****

本科毕业论文（设计）

**英文文献翻译**

文献英文题目 ASP.NET MVC5 入门

学 生 姓 名 徐雯俊

学 号 2014053030

所 在 学 院 计算机学院

专 业 数字媒体技术

班 级 数媒2014级1班

指 导 教 师 廖德钦（副教授）

2018 年 1月

成都信息工程大学 计算机学院

ASP.NET MVC5 入门

徐雯俊

（作者单位，四川省 成都市 邮编610225）

摘要：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(Web Application)框架,它由Castle的MonoRail而来

关键词：ASP.NET；MVC；VS

中图分类号：G4 文献标识码：CP

# 简述

本教程教授具有控制器和视图的ASP.NET Core MVC。Razor页面Razor 页面是 ASP.NET Core 2.0 中的一个新的选择，它是基于页面的编程模型，可以让Web UI的开发更简单、更高效。我们建议先尝试[Razor页面](https://docs.microsoft.com/aspnet/core/mvc/razor-pages)教程，再使用 MVC 版本。Razor页面教程：易于关注,涵盖更多功能,是开发新应用程序的首选方法。

本教程将教授你通过使用Visual Studio 2017生成ASP.NET MVC 5 Web 应用程序所使用的基础知识，本教程的最终开源地址位于GitHub。

你需要一个Azure账户去将此应用程序部署到Azure。

你可以免费建立一个Azure账户-你可以获取信用额度来使用付费版Azure服务，甚至在试用结束后，你可以保留你的账户，继续使用免费的Azure服务。

你可以激活MSDN订阅权益-你的MSDN订阅为你提过的信用额度可以用来支付Azure服务。

# 入门介绍

首先，安装并运行Visual Studio 2017。

Visual Studio是一个IDE,或者说是一个集成的开发环境。就像你使用Microsoft Word来写文档一样，你将使用一个IDE来创建应用程序。在Visual Studio中，在底部有一个列表向你提供可用的功能，同时也有一个菜单提供另一种在IDE中执行任务的方法。（例如：除了在开始界面选择新项目外，你也可以使用文件 > 新项目）。

# 创建第一个应用程序

点击新项目，然后选择位于左侧的Visual C#，然后选择Web，之后再选择ASP.NET Web Application(.NET Framework)，将你的项目命名为"MvcMovie"，最后点击确定。

在新建ASP.NET项目对话框中，点击MVC，然后点击确定。

Visual Studio在你建立的ASP.NET MVC项目中使用一个默认的模版，因此，你可以在不做任何操作的情况下得到一个可执行的项目，这是一个简单的"Hello World! "项目，这个项目是你开发自己的应用程序的起点。

点击 F5开始调试，F5让Visual Studio 启动一个IIS Express并运行你的web程序，然后Visual Studio 启动你的浏览器，并且打开应用程序主页。 你可以注意到，浏览器的地址栏显示的是"localhost:port#"，这个并不像"example.com"。这是因为在本机运行刚才生成的应用程序，localhost总是指向你自己的本地计算机，当Visual Studio运行一个web项目时，web服务器会使用一个随机的端口，在下图中，端口号是1234，当你运行应用程序时，你将会看到一个不同的端口号。

在导航栏的右侧，默认的模版为你提供了主页，联系人，关于的页面，下面这个图没有显示主页，关于和联系人的链接，根据你的浏览器的窗口大小，你可能需要点击导航图标去查看这些链接。

应用程序还提供了注册和登录，下一步要做的是改变此应用程序的工作原理，并且去了解一些关于ASP.NET MVC的知识，关闭ASP.NET MVC应用程序，让我们更改一些代码。

# 4、添加控制器

MVC代表模型，视图，控制器。MVC是一种让开发的应用程序能很好的设计，测试和维护的模式。基于MVC的应用程序包含：Models（用程序的数据和使用验证逻辑以强制实施针对这些数据的业务规则)，Views（用程序使用动态生成 HTML 响应的模板文件），Controllers（理传入的浏览器请求，类检索模型数据，然后指定将响应返回到浏览器的视图模板）。

