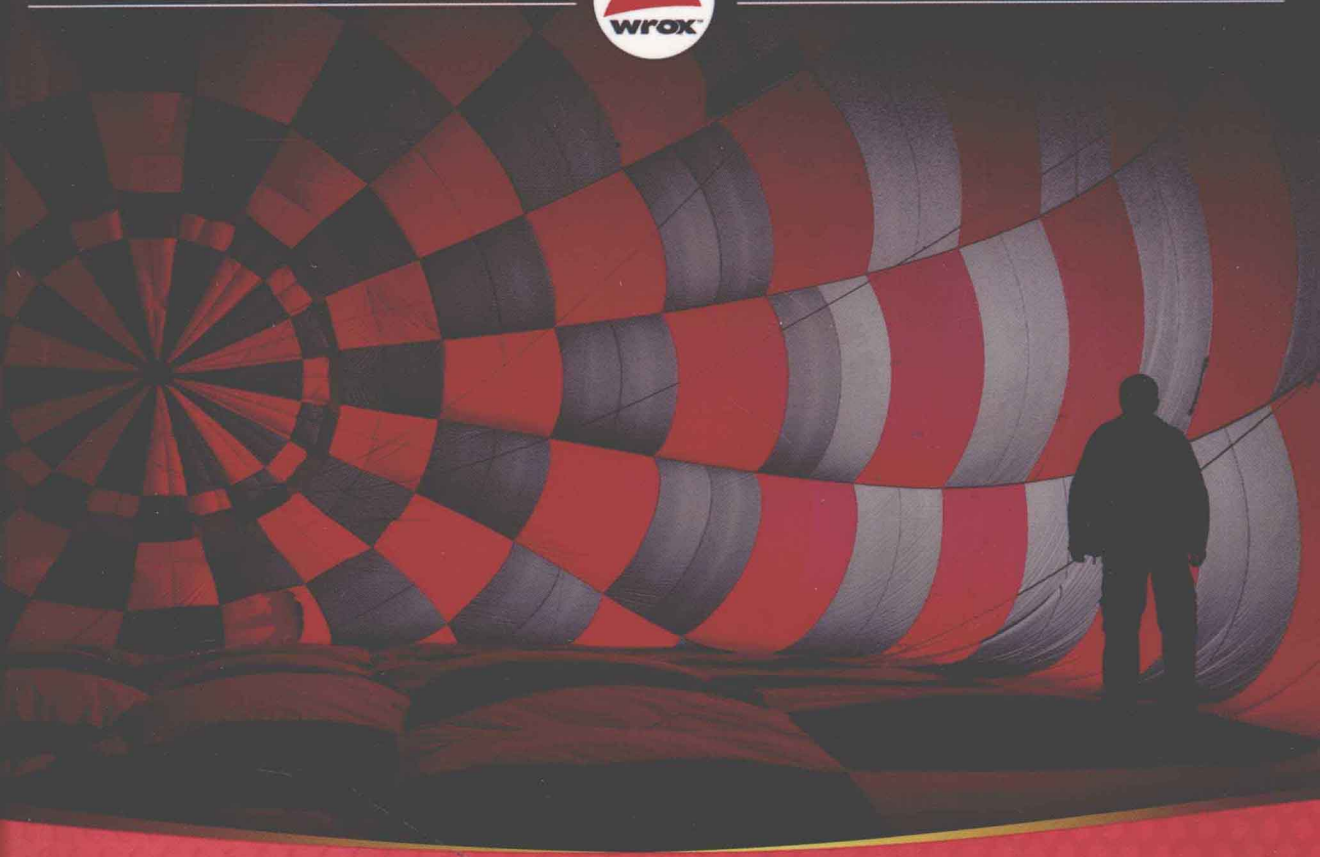
Join the discussion @ p2p.wrox.com

Wrox Programmer to Programmer™



Professional ASP.NET Design Patterns

ASP.NET设计模式

Microsoft首席Program Manager Scott Hanselman
(Microsoft RD, MVP)推荐作序

Microsoft Most Valuable Professional

(美) Scott Millett 著
杨明军 译

清华大学出版社

ASP.NET 设计模式

(美) Scott Millett 著

杨明军 译

清华大学出版社

北 京

Professional ASP.NET Design Patterns
EISBN：978-0-470-29278-5

北京市版权局著作权合同登记号 图字：01-2011-6891

图书在版编目(CIP)数据

ASP.NET 设计模式/(美)米里特(Millett, S.)著；杨明军 译　—北京：清华大学出版社，2011.11
书名原文：Professional ASP.NET Design Patterns
ISBN 978-7-302-26702-7
Ⅰ. A…　Ⅱ. ①米…　②杨…　Ⅲ. 网页制作工具－程序设计　Ⅳ. TP393.092

中国版本图书馆 CIP 数据核字(2011)第 181074 号

责任编辑：王　军　于　平
装帧设计：孔祥丰
责任校对：成凤进
责任印制：杨　艳
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：43.5　字　数：1167 千字
版　　次：2011 年 11 月第 1 版　　印　　次：2011 年 11 月第1 次印刷
印　　数：1～4000
定　　价：79.80 元

产品编号：031654-01

译 者 序

