量。	
2.4 string 类型简单操作	13
字符串对象是“不可变的”，即它们一旦创建就无法更改。对字符串进行操作的方法实际上返回的是新的字符串对象。	
2.5 多种数字类型混合运算	14
数据类型之间的转换可以使用强制转换显式进行，但在某些情况下，也允许隐式转换。	
2.6 结构体声明和初始化	17
如果从结构创建一个对象并将该对象赋给某个变量，变量则包含结构的全部值。	
2.7 枚举类型的格式化	18
enum 关键字用于声明枚举，即一种由一组称为枚举数列表的命名常数组成的独特类型。每种枚举类型都有基础类型，该类型可以是除 char 类型以外的任何整型。	
2.8 字符类型与数字类型的转换	20
本示例说明如何在字符类型和数字类型之间进行转换。	

第3章 数组

22

- 3.1 数组的定义 23
C#语言的数组包括一维数组、多维数组、动态数组和交错数组四种类型，这四种数组均继承了 System.Array 类。动态数组使用 ArrayList 类实现。
- 3.2 遍历数组元素 24
C#语言提供 foreach 语句，该语句提供一种简单明了的方法来循环访问数组的元素。
- 3.3 操作数组 25
在 C#语言中，数组实际上是对象，而不只是像 C 和 C++语言中那样的可寻址连续内存区域。Array 是所有数组类型的抽象基类型。
- 3.4 获取数组属性 27
System.Array 类提供了许多有用的方法和属性，用于排序、搜索和复制数组。
- 3.5 数组参数的使用 29
数组是引用类型，数组作为方法的参数使用时，应注意 in、ref、out 声明的用法。

第4章 C#语句

32

- 4.1 使用 break 语句 33
break 语句用于终止最近的封闭循环或它所在的 switch 语句。
- 4.2 使用 continue 语句 34
continue 语句将控制权传递给它所在的封闭迭代语句的下次迭代。
- 4.3 使用 do 语句 36
do 语句重复执行{}中的一个语句或语句块，直到指定的表达式计算为 false。
- 4.4 使用 foreach 语句 37
foreach 语句为数组或对象集中的每个元素重复一个嵌入语句组。
- 4.5 使用 for 语句 39
for 循环重复执行一个语句或语句块，直到指定的表达式计算为 false。
- 4.6 使用 goto 语句 40
goto 语句将程序控制直接传递给标记语句。goto 的一个通常用法是将控制传递给特定的 switch-case 标签或 switch 语句中的默认标签。
- 4.7 使用 if 语句 42
if 语句根据 Boolean 表达式的值选择要执行的语句。
- 4.8 使用 switch 语句 44
switch 语句是一个控制语句，它通过将控制传递给其体内的一个 case 语句来处理多个选择和枚举。
- 4.9 使用 while 语句 45
while 语句执行一个语句或语句块，直到指定的表达式计算为 false。

第5章 运算符

48

- 5.1 使用 as 运算符 49
用于在兼容的引用类型之间执行转换。as 运算符类似于强制转换，所不同的是，当转换失败时运算符将产生空，而不是引发异常。

5.2	使用 is 运算符.....	50
	检查对象是否与给定类型兼容。例如，可以确定对象是否与 string 类型兼容。is 运算符只考虑引用转换、装箱转换和取消装箱转换。不考虑其他转换，如用户定义的转换。	
5.3	使用 new 运算符.....	52
	用于创建对象和调用构造函数。new 运算符还用于调用值类型的默认构造函数。	
5.4	使用 sizeof 运算符.....	54
	用于获取值类型的字节大小。	
5.5	使用 typeof 运算符.....	55
	用于获取类型的 System.Type 对象。typeof 运算符也能用于公开的泛型类型。具有不只一个类型参数的类型的规范中必须有适当数量的逗号。	
5.6	使用递增递减运算符.....	58
	增量运算符 (++) 将操作数加 1。增量运算符可以出现在操作数之前或之后。减量运算符 (--) 将操作数减 1。减量运算符可以出现在操作数之前或之后。	
5.7	使用赋值运算符.....	60
	提供 13 种赋值运算符。如赋值运算符 (=) 将右操作数的值存储在左操作数表示的存储位置、属性或索引器中，并将值作为结果返回。	
5.8	使用关系运算符.....	62
	提供 6 种关系运算符。如对于预定义的值类型，如果操作数的值相等，则相等运算符 (==) 返回 true，否则返回 false。	
5.9	使用逻辑运算符.....	64
	提供 9 种逻辑运算符。例如，“&”运算符既可作为一元运算符也可作为二元运算符。	
5.10	使用算术运算符.....	67
	提供 5 种算术运算符。例如，“+”运算符既可作为一元运算符也可作为二元运算符。	

