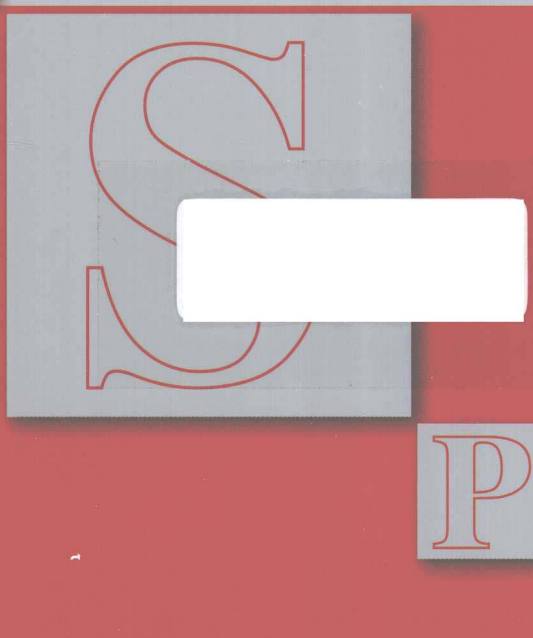
21世纪高等学校计算机专业实用规划教材

ASP.NET MVC 项目开发教程

朱　勇　主编



清华大学出版社

21世纪高等学校计算机专业实用规划教材

ASP.NET MVC 项目开发教程

朱　勇 主编

清华大学出版社
北京

内 容 简 介

本书讲述5个项目的开发过程，主要内容包括ASP. NET MVC 3编程技术、LINQ、ADO. NET实体框架、敏捷方法和用户故事、团队合作开发和TFS团队服务器的使用。

本书是从理论到实践的一体化教材，知识与技能紧密结合，项目难度适中，既可作为高职院校计算机相关专业的教材，也可作为初学者使用的入门书籍。

图书在版编目(CIP)数据

ASP. NET MVC项目开发教程/朱勇主编. --北京：清华大学出版社，2015
21世纪高等学校计算机专业实用规划教材
ISBN 978-7-302-39142-5

Ⅰ. ①A…　Ⅱ. ①朱…　Ⅲ. ①网页制作工具－程序设计－教材　Ⅳ. ①TP393.092

中国版本图书馆CIP数据核字(2015)第017691号

责任编辑：黄　芝　王冰飞
封面设计：何凤霞
责任校对：时翠兰
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　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
课 件 下 载：http://www.tup.com.cn，010-62795954

印 刷 者：北京季蜂印刷有限公司
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：14.25　　**字　　数**：350千字
版　　次：2015年7月第1版　　**印　　次**：2015年7月第1次印刷
印　　数：1～2000
定　　价：29.00元

产品编号：062503-01

前　言

ASP. NET MVC 是微软官方提供的以 MVC 模式为基础的 ASP. NET Web 应用程序框架。MVC 模式将应用程序的输入、处理和输出强制性地分离到 3 个相对对立的应用程序组件中。这种分离给复杂应用程序的管理、程序单元的独立开发与测试、团队环境下的分组开发都带来了极大的好处。MVC 模式已成为目前软件企业软件架构的首选技术。

本书体现了理实一体化和项目课程的教学理念，以工作任务为课程设置和内容选择的参照点，以项目为单位组织内容，并以项目活动为主要学习方式。书中的项目和任务的匹配模式结合了循环式和层进式的特点，项目从简单到复杂，每个项目的任务既有重复也有提高，符合学习的认知规律，循序渐进地将 ASP. NET MVC 项目开发的知识逐步引入项目。

本书以工作体系来安排知识和内容，并注重对职业技能的培养。实践先行，学习者可以按照任务实施步骤逐步实践，很快可以看到工作成果，以激发学习者的学习兴趣。完成工作任务后，再对工作过程中涉及的知识与技能进行分析，以完善学习者的知识体系。

