Join the discussion @ p2p.wrox.com

Wrox Programmer to Programmer™

Professional C# 4 and .NET 4

C#高级编程

（第7版）

(美) Christian Nagel
Bill Evjen 等著
Jay Glynn
李 铭 译
黄 静 审校

本书源代码与
第48~57章(电子书)

清华大学出版社

C#高级编程

(第7版)

Christian Nagel
(美) Bill Evjen 等著
Jay Glynn

李 铭 译

清华大学出版社

北 京

Christian Nagel，Bill Evjen，Jay Glynn，et al
Professional C# 4 and .NET 4
EISBN：978-0-470-50225-9

图书在版编目(CIP)数据

C#高级编程(第 7 版)/(美)内格尔(Nagel, C.)，(美)埃夫琴(Evjen, B.)等著；李铭 译；黄静 审校.
—北京：清华大学出版社，2010.11
书名原文：Professional C# 4 and .NET 4
ISBN 978-7-302-23937-6

Ⅰ.C… Ⅱ.①内… ②埃… ③李… ④黄… Ⅲ.C 语言－程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2010)第 195700 号

责任编辑：王　军　谢晓芳
装帧设计：孔祥丰
责任校对：成凤进
责任印制：王秀菊
出版发行：清华大学出版社　　地　　址：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　邮　　编：100084
社　总　机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：清华大学印刷厂
装 订 者：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185×260　印　张：94.25　字　数：2529 千字
附光盘 1 张
版　　次：2010 年 11 月第 1 版　　印　　次：2010 年 11 月第1 次印刷
印　　数：1～5000
定　　价：148.00 元

产品编号：035929-01

作 者 简 介

CHRISTIAN NAGEL 是 Microsoft 区域董事、Microsoft MVP，thinktecture 的合作伙伴，CN 革新技术的拥有者，他是一位软件架构师和开发人员，为开发 Microsoft .NET 解决方案提供培训和咨询服务。他具备超过 25 年的软件开发经验。Christian 从 PDP 11 和 VAX/VMS 系统开始其计算机生涯，熟悉各种语言和平台。自从 2000 年以来，(那时.NET 还只是一个技术框架)他就开始使用各种.NET 技术构建大量.NET 解决方案。他具备 Microsoft 技术的深厚功底，编写了大量.NET 图书，并获得了 Microsoft 认证培训师和专业开发人员证书。Christian 在国际会议发表演讲(如 TechEd 和 Tech Days)并创立 INETA Europe，以支持.NET 用户组。通过 Web 站点 www.cninnovation.com 和 www.thinktecture.com 可以联系 Christian，在 www.twitter.com/christiannagel 上可以找到他。

BILL EVJEN 是.NET 技术和基于社团的.NET 学习活动的积极倡导者，他自从.NET 在 2000 年第一次发布以来就积极涉足.NET。同年，Bill 成立了 St. Louis .NET 用户组(www.stlnet.org)，这是世界上第一个这样的用户组，Bill 还是国际.NET 协会(www.ineta.org)的奠基人和前任执行主管，该协会在全世界有 500 000 多位成员。

Bill 住在密苏里州的圣路易斯，是 ASP.NET 和 Web 服务的一位著名作者和演说家。他编写或与他人合作编写了 20 多本书，包括 *Professional ASP.NET 4*、*Professional VB 2008*、*ASP.NET Professional Secrets*、*XML Web Services for ASP.NET* 和 *Web Services Enhancements: Understanding the WSE for Enterprise Applications*(均由 Wiley 出版)。除了写作之外，Bill 还在许多会议上发表演讲，包括 DevConnections、VSLive 和 TechEd。除了这些活动之外，Bill 还与 Microsoft 联系紧密，是 Microsoft 区域董事和 MVP。

Bill 是国际新闻及财务服务公司(www.thomsonreuters.com)Thomson Reuters, 即 Lipper 的全球平台架构师。他毕业于华盛顿州贝灵翰姆的西华盛顿大学，获得了俄语学位。他闲暇时通常在芬兰的 Toivakka 度假。在 Twitter 网站上可以通过@billevjen 联系到 Bill。