按照传统 ASP.NET 的开发方式，采用微软的 RAD 开发工具 Visual Studio.NET 快速开发表单式 Web 应用程序是一种理想的选择。通过简单的拖曳和所见即所得的应用程序设计界面，使得人们能够快速上手，一致的编程模型也有利于桌面应用程序开发者向 Web 应用程序开发转移。此外，由于编码模式与设计模式可以简单地进行切换，平面设计师在设计阶段就能够看到与运行时接近的界面，而不必频繁地运行调试模式或刷新网页，这使平面设计师能全程参与应用程序开发，从而提高了开发效率。然而，这种固化的表单式应用程序设计模式也存在先天不足。随着业务需求的变化和规模的不断增长，如果仍然把所有的业务逻辑放在后置代码中，将使代码日益臃肿，而且存在大量的重复代码。同时，这种 Web 表单式设计也不利于在应用程序中采用 AJAX 技术，很难在 Web 表单和 Web 服务程序之间共享代码。

针对这些开发问题，人们提出了多种解决办法。通过更灵活的应用程序设计框架、更细致的应用程序分层设计、更多地采用经过实践检验的模式和原则，改进应用程序的代码结构。本书系统地介绍了在解决 ASP.NET 应用程序开发问题的过程中涉及的各种设计模式和设计原则，对编写具有更好结构的代码具有很大的启示。如果读者了解 Java 开发中普遍使用的 Spring、Struts、Hibernate 等框架/技术，那么对本书的内容应该会比较熟悉。此外，书中采用了多种流行的开源工具来解决实际的问题，对于实际的编码工作也有极好的借鉴作用。

值得一提的是，随着.NET 4 框架的发布，微软推出的 ASP.NET MVC 框架也日趋成熟。阅读本书并掌握书中提到的设计理念和实现工具，对于更好地理解 ASP.NET MVC 框架中的概念颇有益处。

译 者

序

从建造房屋、制造厂生产产品到装配线组装汽车，所有这些都得益于知名的、已被认可的构建事物的模式。对于那些已经有很好解决方法的任务，没有理由再去进行重复劳动。

著名建筑学家克里斯托弗·亚历山大曾经说过：

每种模式描述了一个在我们周围不断重复发生的问题，以及该问题解决方案的核心，这样你就可以一次又一次地使用该方案而不必做重复劳动。

GoF(Gang of Four，稍后就会学习相关内容)为软件工程师写了第一本有关设计模式的书籍时，是该学科第一次正式地表达模式。在本书中，我们不仅会学习模式，还将学习反模式以及从中吸取什么教训。

有时候最佳实践并不总是那么清晰，而在 ASP.NET 中将设计模式映射到真实的用例并非易事。Scott Millett 的这本著作介绍了一些经过时间检验的设计模式，讲解如何阅读它们，然后以一种具体的、明确的方式运用它们来解决 ASP.NET 程序员每天都必须处理的问题。