我们将涵盖所有这些概念讲解本系列教程，向你展示如何使用它们来生成应用程序。

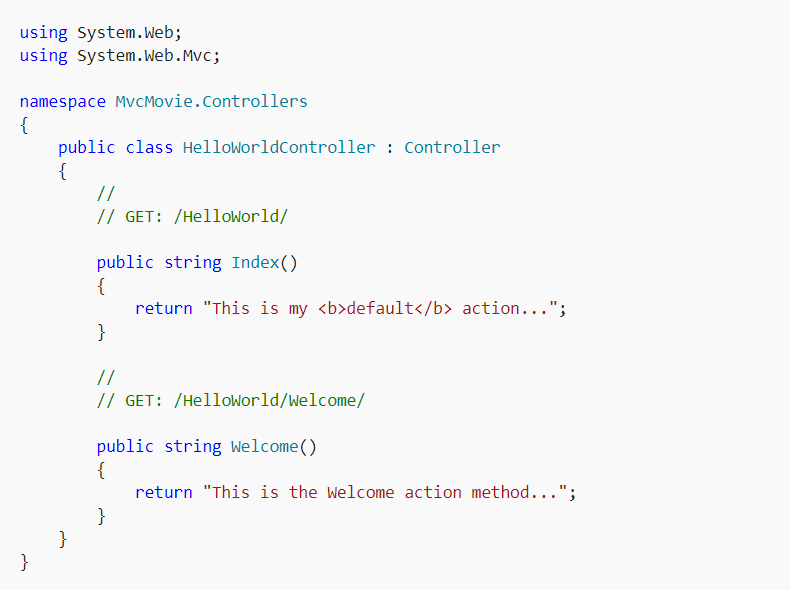
首先创建了控制器类。在解决方案资源管理器，右键单击控制器文件夹，然后单击添加，然后控制器。

在添加基架对话框中，单击MVC 5 控制器-空，然后单击添加。

将新控制器取名为"HelloWorldController"，然后单击添加。

请注意，在解决方案资源管理器，已经生成了一个命名为HelloWorldController.cs的文件和一个路径为Views\HelloWorld的新文件夹。

将文件的内容替换为以下代码。

图 1 示例代码

例如，控制器方法将返回 HTML 的字符串。控制器被命名为HelloWorldController和第一个方法命名为Index。让我们从浏览器调用它。运行应用程序（按 F5 或 Ctrl + F5）。在浏览器中，追加"HelloWorld"地址栏中的路径。(例如，在下图中的<http://localhost:1234/HelloWorld>。) 在浏览器页面将呈现如下所示的屏幕截图。在上面的方法中，代码直接返回一个字符串。你告知系统仅返回一些 HTML，它依之照办！

ASP.NET MVC 调用另一个控制器类 （使用不同的操作方法），具体取决于传入的 URL。 使用如下格式的 ASP.NET MVC 的默认 URL 路由逻辑来确定哪些代码调用：/[Controller]/[ActionName]/[Parameters]。