JAY GLYNN 是 PureSafety 的首席构架师，PureSafety 是一家为劳动力的安全和健康提供结果驱动的软件和信息解决方案的业界领先的提供商。Jay 开发软件的时间有 25 余年，使用过各种语言和技术，包括 PICK Basic、C、C++、Visual Basic、C#和 Java。Jay 目前与妻儿住在田纳西州的富兰克林。

KARLI WATSON 是 Infusion Development (www.infusion.com)的顾问，Boost.net (www.boost.net)的技术架构师和 IT 自由撰稿专业人士、作家和开发人员。他主攻.NET(尤其是 C#和后来的 WPF)，在这个领域编写了许多图书。他擅长以通俗易懂的方式激情澎湃地阐述复杂的理念，并花了大量的时间研究新技术，寻求可教给其他人的新技术。

在不工作时(这种时间似乎没有)，Karli 希望到山上滑雪，或者尝试发表他的小说。他喜欢穿颜色鲜亮的衣服，可以在 www.twitter.com/karlequin 网站上找到他，也许有一天他自己会建立一个网站。

MORGAN SKINNER 年轻时对 Sinclair ZX80 很感兴趣，在校期间就开始了计算机生涯，当时他对教师编写的一些代码不感兴趣，便开始用汇编语言编程。从此以后他使用各种语言和平台，包括 VAX 宏汇编程序、Pascal、Modula2、Smalltalk、X86 汇编语言、PowerBuilder、C/C++、VB 和目前的 C#，自从 2000 年发布 PDC 以来，他就用.NET 编程，而且非常喜欢.NET，于是在 2001 年加入了 Microsoft。他现在是开发人员的主要支持人员，而且花了大多数时间帮助客户使用 C#。在 www.morganskinner.com 上可以联系到 Morgan。

前　言

对于开发人员，把C#语言及其相关联的.NET Framework环境描述为多年来最重要的新技术一点都不夸张。.NET提供了一种环境。在这个环境中，可以开发在Windows上运行的几乎所有应用程序，而C#是专门用于.NET Framework的编程语言。例如，使用C#可以编写动态Web页面、Windows Presentation Foundation应用程序、XML Web服务、分布式应用程序的组件、数据库访问组件、传统的Windows桌面应用程序，甚或可以联机/脱机运行的新型智能客户端应用程序。本书介绍.NET Framework 4。如果读者使用以前的版本编码，本书的一些章节就不适用。本书将标注出专用于.NET Framework 4的新增内容。

不要被这个Framework名称中的.NET所迷惑，认为这是一个只关注Internet的架构。这个名称中的.NET仅强调Microsoft相信分布式应用程序是未来的趋势，即处理过程分布在客户端和服务器上。理解C#不仅仅是编写Internet或与网络能识别的应用程序的一种语言也很重要。它还提供了一种编写Windows平台上几乎任何类型的软件或组件的方式。另外，C#和.NET都对开发人员编写程序的方式进行了革新，更易于实现在Windows上的编程。

那么，.NET和C#有什么优点？

.NET和C#的重要性

为了理解.NET的重要性，了解一下过去18年来出现的许多Windows技术的本质，会有一定的帮助。尽管所有Windows操作系统在表面上看来完全不同，但从Windows 3.1(1992年引入)到Windows 7和Windows Server 2008 R2，在内核上都有相同的Windows API。在我们转而使用Windows的新版本时，虽然API中增加了非常多的新功能，但这是一个演化和扩展API的过程，并非替换它。

开发Windows软件所使用的许多技术和架构也是这样。例如，组件对象模型(Component Object Model，COM)源自对象链接和嵌入(Object Linking and Embedding，OLE)。最初，因为它在很大程度上仅把不同类型的Office文档链接在一起，所以利用它，例如，可以把一个小型Excel电子表格放在Word文档中。之后，它逐步演化为COM、DCOM(Distributed COM，分布式组件对象模型)和最终的COM+。COM+是一种复杂的技术，它是几乎所有组件通信方式的基础，实现了事务处理、消息传输服务和对象池。

Microsoft选择这种革新方法的原因非常明显：它关注后向兼容性。在过去的这些年中，第三方软件编写了大量Windows软件，如果Microsoft每次都引入一项不遵循现有基本代码的新技术，Windows就不会获得今天的成功。