本书共 5 个项目。第 1 个项目涉及 ASP. NET MVC 编程基础知识，主要内容包括控制器与视图的创建、ASP. NET MVC 路由机制、Razor 视图引擎和源代码管理。第 2 个项目引入了模型的概念，主要内容包括实体数据模型的创建、第三方组件的引用、LINQ、视图辅助方法等内容。第 3 个项目引入了敏捷方法与用户故事，主要内容包括敏捷方法的概念、用户故事的需求表达、团队开发、发布计划和迭代计划的管理、代码优先实体数据模型的创建、模型绑定与模型验证、授权管理等内容。第 4 个项目使用模型优先方式创建实体模型，主要内容包括基于模型优先的实体模型创建方式和多实体关联情况下的实体增删改查操作。第 5 个项目针对一个相对完整(包含前台与后台)的网站进行分析与开发，进一步加大模型的复杂性，主要内容包括自定义布局页、创建多实体关联实体数据模型、扩展方法、分布视图、MVC 区域等内容。

使用本书时的开发环境如下：

Microsoft Visual Studio 2010

Microsoft Visual Studio 2010 SP1

MVC 3 Framework

Microsoft SQL Server Compact 4.0(runtime+tools)

SQL Server express(optional)

服务器环境：

Microsoft SQL Server 2008 R2

Microsoft Team Foundation Server 2010

Microsoft Team Foundation Server 2010 sp1

本书可用作高职院校计算机相关专业的教材，也可用作 ASP. NET MVC 编程的初学者使用的入门书籍。本书读者需要先行了解网页设计、数据库技术、C＃编程等相关知识。

希望本书能对读者初学 ASP. NET MVC 编程有所帮助，并请读者对不当之处批评指正。

编　者

2015 年 3 月

目　录

项目一　Hello World

【项目解析】

“Hello, World”程序几乎是每一种计算机编程语言教程中最基本、最简单的程序，也通常是初学者所编写的第一个程序，其功能相当简单，主要是实现在程序运行界面上显示“Hello, World”。本项目旨在使初学者能初步掌握 ASP. NET MVC Web 应用程序的创建过程并熟悉程序的基本结构。

任务一　ASP. NET MVC 3 项目的创建

【技能目标】

- 学会创建 ASP. NET MVC 3 项目；
- 学会编译和运行项目。

【知识目标】

- 理解 Web 应用程序的工作原理；
- 了解 MVC 应用程序的架构模式；
- 理解 ASP. NET MVC 3 应用程序的工作过程；
- 理解 ASP. NET MVC 3 路由机制；
- 了解 ASP. NET MVC 3 应用程序的目录结构；
- 理解 ASP. NET MVC 3 命名约定。

一、任务实施

1. 启动 Visual Studio 2010

启动后的界面如图 1.1 所示。

2. 新建名为 MvcHelloWorld 的 MVC 3 项目

从“文件”菜单中选择“新建项目...”，打开图 1.2 所示的对话框，在左边选择 Visual C# 模板类别，在右边选择“ASP. NET MVC 3 Web 应用程序”项目模板，将项目名称改为 MvcHelloWorld，项目位置改为“D:\”(也可以保存在其他位置)，其他项保持默认，单击“确定”按钮。

如图 1.3 所示，在“新 ASP. NET MVC 3 项目”对话框中，选择“Internet 应用程序”，视图引擎选择 Razor，单击“确定”按钮。

如图 1.4 所示，VS 2010 为开发人员使用默认模板创建了一个 ASP. NET MVC 3 项目，所以在开始任何工作之前已经有了一个可以运行的默认应用程序。这是一个简单的