你可以在\_Start/RouteConfig.cs文件中设置路由的格式。

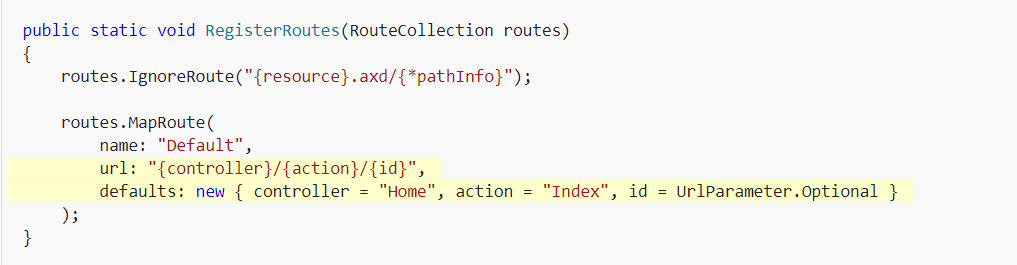


图 2 示例代码

当你运行应用程序，并且不提供任何 URL 段时，它默认为"Home"控制器和"Index"操作方法, 通过在上面的代码中的默认值部分中指定。

URL 的第一部分确定要执行的控制器类。 因此/HelloWorld映射到HelloWorldController类。 URL 的第二部分确定类上要执行的操作方法。 因此/HelloWorld/Index将执行HelloWorldController类的Index方法。 请注意，我们浏览到调用Index方法时默认情况下仅需要输入/HelloWorld。 这是因为方法Index是一个默认的执行方法，如果控制器未显式指定调用某个方法时。 URL 段的第三部分 (Parameters) 针对的是路由数据。 在本教程中，我们随后来看路由数据。

浏览 <http://localhost:xxxx/HelloWorld/Welcome>时。 Welcome方法将运行并返回字符串"This is the Welcome action method... "。 默认 MVC 映射是/[Controller]/[ActionName]/[Parameters]。 对于此 URL，采用 HelloWorld 控制器和 Welcome 操作方法。 目前尚未使用 URL 的 [Parameters] 部分。

让我们稍微修改该示例，以便你可以将某些参数信息从 URL 中传递到控制器 (例如， /HelloWorld/Welcome?name=Scott&numtimes=4)。 更改你Welcome方法以包括两个参数，如下所示。 请注意，代码将使用 C# 默认参数的功能，，如果该参数不传递任何值，numTimes参数应默认为 1。

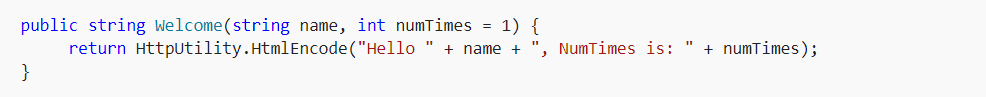


图 3 示例代码

运行你的应用程序并输入示例 URL (http://localhost:xxxx/HelloWorld/Welcome?name=Scott&numtimes=4)。 你可以在 URL 中尝试不同的值name和numtimes。[ASP.NET MVC 模型绑定系统](http://odetocode.com/Blogs/scott/archive/2009/04/27/6-tips-for-asp-net-mvc-model-binding.aspx)自动将映射地址栏中的查询字符串中命名的参数到你的方法中的参数。

在上面的示例中，URL 的Parameters字段未使用，name和numTimes参数作为进行传递[的查询字符串](http://en.wikipedia.org/wiki/Query_string)。?（问号） 在上述 URL 是一个分隔符，并且查询字符串在这之后。& 字符用于分隔查询字符串。

将Welcome方法替换为以下代码：

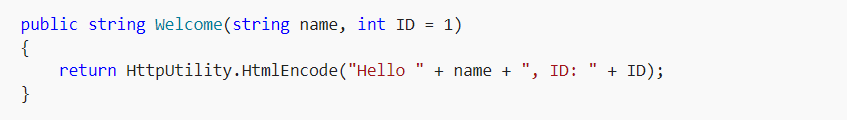


图 4 示例代码

运行应用程序并输入以下<URL:http://localhost:xxx/HelloWorld/Welcome/1?name=Scott>，这一次的第三个 URL 段匹配的路由参数ID。Welcome操作方法包含的参数 (ID) 与RegisterRoutes方法中的 URL 规则相匹配。