后向兼容性是Windows技术的极其重要的功能，也是Windows平台的一个长处。但它有一个很大的缺点：每次某项技术更新换代，增加了新功能后，它都会比它以前更复杂。

很明显，对此必须进行改进。Microsoft不可能一直扩展相同的开发工具和语言，总是使它们越

来越复杂，既要保证能跟上最新硬件的发展步伐，又要与 20 世纪 90 年代初开始流行的 Windows 产品向后兼容。如果要得到一系列简单而专业的语言、环境和开发工具，让开发人员轻松地编写最新的软件，就需要一个新的开端。

这就是 C#和.NET 的作用。粗略地说，.NET 是一种在 Windows 平台上编程的架构——一种 API。C#是一种从头开始设计的用于.NET 的语言，它可以利用.NET Framework 及其开发环境中的所有新增功能，以及在最近 25 年来出现的面向对象的编程方法。

在继续介绍前，必须先说明，后向兼容性并没有在这个演化进程中丧失。现有的程序仍可以使用，.NET 也兼容现有的软件。现在，在 Windows 上软件组件之间的通信几乎都使用 COM 实现。因此，.NET 能够提供现有 COM 组件的包装器(wrapper)，以便.NET 组件与之通信。

我们不需要学习了 C#才能给.NET 编写代码，因为 Microsoft 已经扩展了 C++，还对 Visual Baisc 进行了很多改进，把它转变成了功能更强大的语言，并允许把用这些语言编写的代码用于.NET 环境。但其他这些语言都因有多年演化的遗留痕迹，并非一开始就用现在的技术来编写，导致它们不能用于.NET 环境。

本书将介绍 C#编程技术，同时提供.NET 体系结构工作原理的必要背景知识。我们不仅会介绍 C#语言的基础，还会给出使用各种相关技术的应用程序对应的示例，包括数据库访问、动态的 Web 页面、高级的图形和目录访问等。

.NET 的优点

前面阐述了.NET 的优点，但并没有说它会使开发人员的工作更易完成。本节将简要讨论.NET 的改进功能。

- **面向对象编程**：.NET Framework 和 C#从一开始就完全基于面向对象的原则。
- **优秀的设计**：一个基类库，它以一种非常直观的方式设计出来。
- **语言无关性**：在.NET 中，Visual Baisc、C#和托管 C++等语言都可以编译为通用的中间语言(Intermediate Language)。这说明，语言可以用以前没有的方式交互操作。
- **对动态 Web 页面更好的支持**：虽然 ASP 具有很大的灵活性，但效率不是很高，这是因为它使用了解释性的脚本语言，且缺乏面向对象的设计，从而导致 ASP 代码比较混乱。.NET 使用 ASP.NET，为 Web 页面提供了一种集成支持。使用 ASP.NET，可以编译页面中的代码，这些代码还可以使用.NET 能识别的高级语言来编写，如 C#或 Visual Baisc 2010。.NET 现在还添加了对最新 Web 技术的重要支持，如 Ajax 和 jQuery。
- **高效的数据访问**：一组.NET 组件，统称为 ADO.NET，提供了对关系数据库和各种数据源的高效访问。这些组件也可用于访问文件系统和目录。尤其是，.NET 内置了 XML 支持，可以处理从非 Windows 平台导入或导出的数据。
- **代码共享**：.NET 引入了程序集的概念，替代了传统的 DLL，可以完美无暇地改进代码在应用程序之间的共享方式。程序集是解决版本冲突的正式设备，程序集的不同版本可以并存。
- **增强的安全性**：每个程序集还可以包含内置的安全信息，这些信息可以准确地指出谁或哪种类型的用户或进程可以调用什么类的哪些方法。这样就可以非常准确地控制用户部署的程序集的使用方式。
- **对安装没有任何影响**：有两种类型的程序集，分别是共享程序集和私有程序集。共享程序集是可用于所有软件的公共库，而私有程序集只用于特殊软件。由于私有程序集完全自包