“Hello World!”项目，是开发应用程序的一个好的开始。

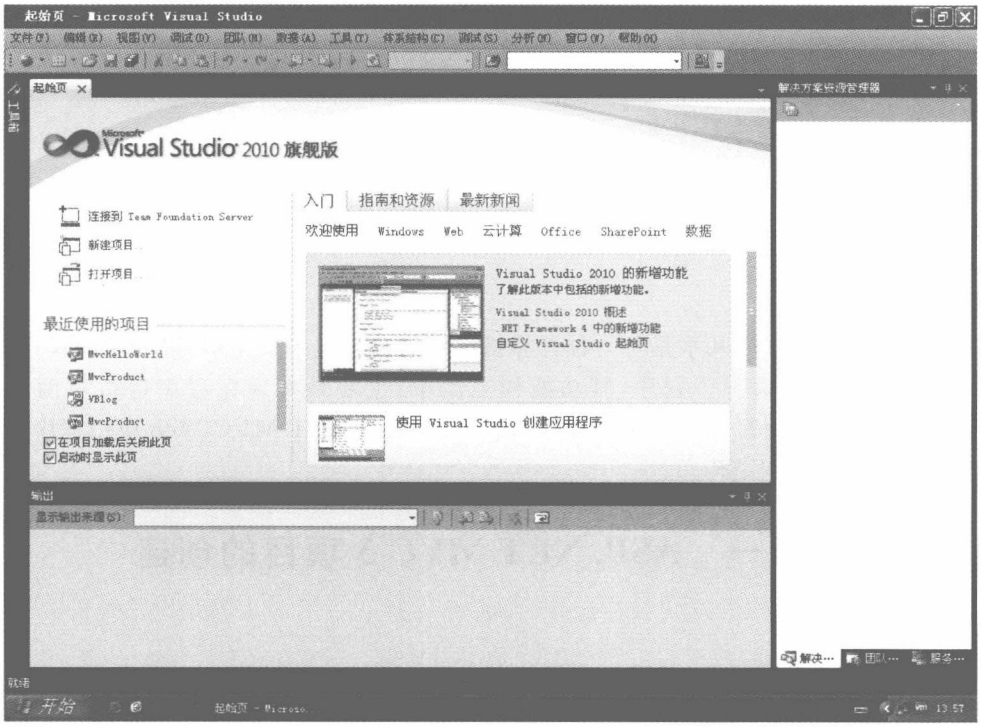


图 1.1　Visual Studio 2010 集成开发环境

图 1.2　新建项目对话框

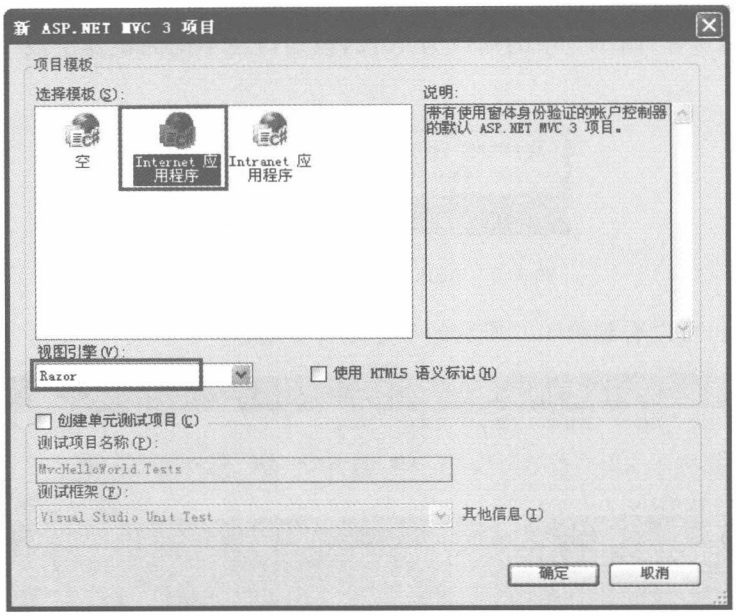


图 1.3　新 MVC 3 项目设置对话框

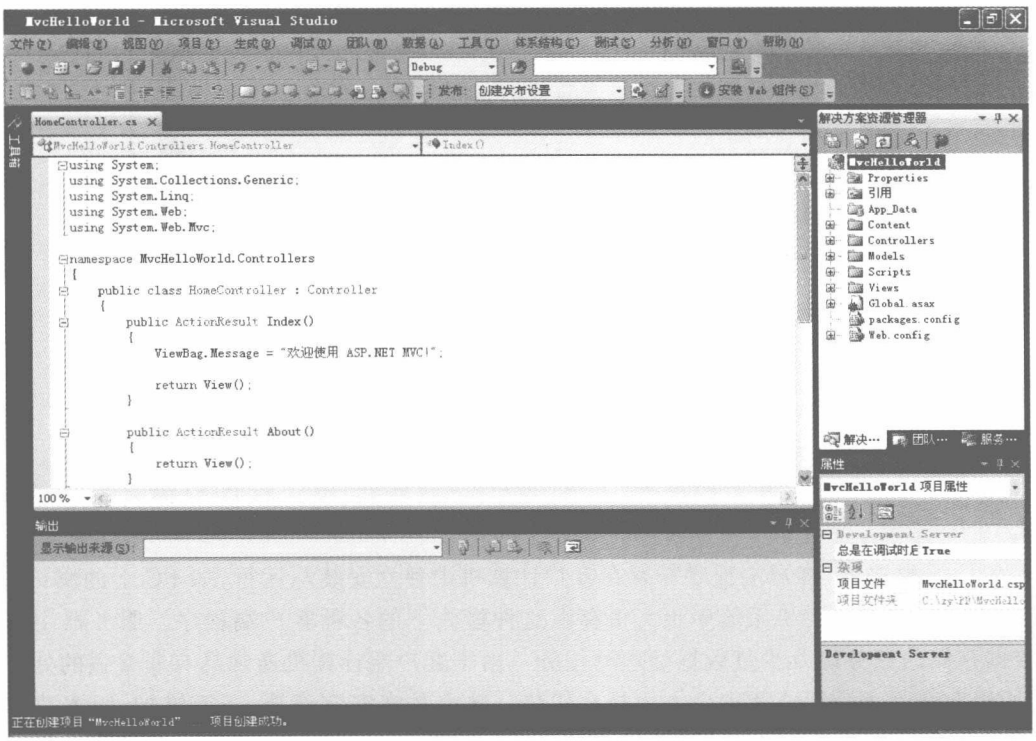


图 1.4　VS 2010 默认 MVC 3 项目结构

3. 运行 Web 应用程序

从“调试”菜单中，选择“启动调试”，或者直接按 F5 键。VS 2010 首先会编译整个项目，

并自动启动 ASP. NET Web 开发服务器。如图 1.5 所示，开发服务器启动后，会启用一个端口以提供本地服务，图 1.5 中的端口为 1072，该端口在不同的机器上或下次运行时可能都不一样。