与那些乏味空洞的作品不同，本书包含大量的代码，作者努力将各个知识点串接起来，让这些模式在开发者的日常工作中变得鲜活起来，可加以运用并取得实际效果。在这个过程中，Millett 不仅讲解了微软的 ASP.NET 现有的模式，而且还展示了开源软件中的一些佼佼者，如 Castle ActiveRecord、StructureMap、AutoMapper 和 NHibernate。

从 GoF 到 Robert C Martin(又称 Bob 大叔)的 S.O.L.I.D.原则，再到 Fowler 的企业模式，Scott(多么可爱的名字)将这些永恒的模式与现今的最新技术(如 jQuery 和 JSON、Entity Framework 以及 WCF)结合到一起。

阅读这本佳作是一次愉悦的体验，希望您也一样。

Scott Hanselman
微软公司程序经理
http://hanselman.com，Twitter 账号@shanselman

作 者 简 介

Scott Millett 是一位就职于伦敦 Wiggle.co.uk 公司的企业软件架构师，Wiggle.co.uk 是一家专门从事自行车和铁人三项体育运动业务的电子商务公司。他从 1.0 版就开始从事.NET 开发，并于 2010 年被授予 ASP.NET MVP 称号。他还是 Wrox 出版社出版的 *Professional Enterprise .NET* 一书的作者之一。在从事.NET 写作和开发工作之余，他会参加格拉斯顿伯里音乐节以及夏季英国所有的大型音乐节，放松一下，欣赏音乐。如果希望与 Scott 讨论本书或任何与.NET 开发有关的话题，或英国音乐节实况，那可以给他的邮箱 scott@elbandit.co.uk 发邮件，或访问他的 Tweet 账号@ScottMillett。

致　　谢

我要感谢 Brian Herrmann、Paul Reese 以及所有帮助我完成本书创作的 Wrox 员工。我还要感谢 Joe Fawcett，他出色地完成了技术编辑工作。

万分感谢 Imar Spaanjaars(http://imar.spaanjaars.com/)牺牲个人时间审阅本书并给我提供了极好的反馈。

我还想利用这个机会感谢几个人，在过去几年中我从他们那里学到了很多有益的思想。我在 2009 年夏季参加了 JP Boodhoo(http://blog.jpboodhoo.com/)的.NET 训练营，在那里的几周时间可能是我曾经度过的最受鼓舞的一段时光，而且让我明白为什么钟爱自己的工作。感谢 JP。

当 MVC 面世时，Rob Conery(http://blog.wekeroad.com/)开始撰写有关创建 MVC 网店的系列博客文章。他在网店的构建过程中研究了许多了不起的技术和方法学，包括 BDD、TDD、DDD、KanBan 和持续集成等。我所学到的知识超过自己的想象，这主要归功于 Rob 的表达方式契合实际而且非常有趣。如果本书能够达到这些视频一半的效果，我就十分满足了。Rob 创建了一家专门为开发者提供极佳视频资源(www.tekpub.com/)的公司。强烈推荐访问该网站！

前　言

本书将向您展示如何在实际的 ASP.NET 应用程序中发挥设计模式和核心设计原则的威力。本书的目标是向开发者讲解能够帮助其成为更好的程序员的面向对象编程基础、设计模式、原则及方法学。设计模式和原则支持松散耦合、高内聚的代码，而这将提升代码的可读性、灵活性和可维护性。每一章内容关注企业 ASP.NET 应用程序中的一层，并展示如何利用那些经过实践证明的模式、原则和最佳实践来解决问题并改进代码设计。此外，本书使用一个专业级的、完整的研究案例来讲解如何在实际的网站中运用最佳实践设计模式和原则。

读者对象

本书是为那些熟悉.NET 框架但希望了解如何改进编码方式以及如何运用设计模式、设计原则和最佳实践来提高代码的可维护性和适应性的 ASP.NET 开发者而写的。那些以前已经体验过设计模式的读者可能希望跳过本书的第 I 部分，这部分介绍了 GoF 提出的设计模式以及其他常见设计原则，包括 S.O.L.I.D 原则和 Martin Fowler 的企业设计模式。所有的代码示例均采用 C#语言编写，但这些概念可以非常轻松地用于 VB.NET。