第 6 章 预处理指令

69

6.1	使用 region 定义代码区间.....	70
	#region 在使用 Visual Studio 代码编辑器的大纲显示时，指定可展开或折叠的代码块。	
6.2	使用 define 指令.....	71
	#undef 可以取消符号的定义，以便通过将该符号用做 #if 指令中的表达式，使表达式的计算结果为 false。	
6.3	使用 warning 和 error 指令.....	72
	#warning 从代码的特定位置生成一级警告。#error 可以从代码中的特定位置生成错误。	
6.4	使用条件指令.....	73
	提供 4 种条件指令。	

第 7 章 异常处理

75

7.1	创建自定义异常.....	76
	C#程序员使用 try 块对可能受异常影响的代码进行分区，并使用 catch 块来处理所产生的任何异常。不管是否引发异常，都可以使用 finally 块来执行代码。	

7.2 获取异常信息	77
catch 块可以指定要捕捉的异常类型。这个类型称为“异常筛选器”，它必须是 Exception 类型，或者必须从此类型派生。	
7.3 使用 finally 关键字	78
finally 块允许清理在 try 块中执行的操作。如果存在 finally 块，它将在执行完 try 和 catch 块之后执行。	
7.4 使用 try-catch 语句捕获异常	80
try-catch 语句由一个 try 块后跟一个或多个 catch 子句构成，这些子句指定不同的异常处理程序。	
7.5 使用多 catch 语句捕获异常	82
在同一个 try-catch 语句中可以使用一个以上的特定 catch 子句。这种情况下 catch 子句的顺序很重要，因为会按顺序检查 catch 子句。	

第二篇 C#面向对象篇

第 8 章 类与接口

86

8.1 传递类与传递结构体的区别	87
本示例演示在向方法传递结构时，传递的是该结构的副本，而在传递类实例时，传递的是一个引用。	
8.2 多个接口的继承	89
类和结构可从多个接口继承。接口自身可从多个接口继承。	
8.3 复制构造函数	91
与有些语言不同，C# 不提供复制构造函数。如果创建了新的对象并希望从现有对象复制值，必须自行编写适当的方法。	
8.4 类的定义	93
类是 C# 中功能最为强大的数据类型。像结构一样，类也定义了数据类型的数据和行为。	
8.5 类的继承	95
继承是通过使用派生来实现的，它的数据和行为从基类继承。	
8.6 使用抽象类	98
可以将类声明为抽象类。方法是在类定义中将关键字 abstract 置于关键字 class 之前。	
8.7 使用静态构造函数	100
静态构造函数用于初始化任何静态数据，或用于执行仅需执行一次的特定操作。在创建第一个实例或引用任何静态成员之前，将自动调用静态构造函数。	
8.8 使用私有构造函数	101
私有构造函数是一种特殊的实例构造函数，它通常用在只包含静态成员的类中。	
8.9 使用析构函数	103
析构函数用于析构类的实例。	
8.10 显式实现接口方法	104
如果类实现两个接口，并且这两个接口包含具有相同签名的成员，那么在类中实现该成员将导致两个接口都使用该成员作为它们的实现。	

8.11 显式实现接口属性	106
本示例声明一个接口 IDimensions 和一个类 Box，该类显式实现接口成员 getLength 和 getWidth。通过接口实例 dimensions 访问这些成员。	

第 9 章 方法和属性 108

9.1 方法的参数使用	109
方法参数放在括号中，并用逗号隔开。空括号表示方法不需要参数。	
9.2 方法的改写	111
派生类型可以重写继承的虚方法。重写方法提供了对方法的一种新定义，将根据运行时的值的类型，而不是编译时已知的变量类型来调用方法。	
9.3 方法的重载	112
每个类型成员都有一个唯一的签名。方法签名由方法名称和一个参数列表（方法的参数的顺序和类型）组成。	
9.4 类的属性定义	113
类的属性是实现封装性的数据结构，具有 get 访问器和 set 访问器，这两个访问器定义了属性的读取和赋值方式。	
9.5 使用抽象属性	115
抽象属性定义属性的读写方式，在具有抽象属性的类派生的子类中，必须使用 override 关键字实现该抽象属性。	
9.6 使用静态属性	117
类的静态属性通过类名称调用，而不通过类实例的名称调用。	
9.7 属性的继承	118
子类继承了父类的属性，此时在子类中虽然不包含属性的声明，也能够使用该属性，如果子类需要重新定义属性，可使用 new 关键字覆盖父类的属性。	