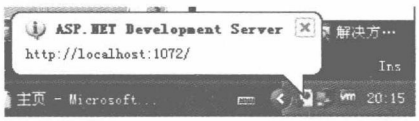


图 1.5　ASP. NET Web 开发服务器

调试运行后，可以看到图 1.6 所示的程序运行界面。

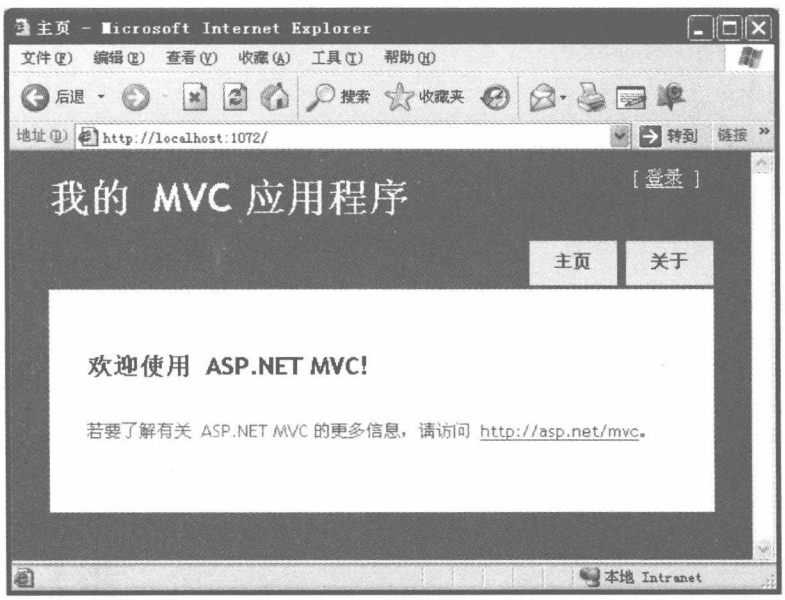


图 1.6　程序运行界面

二、相关知识

1. Web 应用程序

网络应用程序有两种架构模式：C/S(客户端/服务器)模式和 B/S(浏览器/服务器)模式。

在 C/S 模式下，客户端程序需要在客户计算机中独立安装与运行，如 PC 上的腾讯 QQ 客户端程序，现在的手机系统中也大量存在这种模式下的各种客户端程序。服务器主要负责数据存储的任务以及少量数据处理的任务。由于客户端计算机通常具有非常强的处理能力，所以在交互表现形式和安全方面具有优势。缺点是在安装部署、升级维护、版本兼容方面存在不足。一般适合程序使用场景比较固定和需要复杂的交互表现形式的场合。

B/S 模式实际上是一种特殊的 C/S 模式，客户端由浏览器(如：IE 浏览器)来担任，而浏览器是操作系统的标配，一般无须安装。B/S 模式是目前被广泛使用的一种模式，如网页版的 163 邮件系统。在这种模式下，用户通过在浏览器中输入 URL 来访问应用程序，数据

处理的大部分工作由服务器完成，服务器将处理结果以 HTML 的形式发送给浏览器，再由浏览器将 HTML 呈现给用户。浏览器的使用给应用程序的访问带来了很大的方便性，用户可以随时随地通过浏览器进行工作和娱乐，比较适合访问地点不固定的用户。由于客户端无需安装特定的应用程序，因此 B/S 模式具有升级维护方便的优势，但由于交互表现形式受到浏览器的限制，不太适合界面复杂的应用程序。

Web 应用程序是一种工作在 B/S 模式下，可以通过浏览器访问的应用程序。浏览器和服务器之间的通信一般借助于 HTTP 协议。

最简单的 Web 应用程序其实就是一些 HTML 文件和其他的一些资源文件组成的集合。Web 站点可以包含多个 Web 应用程序。它们位于 Internet 上的一个服务器中，一个 Web 站点其实就对应着一个网络服务器(Web 服务器)。

Web 应用程序的工作原理如图 1.7 所示。

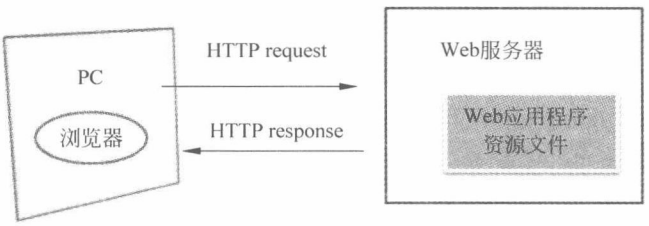


图 1.7　Web 应用程序工作原理

客户端浏览器通常通过 URL 来访问服务器提供的 Web 应用程序资源，如果访问的资源是静态资源(如. jpg 文件、. htm 文件等)，服务器直接将资源文件的内容返回给客户端，如果访问的是动态资源(如. aspx 文件、. jsp 文件等)，服务器将文件执行的结果返回给客户端。

当通过 URL 请求的服务器资源不存在时，服务器会返回给客户端一个 404 错误状态码，告诉浏览器被请求的资源并不存在。导致这种错误的原因可能是 URL 拼写错误或者被请求的资源文件被移动了位置。

2. MVC 模式

应用程序所做的事情不外乎以下几种：输入、输出、数据处理、数据存储。

MVC 是一种程序架构模式，全名是 Model View Controller，是模型(model)-视图(view)-控制器(controller)的缩写，它强制性地使应用程序的输入、处理和输出分开。MVC 将应用程序分成 3 个核心部件：模型、视图、控制器，它们各自处理自己的任务，其关系如图 1.8 所示。