主要内容

本书涵盖了开发企业级 ASP.NET 应用程序的知名模式和最佳实践。本书用到的模式可以用于从 ASP.NET 1.0 到 ASP.NET 4.0 的任何版本。不必管模式本身所用的语言，可以将模式用于任何面向对象编程语言。

结构安排

本书既可以作为一个分步推进的指南，也可以作为闲暇时随意翻阅的常备参考书。本书分为 3 个部分。第 I 部分介绍了模式和设计原则。第 II 部分讲解如何在 ASP.NET 应用程序的不同层中使用模式和原则。第III部分展示了一个完整的研究案例，用来演示本书涵盖的多个模式。既可以在阅读研究案例之前通读各章内容；也可以首先阅读研究案例，然后在涉及具体的模式和原则时再回过头来查阅第 II 部分以获取更详细的内容，这种紧密结合实际的方法可能会让学习过程变得更加轻松。

第 I 部分　模式与设计原则

本书第 I 部分首先介绍设计模式概念、企业模式及设计原则，包括 S.O.L.I.D.设计原则。

第 1 章：成功应用程序的模式

该章研究了专业开发者为何需要理解设计模式和原则，以及(更重要的是)如何在实际的企业级应用程序中加以利用。该章讲解了 GoF 设计模式的起源、它们在当今世界中的关联性以及与具体编程语言脱钩。然后浏览了一些常见设计原则和 S.O.L.I.D.设计原则，最后描述了 Fowler 的企业模式。

第 2 章：剖析模式的模式

该章介绍了使用模式模板所需的实用知识以及如何利用设计模板来阅读 GoF 设计模式。然后将教您如何理解设计模式分类，并讲解如何选择和应用设计模式。最后给出了一个示例来演示如何重构现有的代码，以便使用设计模式和原则来提升可维护性。

第 II 部分　剖析 ASP.NET 应用程序：学习并应用模式

本书的第 II 部分演示如何将前两章介绍的模式和原则运用到企业级 ASP.NET 应用程序的各个层次中。

第 3 章：应用程序分层与关注点分离

该章描述了分层设计与传统的 ASP.NET Web 表单代码隐藏模型相比所具有的优势。该章继续讲解逻辑分层和应用程序关注点分离的概念。然后定义了企业级 ASP.NET 应用程序中各个不同层次的职责，这部分的其他几章将讨论这些层次。该章最后是一个练习，将一个 Smart UI 反模式按照分层体系结构方法进行重构。

第 4 章：业务逻辑层：组织

该章涵盖了为组织业务逻辑层而设计的模式。该章首先描述了 Transaction Script 模式；然后是 Active Record 模式，并利用一个使用 Castle Windsor 项目的练习来演示该模式；最后一个模式是 Domain Model 模式，用 NHibernate 练习进行演示。该章最后评论了领域驱动设计(domain-driven design，DDD)方法学，以及如何运用该方法学将精力集中在业务逻辑而非基础设施。

第 5 章：业务逻辑层：模式

第 5 章与第 4 章类似，仍然介绍业务层，但该章关注的是构建对象时可以使用的模式和原则，以及如何确保构建可伸缩、可维护的应用程序。该章涉及的模式包括 Factory、Decorator、Template、State、Strategy 和 Composite。所涉及的企业模式包括 Specification 和 Layer Supertype。最后介绍了一些能够改进代码可维护性和灵活性的设计原则，包括 Dependency Injection、Interface Segregation 和 Liskov Substitution 原则。

第 6 章：服务层

该章介绍了服务层在企业 ASP.NET 应用程序中扮演的角色。该章首先简要地介绍了 Service

Oriented Architecture 及其必要性。然后讨论了 Façade 设计模式。接着讨论了 Document Message、Request-Response、Reservation 和 Idempotent 模式等 Messaging 模式。最后给出了一个 WCF 实体的练习，来演示本章涵盖的所有模式。