含，所以安装过程非常简单。没有注册表项，只需把相应的文件放在文件系统的相应文件夹中即可。

- **Web 服务的支持**：.NET 完全集成了对开发 Web 服务的支持，用户可以轻松地开发任何类型的应用程序。
- **Visual Studio 2010**：.NET 附带了一个 Visual Studio 2010 开发环境，它同样可以很好地利用 C++、C#、Visual Basic 2010 和 ASP.NET 或 XML 进行编码。Visual Studio 2010 集成了这个 IDE 所有以前版本中的各种语言专用的环境中的所有最佳功能。
- **C#**：是使用.NET 的一种面向对象的强大且流行的语言。

第 1 章将详细讨论.NET 体系结构的优点。

.NET Framework 4 中的新增特性

.NET Framework 的第 1 版(1.0 版)在 2002 年发布，赢得了许多人的喝彩。.NET Framework 2.0 在 2005 年发布，认为它是该架构的一个主要版本。.NET Framework 4 是该产品另一个重要的版本，包含了许多重要的新功能。

对于.NET Framework 的每个版本，Microsoft 总是试图确保对已开发出的代码进行尽可能少的不兼容的更改。到目前为止，Microsoft 在这方面做得很成功。

下面将详细描述 C# 2010 和.NET Framework 4 中的一些新变化。

动态类型

编程界在动态语言(如 JavaScript、Python 和 Ruby)方面的进步非常快。由于这类编程越来越流行，Microsoft 在 C#中发布了一个新的动态类型功能。并不总是可以以静态方式确知对象最终是什么类型。现在不使用 object 关键字和从这个类型生成的所有对象，而可以让动态语言运行库(Dynamic Language Runtime，DLR)在运行期间动态地确定对象的类型。

使用 C#新增的动态功能，可以更好地进行交互操作。我们可以与各种动态语言交互操作，更容易地使用 DOM。甚至现在使用 Microsoft Office COM API 也更容易。

在.NET Framework 4 这个版本中，Microsoft 包含了动态语言运行库。DLR 建立在公共语言运行库(Common Language Runtime，CLR)的基础上，提供了把所有动态语言交互操作连接起来的功能。

C#使用新的 dynamic 关键字访问新的 DLR。这对于编译器是一个标记，只要遇到这个关键字，编译器就认为它是一个动态调用，而不是一般的静态调用。

可选参数和命名参数

虽然可选参数和命名参数在 Visual Basic 中已存在一段时间了，但在.NET 4 发布之前，它们不能在 C#中使用。可选参数允许为方法的一些参数提供默认值，并允许使用者重载类型，因此，即使只有一个方法，也能处理所有变体。下面是一个例子：

```
public void CreateUser(string firstname, string lastname,
bool isAdmin, bool isTrialUser)
{
}
```

如果要重载这个方法，并为两个 bool 对象提供默认值，就很容易得到好几个方法，为使用者填充这些值，然后调用主方法，以实际创建用户。现在通过可选参数，就可以编写下面的代码：

```
public void CreateUser(string firstname, string lastname,
bool isAdmin = false, bool isTrialUser = true)
{
}
```

查看这段代码，firstname 和 lastname 参数没有设置默认值，而 isAdmin 和 isTrailUser 参数设置了默认值。使用者现在可以编写如下代码：

```
myClass.CreateUser("Bill", "Evjen");
myClass.CreateUser("Bill", "Evjen", true);
myClass.CreateUser("Bill", "Evjen", true, false);
myClass.CreateUser("Bill", "Evjen", isTrailUser: false);
```

上一个例子使用了命名参数，这也是在.NET Framework 的这个版本中 C#的一个新功能。命名参数会潜在地改变编写代码的方式。这个新功能能使代码更容易阅读和理解。例如，看一下 System.IO 名称空间中的 File.Copy()方法，它一般构建为：

```
File.Copy(@"C:\myTestFile.txt", @"C:\myOtherFile.txt", true);
```

在这行代码中，这个简单的方法使用 3 个参数，但实际传递给 Copy()方法的是什么内容？除非知道这个方法的前前后后，否则仅看一眼该方法，很难判断出该方法会执行何种操作。而通过命名参数，就可以在提供参数值之前使用代码中的参数名，如下面的示例所示：

```
File.Copy(sourceFileName: @"C:\myTestFile.txt",
destFileName: @"C:\myOtherFile.txt", overwrite: true);
```