“视图”是用户看到并与之交互的界面。对 Web 应用程序来说，视图就是由 HTML 元素组成的界面。

“模型”表示企业数据和业务规则。在 MVC 的 3 个部件中，模型拥有最多的处理任务。

“控制器”接收用户的输入并调用模型和视图去完成用户的需求，当单击 Web 页面中的超链接和发送 HTML 表单时，请求首先被控制器捕获。控制器本身不输出任何信息和做任何业务处理。它只是接收请求并决定调用哪个模型部件去处理请求，然后再确定用哪个视图来显示返回的数据。

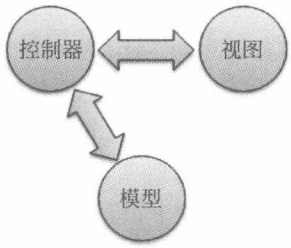


图 1.8　MVC 关系图

ASP. NET 支持 3 种不同的开发模式：Web Pages(Web 页面)、MVC(Model View Controller,模型-视图-控制器)、Web Forms(Web 窗体)。ASP. NET MVC 实现了 MVC 架构模式,并简化了 MVC 应用程序的开发过程。

在 ASP. NET MVC 中,MVC 的 3 个主要部件的定义如下：

- 模型：模型是描述程序设计人员感兴趣的问题空间的一些类,这些类通常封装存储数据库中的数据以及跟操作这些数据和执行特定业务逻辑有关的代码。在 ASP. NET MVC 中,模型就像是一个使用了某个工具的数据访问层(Data Access Layer),这个工具如实体框架(Entity Framework)。
- 视图：一个动态生成 HTML 页面的模板。
- 控制器：一个协调视图和模型之间关系的特殊类。它响应用户输入,与模型进行交互,并决定呈现哪个视图。在 ASP. NET MVC 中,这个类文件的名称通常以 Controller 结尾。

3. URL 路由

在传统 Web 应用程序中(如 ASPX、JSP、PHP 等),URL 表示一个磁盘上的文件。例如,当看到“http://www. yzpc. edu. cn/jwc/index. aspx”这个 URL 时,可以很确定地说在 Web 站点上肯定有一个 jwc 目录,在 jwc 目录下一定有一个文件名为 index. aspx 的文件。在这种情况下,URL 与磁盘文件存在着一种对应关系。如果指定的 URL 所对应的文件在服务器磁盘上不存在,浏览器会收到服务器返回的 404 错误。

在 ASP. NET MVC Web 应用程序中,URL 被映射为对一个类的方法调用,而不是服务器磁盘文件。被映射的类称为控制器(Controller)类,被调用的方法称为操作(Action)方法。每一个 ASP. NET MVC Web 应用程序至少需要一个路由来说明 URL 如何映射到 Controller 类及 Action 方法。一个 ASP. NET MVC Web 应用程序中可以有多个路由,这些路由存储在路由集(RouteCollection)中,它们共同决定了请求 URL 如何映射到一个资源。

创建 MVC 项目后,在 Global. asax. cs 文件的 Application_Start 方法中通过调用 RegisterRoutes 方法定义了一个默认路由并将其加入到路由集中,部分代码如代码清单 1.1 所示。

代码清单 1.1

```
20    public static void RegisterRoutes(RouteCollection routes)
21    {
22        routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
23
24        routes.MapRoute(
25            "Default", // 路由名称
26            "{controller}/{action}/{id}", // 带有参数的 URL
27            new { controller = "Home", action = "Index",
28                id = UrlParameter.Optional } // 参数默认值
29        );
30
31    }
```