第 7 章：数据访问层

如何利用数据存储来使业务对象状态持久化是应用程序体系结构的关键部分。该章将学习该层中使用的设计模式，以及如何将它们整合在一起。这里演示了两种帮助组织持久层的数据访问策略：Repository 和 Data Access Object。接着，介绍了一些有助于优雅地满足数据访问需求的企业模式和原则，包括 Lazy Loading、Identity Map、Unit of Work 和 Query Object。之后介绍了 Object Relational Mapper 以及它们解决的问题。最后，给出了一个企业 Domain Driven 练习，它的 POCO 业务实体同时使用了 NHibernate 和 MS Entity Framework。

第 8 章：表示层

该章介绍了一些为了组织表示层逻辑并让其与应用程序中的其他层保持分离状态的模式。该章首先解释如何使用 Structure Map 和 Inversion of Control 容器将松散耦合的代码连接起来。然后描述了几种表示层模式，包括利用 Model-View-Presenter 模式和 ASP.NET Web 表单实现视图，利用 Command 和 Chain of Responsibility 模式实现 Front Controller 表示模式，以及利用 ASP.NET MVC 框架和 Windsor 的 Castle Monorail 框架来实现 Model-View-Controller 模式。最后讨论的表示层模式是 ASP.NET Web 表单中使用的 PageController。该章最后讲解一种可与组织模式一起使用的 ViewModel 模式，以及如何利用 AutoMapper 自动地把领域实体映射到 ViewModel。

第 9 章：用户体验层

在第 II 部分的最后一章介绍用户体验层。该章首先讲解什么是 Ajax 及其所依赖的技术。然后讲解了 JavaScript 库，说明如何使用 jQuery 等强大代码库来简化 JavaScript 处理。该章主要描述了一些常见的 Ajax 模式：Ajax Periodic Refresh 和 Timeout 模式，用于维护历史的 Unique URL 模式，用于实现客户端数据绑定的 Jtemplate 模式，以及 Ajax Predictive Fetch 模式。

第III部分　案例研究：在线电子商务商店

本书最后一部分使用一个完整的示例应用程序来演示在第 II 部分中介绍的众多模式。

第 10 章：需求和基础设施

这是关于案例研究的第 1 章，介绍了将在其余 4 章中构建的 Agatha 电子商务商店。该章描述了该网站的需求，以及将要用到的基础设施和总体架构。表示层采用 ASP.MVC，中间层的组织采用领域模型，利用 NHibernate 使业务实体持久化并从数据库中检索业务实体。

第 11 章：创建商品目录

该章构建了商店的商品目录浏览功能。大量使用 jQuery 来提供丰富的 Web 2.0 观感。利用 JSON 来实现控件和 ASPX 视图之间的通信以提供 Ajax 功能。使用 ViewModel 为控件提供扁平的领域视图。采用 AutoMapper 自动地把领域实体转换成 ViewModel。

第 12 章：实现购物车

该章实现了顾客购物车。使用顾客 Cookie 来存放购物车内容摘要，创建一项服务将访问 Cookie 存储的逻辑抽象出来。为了保持 Web 2.0 观感，购物车上执行的所有动作均通过 Ajax 调用进行。

第 13 章：顾客会员

该章解决顾客会员资格和身份验证问题。使用 ASP.NET 用户凭据提供者进行就地身份验证；但也使用了第二种身份验证方法，让顾客使用其现有的基于 Web 的账号(比如 Facebook 和 Google 账号)来进行身份验证。本章还会开发顾客账号屏幕。

第 14 章：订购与支付

在案例研究练习的最后一章中，将研究该网站的支付和结账功能。本章选中的支付机制是 PayPal，但把逻辑代码抽象出来，这样就可以轻易地将其替换成任何其他支付手段。最后向顾客的账号部分中添加订单历史记录功能。

源代码

在阅读本书示例的过程中，可以选择采用手工方式录入所有代码，也可以选择本书所附的源代码文件。本书中所用的所有源代码均可以在 www.wrox.com 和 http://www.tupwk.com.cn/downpage 下载。在该网站，搜索本书的书名(可以通过搜索栏或使用书名列表)，然后在本书详细信息页面中单击 Download Code 链接来获取本书的源代码。

因为许多书籍都有着相似的书名，所以最简单的方式是按照 ISBN 搜索。本书的 ISBN 是 978-0-470-29278-5。

在下载代码之后，使用压缩工具解压。或者，可以打开 Wrox 主代码下载页面 http://www.wrox.com/dynamic/books/download.aspx，查看本书以及所有其他 Wrox 书籍的代码。