第 10 章 命名空间 120

10.1 创建自定义命名空间	121
使用 namespace 关键字创建自定义的命名空间。	
10.2 获取类的完整命名空间	122
在程序中，通过实例的 GetType 方法获取实例的类结构，再调用该类的 Namespace 属性获取类的完整命名空间。	
10.3 使用命名空间别名限定符	123
使用 “=” 运算符为命名空间声明别名，使用 “: :” 运算符获取别名定义的命名空间的成员。	

第 11 章 委托和事件 125

11.1 实现接口中的事件	126
接口中的事件在类中具有显式和隐式两种实现方式。	
11.2 使用事件和委托	128
事件处理方法可以使用委托的方式由外部程序提供，当类中的代码产生事件时，应判断委托是否已经被指定，然后执行委托中指定的事件处理方法。	

11.3	使用委托	130
	使用委托时会产生协变和逆变的现象。协变表现为指定方法返回的类型是委托声明类型的派生类，逆变表现为方法的参数类型是委托声明的基类。	
11.4	使用字典存储事件	132
	使用泛型的字典能够存储事件，灵活地实现多个事件的处理方法。	
11.5	委托方法数量的增加与减少	135
	使用“+=”运算符向委托中增加方法，使用“-=”运算符向委托中移除方法。可以重复添加。	
11.6	在类中实现事件处理	136
	类内部的事件指的是在类的封装范围之内的事件，这类事件的特点是不向外部程序提供事件，而且由内部程序提供处理方法。	

第三篇 高级技术篇

第 12 章 ADO.NET 编程

142

12.1	创建 Web 程序连接 Access 数据库	143
	使用 OLEDB 数据库引擎连接 Access 数据库，并在程序中获取当前网站所在的完整路径，以连接该路径下的 Access 数据库。	
12.2	创建 Web 程序连接 SQL Server Express 数据库文件	145
	使用“\SQLEXPRESS”数据库引擎连接 SQL Server Express 数据库，并在程序中获取当前网站所在的完整路径，以连接该路径下的 SQL Server Express 数据库。	
12.3	创建 Web 程序连接 SQL Server 数据库	147
	VS2008 中不仅提供了 SQL Server Express 的集成，也对 SQL Server 数据库提供了强大的支持，以满足实际应用中服务器端数据库策略的需求。	
12.4	创建 Windows 程序连接 SQL Server 数据库	149
	.NET Framework 下的 Windows 应用连接 SQL Server 数据库时，与 Web 程序连接 SQL Server 数据库大同小异，其中的差别只是界面的设计与触发事件的不同。	
12.5	定义 DataTable 架构	151
	使用 DataTable 的构造函数创建 DataTable 类实例，并使用 Columns 集合的 Add 方法定义数据表字段，使用 Rows 集合的 Add 方法添加数据。	
12.6	将数据集数据写入到 XML 和从 XML 读取数据	153
	使用 DataSet 类中定义的 GetXml 方法将 DataSet 中的数据转换为 XML 字符串。使用 DataSet 类中定义的 ReadXml 方法将 XML 中的数据加载到 DataSet 中。	
12.7	使用 BeginExecuteReader 实现异步读取数据	156
	使用 DataReader 类中定义的 BeginExecuteReader 方法实现异步读取数据。	
12.8	使用 DataTableReader 浏览数据表中的数据	160
	DataTableReader 类表示包含一个或多个的、只读向前的结果集的数据读取器。当 DataTableReader 包含多个结果集时，游标将会在结果集之间贯穿。	
12.9	使用 DataView 对数据进行排序和过滤	162
	DataView 表示数据的视图，具有对数据进行排序和筛选的功能。	

12.10	使用 DataView 获取数据视图	164
	DataView 通过数据关系的定义能够创建从表的数据视图。	
12.11	使用 DataView 类修改数据	166
	DataView 的 AllowNew 属性、AllowEdit 属性和 AllowDelete 属性分别表示是否允许新增、修改和删除 DataView 中的数据。	
12.12	使用 ExecuteScalar 读取数据	169
	ExecuteScalar 是 DbCommand 类的一种查询方式，该方法执行返回的是查询结果集的第一行第一列的数据。	
12.13	使用 ExecuteXmlReader 读取数据	170
	ExecuteXmlReader 方法返回的是一个 XmlReader 类实例，该实例包含 XML 数据，要求 ExecuteXmlReader 方法中的 SQL 语句执行的结果必须是 XML 格式的数据。	
12.14	使用 SqlConnection 创建数据库连接	171
	SqlConnection 类表示到 SQL Server 数据库的一个连接，通过指定该类中包含的 ConnectionString 属性，可以使用不同的方式，或者不同的用户创建数据库连接。	
12.15	使用 SqlDataAdapter 获取数据	174
	SqlDataAdapter 类表示的是 SQL Server 数据库的适配器。该适配器的主要作用是在数据库连接和 DataSet 之间进行交互。	
12.16	使用 SqlDataReader 获取数据	176
	SqlDataReader 类是为 ADO.NET 读取 SQL Server 数据的、只进向前的读取器。	
12.17	使用 T-SQL 命令填充数据对象	178
	执行 T-SQL 命令最常用的类是 SqlCommand 类。	
12.18	使用存储过程填充数据对象	179
	存储过程是保存在数据库中的一组 SQL 语句命令，可以为存储过程指定一个或多个参数。	