第 26 行代码中的“{controller}/{action}/{id}”是 URL 模式。这是一种模式匹配规则，用来决定该路由是否适用于传入的 URL 请求。针对本例，URL 模式被“/”分割成 3 段，每个段都包含了一个由一对花括号定义的 URL 参数，因此，示例中定义的规则可以匹配任何带有 3 个段的 URL。当该路由与带有 3 个段的 URL 匹配时，URL 第 1 个段中的文本对应于{controller}参数，同理，第 2 个段的文本对应于{action}参数，第 3 个段的文本对应于{id}参数。

URL 参数可以命名为任何想要的参数，但是为了保证程序能正确的运行，ASP. NET MVC 框架要求参数中必须包含{controller}和{action}参数。

第 27～28 行代码中的“new { controller = "Home", action = "Index", id = UrlParameter. Optional }”定义了路由默认值，它与 URL 模式共同决定了如何匹配 URL 请求。前面的 URL 模式决定了本例中的路由只能匹配含有 3 个段的请求 URL，而在图 1.6 的运行界面中发现 URL 为“http://localhost:xxxx/”，去除协议、主机地址和端口后，这个 URL 不包含任何段，但程序为什么依旧可以正常运行呢？这与路由默认值有关。本例中的路由默认值定义了在请求 URL 中不包含任何段的情况下的默认取值，也就是说这些段在缺失时各用什么默认值来取代。在本例中，当{controller}参数段缺省时的取值为 Home，当{action}参数段缺省时的取值为 Index，{id}参数段可有可无。表 1.1 说明在定义了该路由默认值的情况下的几种可能的匹配情况。注意，缺省的顺序只能是从右向左缺省。

表 1.1　URL 模式匹配举例

URL 模式	传入 URL	匹　配　值
{controller}/{action}/{id}	/	controller＝HomeController action＝Index id＝null
	/Hello	controller＝HelloController action＝Index id＝null
	/Hello/Welcome	controller＝HelloController action＝Welcome id＝null
	/Hello/Welcome/tom	controller＝ HelloController action＝ Welcome id＝tom

根据 ASP. NET MVC Web 应用程序的运行特点，每一个 URL 请求必须要能够解析出相应的{controller}和{action}。{controller}参数的值用来实例化一个 Controller 类来处理请求，根据约定，MVC 系统会通过在{controller}参数值的后面添加一个 Controller 后缀的方式来确定控制器类的名称，并试图在程序中寻找以该名称为类名的控制器类。例如，如果请求的 URL 是“http://localhost:xxxx/Home/Index”，那么 MVC 会试图寻找一个名为 HomeController 的类，如果找不到，服务器会返回 404 错误。

注意

这里需要注意的是，在 URL“http://localhost:xxxx/Home/Index”中 xxxx 是服务端口号。由于 ASP. NET 的本地开启服务器时每次启用的端口号可能不同，在实际运行时是一个 4 位数字的随机端口号，如 1072。这个端口号在不同的机器上有可能不一样。在下文中统一使用 xxxx 来表示端口号。

{action}参数的值用来决定调用控制器类的哪个操作方法，以 URL“http://localhost:xxxx/Home/Index”为例，MVC 系统会调用 HomeController 类的 Index 操作方法，如果该类或方法不存在，服务器也会返回 404 错误。

根据 ASP. NET MVC Web 应用程序中 URL 路由的规则，图 1.6 中的 URL“http://localhost:1072/”应该等同于 URL“http://localhost:1072/Home/Index”，其映射过程如图 1.9 所示。

图 1.9　URL 映射过程

4. ASP. NET MVC 应用程序结构

如图 1.10 所示，当用 VS 2010 创建一个新的 ASP. NET MVC 3 应用程序项目时，VS 2010 会自动向项目中添加一些目录和文件。

默认的 ASP. NET MVC 项目有 6 个顶层目录，如表 1.2 所示。

- /Controllers 目录，展开该目录，将会发现 Visual Studio 默认向该项目中添加了两个 Controller 类：HomeController 和 AccountController 类。
- /Views 目录，展开该目录，将会发现 3 个子目录（Home、Account 和 Shared）以及一些模板文件。
- /Content 和/Scripts 目录，展开这两个目录，将会发现一个 Site. css 文件和 JavaScript 库文件。这些文件属于站点的静态资源。