勘误表

我们尽力确保本书正文及代码正确无误。但人无完人，错误在所难免。如果发现本书中有错误之处，如拼写错误或代码错误，我们将非常感谢您的反馈。通过提交勘误信息，可以让其他读者免于枉费数小时宝贵时间，还能帮助我们提供更高品质的信息。

要查找本书的勘误页，请打开 www.wrox.com 网站，并使用搜索框或书名列表查找本书的书名。然后，在本书详细信息页面上，单击 Book Errata 链接。在打开的页面上，可以浏览已经由读者为本书提交的以及由 Wrox 编辑通告的所有勘误信息。还可以在 http://www.wrox.com/misc-pages/booklist.shtml 找到完整的包含每本书勘误信息链接的书籍列表。

如果没有在 Book Errata 页找到自己发现的错误，请打开 http://www.wrox.com/contact/techsupport.shtml 页面，填写其中的表单，将您发现的错误发送给我们。我们将检查该信息，如果确认存在该错误，我们将会向本书的勘误页发送一条消息并在本书后续版中纠正该问题。

P2P.wrox.com 网站

如果希望与作者和同行讨论，请加入 P2P 论坛 http://p2p.wrox.com。论坛属于基于 Web 的系统，可供提交关于 Wrox 书籍和相关技术的消息，并与其他读者和技术用户交流。论坛提供了一项订阅功能，当论坛上您所感兴趣的主题有新帖时，系统会通过电子邮件通知您。Wrox 作者、编辑、其他行业专家及同行读者都会出现在这些论坛上。

在 http://p2p.wrox.com 网站上，可以找到多个不同的论坛，不仅能够帮助您阅读本书，还能帮助您开发自己的应用程序。要加入论坛，只需要遵循以下步骤即可：

(1) 打开 http://p2p.wrox.com 并单击 Register 链接。

(2) 阅读使用条款并单击 Agree。

(3) 填写加入论坛时必需的信息以及任何您希望提供的可选信息，然后单击 Submit。

(4) 您将收到一封电子邮件，里面描述了如何验证自己的账号并完成加入过程。

虽然不加入 P2P 论坛也能阅读论坛中的消息，但要提交您自己的消息，必须加入。

一旦加入论坛，就可以提交新信息并回复其他用户提交的消息。可以在任何时间阅读 Web 上的消息。如果希望系统将特定论坛的新消息通过电子邮件发送给您，那么在论坛列表中的论坛名称旁单击 Subscribe to this Forum 图标。