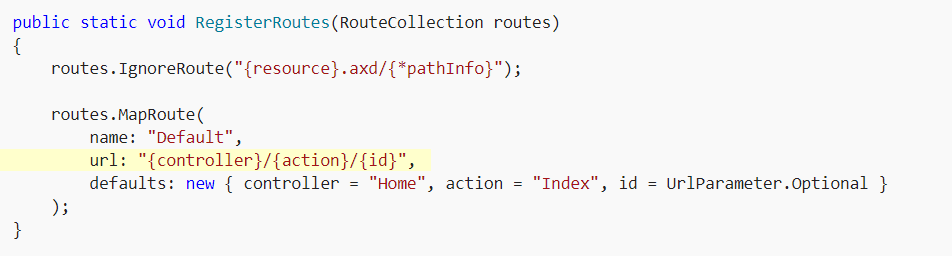


图 5 示例代码

在 ASP.NET MVC 应用程序，更常见的做法是传入作为路由数据 （例如，我们 id 上面所做的那样），而不是将它们作为查询字符串中传递的参数。也可以添加一个路由，在 URL 中的路由数据参数中添加name和numtimes参数。在App\_Start\RouteConfig.cs文件中，添加"Hello"路由：



图 6示例代码

运行应用程序，并输入url:/localhost:XXX/HelloWorld/Welcome/Scott/3。

对于许多 MVC 应用程序，默认路由都能正常工作。在本教程中，随后你将了解使用模型绑定传递数据，而无需修改该默认路由。

在这些示例中，执行操作控制器已经做了"VC"部分 — 也就是说，视图和控制器的工作。控制器直接返回 HTML。通常，你不希望直接返回 HTML，因为这会让代码将变为非常复杂。而是我们通常将使用单独的视图模板文件来帮助生成 HTML 的响应界面。

# 5、添加模型

在本部分中，你将添加用于管理数据的电影类。这些类将位于ASP.NET MVC 应用程序中"Model"中。你将使用.NET Framework 数据访问技术[实体框架](https://docs.microsoft.com/ef/)定义和使用这些模型类。开发范例调用（通常称为 EF）的实体框架支持Code First。Code First可通过编写简单的类来创建模型对象。(这些是也称为 POCO 类，来源于"老式 CLR 对象")，然后，你可以这样非常干净且快速开发工作流，使用类动态创建数据库。如果你需要首先创建数据库，则仍可按照本教程，若要了解有关 MVC 和 EF 应用程序开发。你可以查看Tom Fizmakens [ASP.NET 基架](https://docs.microsoft.com/zh-cn/aspnet/visual-studio/overview/2013/aspnet-scaffolding-overview)教程，里面介绍了数据库第一种连接方法。

在解决方案资源管理器，右键单击模型文件夹，选择添加，然后选择类。

输入类名称"Movie"，然后添加以下五种属性：

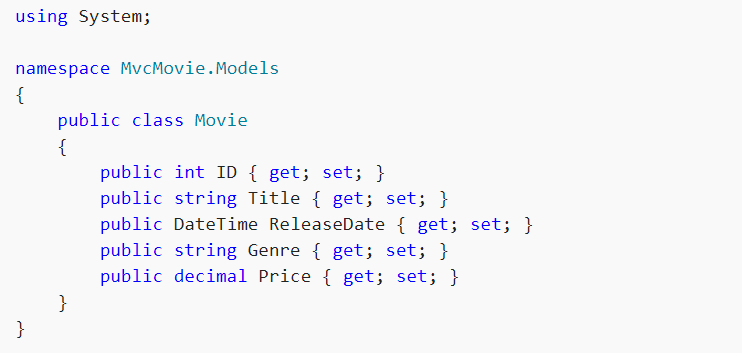


图 7 示例代码

我们将使用Movie类来表示数据库中的电影。 每个实例Movie对象将对应于数据库表中的行，并且每个Movie类中的属性将映射到表中的列。

注意： 若要使用 System.Data.Entity 和相关的类，你需要安装[实体 Framework NuGet 包](https://www.nuget.org/packages/EntityFramework/)。