图 1.10　MVC 应用程序默认结构

表 1.2　MVC 项目默认顶层目录

目录	说　明
/Controllers	该目录中存放处理 URL 请求的控制器类
/Models	该目录中存放处理业务逻辑和数据的类
/Views	该目录中存放视图模板文件
/Scripts	该目录中存放 JavaScript 文件
/Content	该目录中存放 css 和图片文件以及其他静态资源
/App_Date	该目录中存放数据库文件

5. 命名约定

ASP. NET MVC 3 在很大程度上依赖于约定，这可以大大减少开发者花在程序配置方面的时间。ASP. NET MVC 3 使用了如下约定。

- ◇ 每个控制器类的名称都以 Controller 结尾。例如，HelloController、HomeController 等。这些控制器类都放在 Controllers 目录中。
- ◇ 应用程序中的所有视图模板文件都放在 Views 目录中。
- ◇ 控制器所使用的视图模板文件都放在 Views 目录下的以控制器命名的目录中。例如，HomeController 控制器使用的视图默认在/Views/Home 目录中。
- ◇ 所有共享视图文件放在/Views/Shared 目录中。

任务二　控制器的创建

【技能目标】

- ➢ 学会创建控制器；
- ➢ 学会编写操作方法；
- ➢ 学会使用 URL 给控制器传递数据的方法。

【知识目标】

- 理解控制器的概念及其在 MVC 架构中的角色；
- 理解操作方法的概念；
- 理解操作参数的作用。

一、任务实施

1. 新建控制器

如图 1.11 所示，在“解决方案资源管理器”窗口中的 Controllers 文件夹上右击，然后依次选择“添加→控制器(T)”菜单项。

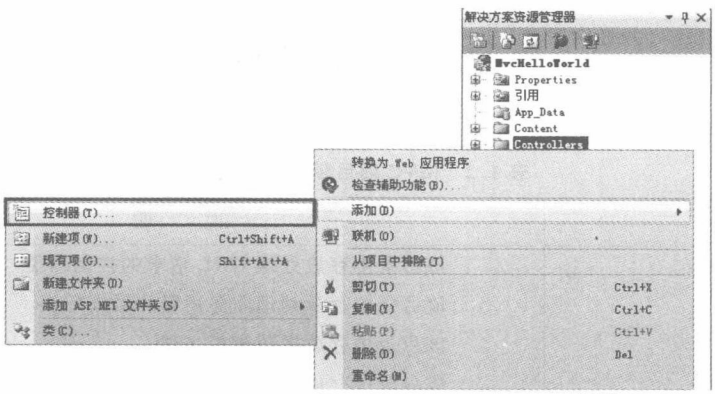


图 1.11　新建控制器操作界面

注意

在调试运行时无法对项目进行修改，所以在修改项目中任何文件之前先要停止调试。

如图 1.12 所示，在“添加控制器”对话框中，将控制器名称改为 HelloController，模板选择“空控制器”，单击“添加”按钮。

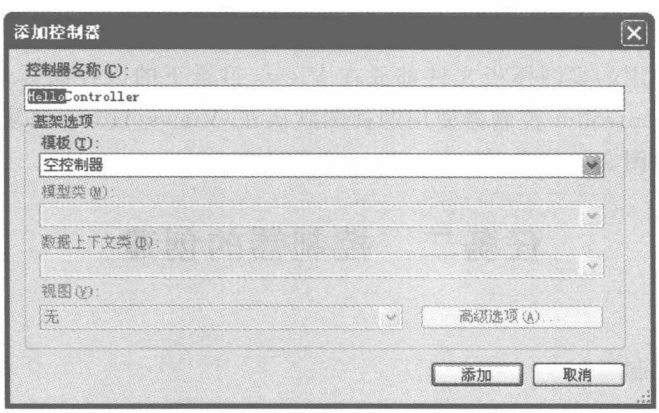


图 1.12　“添加控制器”对话框