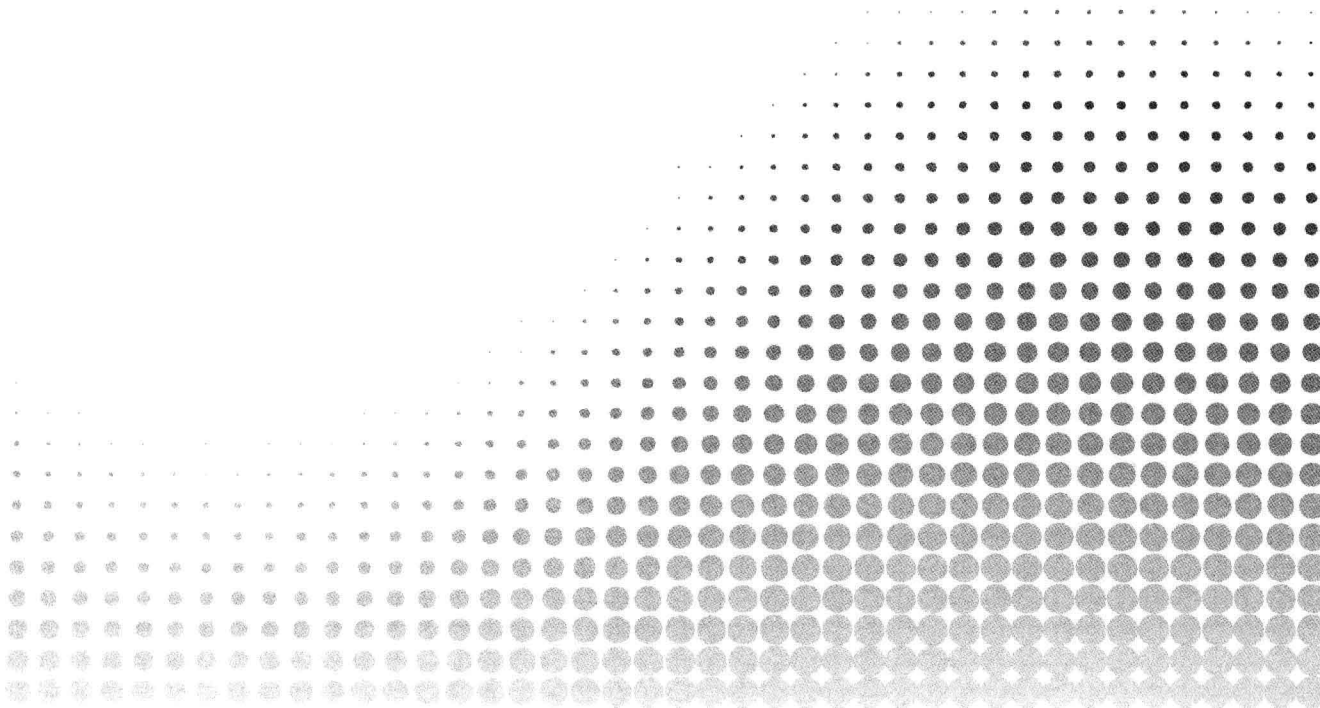
有关如何使用 Wrox P2P 的更多信息，阅读 P2P FAQ 以获得有关论坛软件如何运行以及许多有关 P2P 和 Wrox 书籍常见问题的答案。要阅读 FAQ，在 P2P 页面上单击 FAQ 链接。

目　录

第 I 部分

模式与设计原则

- 第 1 章：成功应用程序的模式
- 第 2 章：剖析模式的模式



第 1 章

成功应用程序的模式

本章内容：

- 介绍“四人组”(Gang of Four，GoF)设计模式
- 概述一些常见的设计原则和 SOLID 设计原则
- 描述 Fowler 企业模式

约翰·列侬曾经写道“没有问题，只有出路”。现在，虽然据我所知列侬先生从未从事 ASP.NET 编程，但是在软件开发甚至人性方面(但这超出了本书的研究范畴)，他所说的这句话却极为中肯。作为软件开发者，我们的工作涉及解决问题，而之前已经有其他开发者曾经无数次不得不解决这些问题，虽然这些问题披着各式的外衣。自从面向对象编程方法出现以来，人们已经发现、命名和归类许多模式、原则和最佳实践。了解了这些模式和常见解决方法词汇表，就可以着手将复杂问题分解，将不同部分封装起来，并采用经过检验的可信解决方案，以一种统一的方式来开发应用程序。

本书的目标是介绍可以运用到 ASP.NET 应用程序中的设计模式、原则和最佳实践。本质上，对于设计模式和原则，其语言可以是不可知的，因此可以把从本书学到的知识运用到 WinForm、WPF 和 Silverlight 应用程序以及其他优秀的面向对象语言。

本章将介绍设计模式的定义、起源以及学习它们的重要性。设计模式的基础是坚实的面向对象设计原则，本章将以 Robert Martin 的 S.O.L.I.D.设计原则为例来讲解这一点。本章还将介绍 Martin Fowler 的 *Patterns of Enterprise Application Architecture* 一书中提出的一些更高级的模式。

1.1 设计模式释义

设计模式是高层次的、抽象的解决方案模板。可以将这些模式视为解决方案的蓝本而不是解决方案本身。从中无法找到一种可以简单地运用到应用程序中的框架；相反，通常是通过重构自己的代码并将问题泛化来实现设计模式。

设计模式不仅适用于软件开发领域，从工程到建筑的所有领域都能够找到它的身影。实际上，模式的思想是由著名建筑大师 Christopher Alexander 在 20 世纪 70 年代引入的，目的是为设计讨论构