第 13 章 事务处理 181

13.1	使用 SqlTransaction 实现数据库操作事务	182
	SqlTransaction 类是对 SQL Server 数据库进行事务处理的类。	
13.2	使用 TransactionScopeOption 实现事务选项控制	184
	TransactionScopeOption 枚举中包含三个成员：Required、RequiresNew 和 Suppress。	
13.3	使用 TransactionScope 实现单数据库连接事务操作	186
	TransactionScope 类表示一个事务范围，位于该范围内的代码块都成为一个事务，在调用 TransactionScope 类中定义的 Complete 方法时，将结束并提交事务。	
13.4	使用 TransactionScope 实现多数据库连接事务操作	188
	当应用程序需要在多个数据库中进行事务性操作的时候，使用 TransactionScope 类可以方便地实现应用程序的这一需求。	

第 14 章 XML 处理 191

14.1	从 XML 文件中加载 DataSet	192
	调用 DataSet 中定义的 ReadXml 方法从 XML 文件中加载 DataSet。	

14.2	从 XML 字符串中加载 DataSet.....	193
	使用 DataSet 的 ReadXml 方法从 XML 字符串中加载 DataSet。	
14.3	将对象层次结构映射到 XML 数据	195
	XML 是一种表示树型结构的数据。XML 数据中包含的内容成为树型结构中的节点。XML 节点映射了 XML 数据中的内容。	
14.4	使用 DataSet 读取和写入 XML 架构.....	198
	使用 ReadXmlSchema 方法读取包含在 XML 数据中的架构信息,创建 DataSet 中数据表的结构。	
14.5	使用 XmlDocument 保存 XML 数据.....	201
	XmlDocument 类描述了一个位于内存中 DOM 结构的 XML 数据的整体。	
14.6	使用 XmlReader 读取 XML 元素.....	202
	XmlReader 类是 XML 数据的读取器,该读取器具有快速、非缓存、只进访问的特点。	
14.7	使用 XmlReader 读取 XML 属性.....	204
	XmlReader 读取 XML 属性时,应首先定位到该属性所在的元素,然后通过 XmlReader 类实例的 GetAttribute 方法,从当前元素的数据中获取属性。	
14.8	使用 XmlWriter 创建 XML 文件内容.....	206
	使用 WriteStartDocument 方法开始一个 XML 文档的编辑,使用 WriteEndDocument 方法则用于来结束一个 XML 文档的编辑。	
14.9	使用 XPathNavigator 编辑 XML 数据	208
	XPathNavigator 通过一个指定的 XPath 字符串作为选择条件,将 XPathNavigator 表示的游标定位到 XML 数据中符合条件的节点。	
14.10	使用 XPathNavigator 选择 XML 数据	210
	XPathNavigator 类中定义的 Select 方法,使用指定 XPath 表达式字符串作为参数,返回符合该参数的节点集的迭代器。	
14.11	使用 XslCompiledTransform 类实现 XSLT 处理器.....	213
	XSLT 表示的是转换 XML 数据的样式表,在 .NET Framework 中提供了使用 XSLT 转换 XML 数据的 XslCompiledTransform 类。	

第 15 章 文件操作

216

15.1	创建目录列表	217
	使用路径参数创建包含了指定目录信息的 DirectoryInfo 类实例,该实例包括目录的属性和文件列表。使用 GetFiles 方法能够获取指定目录的文件列表。	
15.2	从文本文件中读取数据	218
	使用 StreamReader 类能按照指定的编码设置读取文本文件。	
15.3	创建日志文件	220
	使用 StreamWriter 类实例向文件中写入数据,因为部分数据可能由于系统处理缓慢而不能及时写入,所以在所有的写入操作完成后,需要调用 Flush 方法。	
15.4	对日志文件的读写操作	221
	StreamWriter 类定义的 Seek 方法,将写入位置移动到文件末尾,实现将数据以追加方式写入日志文件的功能。Peek 方法则用于判断读取器是否已经读到日志文件末尾。	