现在通过命名参数，就很容易阅读和理解这行代码将执行的操作。使用命名参数对最终的编译没有影响，命名参数仅用在应用程序的编码中。

协变和抗变

虽然在.NET Framework 的以前版本中包含协变和抗变，但它们在.NET 4 中进行了扩展，当处理泛型、委托等时，它们会执行得更好。例如，在.NET 的以前版本中，可以对对象和数组使用抗变，但不能对泛型接口使用抗变。而在.NET 4 中，就可以对泛型接口使用抗变。

ASP.NET MVC

ASP.NET MVC 是 ASP.NET 最新的主要新增内容，它为开发团队带来了许多惊喜。ASP.NET MVC 提供了许多开发人员期待的、使用模型-视图-控制器来创建 ASP.NET 的方式。ASP.NET MVC 在开发人员构建的应用程序中提供了可测试性、灵活性和可维护性。记住，ASP.NET MVC 不是每个人都知道和喜欢的 ASP.NET 的替代品，而只是构建应用程序的另一种方式。

ASP.NET 的这个版本允许使用这个新模型构建应用程序，它完全内嵌在 Framework 和 Visual Studio 中。

C#的优点

C#在某种程度上可以看作是.NET 面向 Windows 环境的一种编程语言。在过去的十几年中，Microsoft 给 Windows 和 Windows API 添加了许多功能，Visual Baisc 2010 和 C++也进行了许多扩展。虽然 Visual Baisc 和 C++最终已成为非常强大的语言，但这两种语言也存在问题，因为它们保留了原来的一些遗留内容。

对于 Visual Basic 6 及其早期版本，它的主要优点是很容易理解，许多编程工作都很容易完成，从很大程度上对开发人员隐藏了 Windows API 和 COM 组件结构的详细信息。其缺点是因为 Visual Basic 从来没有实现真正意义上的面向对象，所以大型应用程序很难分解和维护。另外，因为 Visual Baisc 的语法继承自 BASIC 的早期版本(BASIC 主要是为了让刚入门的程序员更容易理解，而不是为了编写大型商业应用程序)，所以不能真正成为结构良好或面向对象的编程语言。

另一方面，C++基于 ANSI C++语言定义。它与 ANSI 不完全兼容，因为 Microsoft 在 ANSI 定义标准化之前编写其 C++编译器，但它已经相当接近。但是，这导致了两个问题。首先，ANSI C++是在十几年前的技术条件下开发的，因此它不支持现在的概念(如 Unicode 字符串和生成 XML 文档)，某些古老的语法结构是为以前的编译器设计的(如成员函数的声明和定义是分开的)。其次，Microsoft 同时还试图把 C++演变为一种用于在 Windows 上执行高性能任务的语言，为此不得不在语言中添加大量 Microsoft 专用的关键字和各种库。其结果是在 Windows 上，该语言非常杂乱。让 C++开发人员描述字符串有多少种定义就可以证明这一点：char*、LPTSTR、string、CString(MFC 版本)、CString(WTL 版本)、wchar_t*、OLECHAR*等。

现在进入.NET 时代——一种全新的环境，它对这两种语言都进行了新的扩展。Microsoft 给 C++添加了许多 Microsoft 专用的关键字，并把 Visual Baisc 演变为 Visual Baisc 2010，保留了一些基本的 Visual Baisc 语法，但在设计上完全不同于原始 Visual Basic，从实际应用的角度来看，Visual Baisc 2010 是一种新语言。

在这里，Microsoft 决定给开发人员提供另一个选择——专门用于.NET、具有新起点的一种语言，即 C#。Microsoft 在正式场合把 C#描述为一种简单、现代、面向对象、类型非常安全、派生自 C 和 C++的编程语言。大多数独立的评论员对 C#的描述改为“派生自 C、C++和 Java”。这种描述在技术上非常准确，但没有表达出该语言的真正优点。从语法上看，C#非常类似于 C++和 Java，许多关键字都相同，C#也使用类似于 C++和 Java 的块结构，并用花括号({})来标记代码块，用分号分隔各行语句。对 C#代码的第一印象是它非常类似于 C++或 Java 代码。但在这些表面的类似性后面，C#学习起来要比 C++容易得多，但比 Java 难一些。其设计比其他语言更适合现代开发工具，它同时具有 Visual Basic 的易用性，以及 C++的高性能、低级内存访问。C#包括以下一些功能：

- 完全支持类和面向对象编程，包括接口和实现继承、虚函数和运算符重载。
- 一致且定义完善的基本类型集。
- 对自动生成 XML 文档的内置支持。
- 自动清理动态分配的内存。
- 可以用用户定义的属性来标记类或方法。这可以用于文档，对编译有一定的影响(例如，把方法标记为只在调试版本中编译)。
- 可以完全访问.NET 基类库，并易于访问 Windows API(如果实际需要它，这就不常见)。

- 可以使用指针和直接访问内存，但 C#语言可以在没有它们的条件下访问内存。
- 以 Visual Basic 的风格支持属性和事件。
- 改变编译器选项，可以把程序编译为可执行文件或.NET 组件库，该组件库可以用与 ActiveX 控件(COM 组件)相同的方式由其他代码调用。
- C#可以用于编写 ASP.NET 动态 Web 页面和 XML Web 服务。

应该指出，对于上述大多数功能，Visual Baisc 2010 和 Managed C++也具备。事实上，虽然 C#从一开始就使用.NET，但对.NET 功能的支持不仅更完整，而且在比其他语言更合适的语法环境中提供了这些功能。C#语言本身非常类似于 Java，但其中有一些改进，尤其是，Java 并不应用于.NET 环境。

在结束这个主题前，还要指出 C#的两个局限性。一方面是该语言不适用于编写时间急迫或性能非常高的代码，例如一个要占用 1000 或 1050 个机器周期的循环，并在不需要这些资源时，立即清理它们。在这方面，C++可能仍是所有低级语言中的佼佼者。另一方面是 C#缺乏性能极高的应用程序所需要的关键功能，包括能够指定那些保证在代码的特定地方运行的内联函数和析构函数。但这类应用程序非常少。

编写和运行 C#代码的环境

.NET Framework 4 运行在 Windows XP/2003/7 和最新的 Windows Server 2008 R2 上。要使用.NET 编写代码，需要安装.NET 4 SDK。

此外，除非要使用文本编辑器或其他第三方开发环境来编写 C#代码，否则用户几乎肯定也希望使用 Visual Studio 2010。运行托管代码不需要安装完整的 SDK，但需要.NET 运行库。需要把.NET 运行库和代码分布到还没有安装它的客户端上。

本书内容

本书首先在第 1 章介绍.NET 的整体体系结构，给出编写托管代码所需要的背景知识，此后本书分几部分介绍 C#语言及其在各个领域中的应用。

第Ⅰ部分——C#语言

本部分给出 C#语言的背景知识。尽管这一部分假定读者是有经验的编程人员，但它没有假设读者拥有任何特殊语言的知识。首先介绍 C#的基本语法和数据类型，再介绍 C#的面向对象功能，之后是 C#中的一些高级编程主题。

第Ⅱ部分——Visual Studio

本部分介绍全世界 C#开发人员都使用的主要 IDE：Visual Studio 2010。本部分的两章探讨使用工具构建基于.NET Framework 4 的应用程序的最佳方式，另外，本部分还讨论项目的部署。

第Ⅲ部分——基础

本部分介绍在.NET 环境中编程的规则。特别是安全性、线程、本地化、事务、构建 Windows

服务的方式，以及将自己的库生成为程序集的方式等主题。

第Ⅳ部分——数据

本部分介绍如何使用 ADO.NET 和 LINQ 访问数据库，以及与目录和文件的交互。我们还详细说明.NET 对 XML 的支持、对 Windows 操作系统的支持，以及 SQL Server 2008 的.NET 功能。

第Ⅴ部分——显示

本部分讨论传统 Windows 应用程序的构建，在.NET 中这种应用程序称为 Windows 窗体。Windows 窗体是应用程序的胖客户端版本，使用.NET 构建这些类型的应用程序是实现该任务的一种快捷、简单的方式。本部分还阐述如何编写基于 Windows Presentation Foundation 和 Silverlight 的应用程序，如何编写在 Web 站点上运行的组件，如何编写网页。其中包括 ASP.NET 和 ASP.NET MVC 提供的许多新功能。

第Ⅵ部分——通信

这一部分介绍通信，主要论述独立于平台使用 Windows Communication Foundation(WCF)进行通信的服务。通过消息队列，揭示了断开连接的异步通信。本部分还介绍如何利用 Windows Workflow Foundation(WF)、对等网络，以及创建联合源。

第Ⅶ部分——附录

这一部分介绍如何为 Windows 7 和 Windows Server 2008 R2 开发应用程序。

光盘所附章节

即使用这样一本厚书，也不能涵盖 C#以及使用这种语言和其他.NET 技术的所有内容，于是我们在本书附赠光盘放了全书的源代码和另外 10 章内容。这些章节包括各种主题的信息：GDI+(这种技术用于构建包含高级图形的应用程序)、在.NET 客户端和服务器之间通信的.NET Remoting、用于后台服务的 Enterprise Services 和 Managed Add-In Framework(MAF)。光盘中的其他主题包括 VSTO 开发和使用 LINQ to SQL。

如何下载本书的示例代码

在读者学习本书中的示例时，可以手工输入所有的代码，也可以使用本书附带的源代码文件。本书使用的所有源代码都可以从本书合作站点 http://www.wrox.com/和本书附赠光盘上找到。登录到站点 http://www.wrox.com/上，使用 Search 工具或书名列表就可以找到本书。接着单击本书细目页面上的 Download Code 链接，就可以获得所有的源代码。

注释：

许多图书的书名都很相似，所以通过 ISBN 查找本书是最简单的，本书的 ISBN 是 978-0-470-50225-9。

在下载了代码后，只需用自己喜欢的解压缩软件对它进行解压缩即可。另外，也可以进入

http://www.wrox.com/dynamic/books/download.aspx 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有代码。

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但是错误总是难免的，如果您在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

要在网站上找到本书的勘误表，可以登录 http://www.wrox.com，通过 Search 工具或书名列表查找本书，然后在本书的细目页面上，单击 Book Errata 链接。在这个页面上可以查看 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表，网址是 www.wrox.com/misc-pages/booklist.shtml。

如果在 Book Errata 页面上没有看到您找出的错误，请进入 www.worx.com/contact/techsupport.shtml，填写表单，发电子邮件，我们就会检查您的信息，如果是正确的，就在本书的勘误表中粘贴一个消息，我们将在本书的后续版本中采用。

p2p.wrox.com

P2P 邮件列表是为作者和读者之间的讨论而建立的。读者可以在 p2p.wrox.com 上加入 P2P 论坛。该论坛是一个基于 Web 的系统，用于传送与 Wrox 图书相关的信息和相关技术，与其他读者和技术用户交流。该论坛提供了订阅功能，当论坛上有新帖子时，会给您发送您选择的主题。Wrox 作者、编辑和其他业界专家和读者都会在这个论坛上进行讨论。

在 http://p2p.wrox.com 上有许多不同的论坛，帮助读者阅读本书，在读者开发自己的应用程序时，也可以从这个论坛中获益。要加入这个论坛，须执行下面的步骤：

(1) 进入 p2p.wrox.com，单击 Register 链接。

(2) 阅读其内容，单击 Agree 按钮。

(3) 提供加入论坛所需的信息及愿意提供的可选信息，单击 Submit 按钮。

(4) 然后就可以收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。

提示：

不加入 P2P 也可以阅读论坛上的信息，但只有加入论坛后，才能发送自己的信息。

加入论坛后，就可以发送新信息，回应其他用户的帖子。可以随时在 Web 上阅读信息。如果希望某个论坛给自己发送新信息，可以在论坛列表中单击该论坛对应的 Subscribe to this Forum 图标。

对于如何使用 Wrox P2P 的更多信息，可阅读 P2P FAQ，了解论坛软件的工作原理，以及许多针对 P2P 和 Wrox 图书的常见问题解答。要阅读 FAQ，可以单击任意 P2P 页面上的 FAQ 链接。

目　录

第Ⅰ部分　C# 语 言