在同一文件中，添加以下MovieDBContext类：

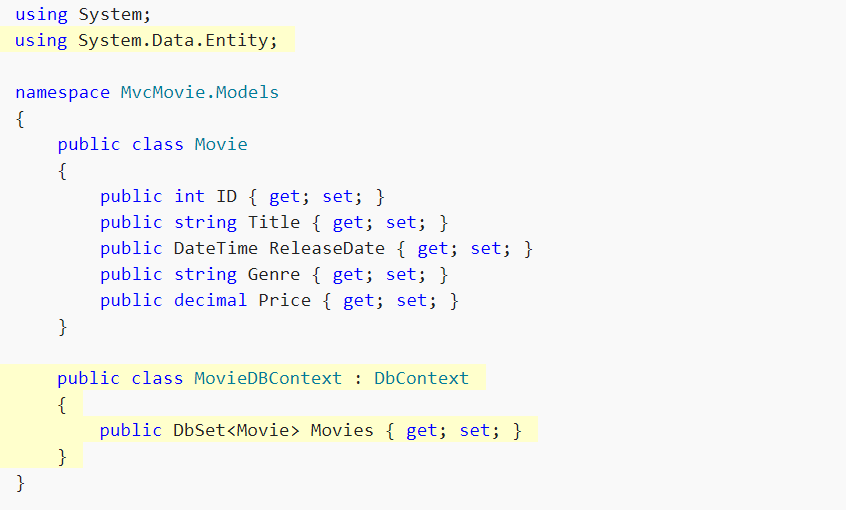


图 8 示例代码

MovieDBContext类表示的实体框架的电影数据库上下文，用于处理提取、 存储和更新数据库中Movie类的实例。 MovieDBContext派生自由实体框架提供的DbContext基类。

若要引用DbContext和DbSet，你需要添加以下using语句文件的顶部：



图 9 示例代码

你可以执行此操作通过手动添加 using 语句，也可以将鼠标悬停在红色波浪线，单击Show potential fixes，然后单击using System.Data.Entity。

注意： 要删除多个未使用using语句。 Visual Studio 将的未使用的依赖关系显示为灰色。 你可以通过将鼠标悬停在灰色的依赖关系中删除 未使用的依赖关系，请单击Show potential fixes单击删除未使用的 Using。

到此，我们添加了模型层(MVC 中的 M) 。

**参 考 文 献 (References)**

1. 陈静,李红灵. 基于B/S模式的管理信息系统开发研究[J]. 云南民族学院学报(自然科学版), 2010.
2. 刘甫迎,刘光会,王蓉.C#程序设计教程（第2版）[M].北京:电子工业出版社,2013.
3. 王慧芳, 毕建权. 软件工程[M]. 浙江: 浙江大学出版社, 2005.
4. 吉根林, 崔海源. Web程序设计(第二版)[M]. 北京: 电子工业出版社, 2006.
5. 陈静,李红灵. 基于B/S模式的管理信息系统开发研究[J]. 云南民族学院学报(自然科学版), 2010.
6. W3Cschool ASP.NET教程. https://www.w3cschool.cn/aspnet/
7. 胡新强. 基于ASP.NET的物流管理系统的设计及车辆调度优化[D].东华大学,2017.
8. 朱帮明. 基于MVC与ASP.NET的学生信息管理系统的设计与实现[D].吉林大学,2015.
9. 吴清芳. 基于ASP.NET的网页设计精品课程网络教学系统的设计与实现[D].中国海洋大学,2011.
10. 宣帅. .NET平台下MVC框架应用研究[D].南京航空航天大学,2010.
11. 董贺. 基于Entity Framework的网络劳资考勤管理系统设计与实现[D].河北师范大学,2016.
12. 刘坤. 基于LINQ的数据查询服务设计及应用研究[D].中国海洋大学,2015.

# Getting Started with ASP.NET MVC 5

# 1. introduction

This tutorial teaches ASP.NET Core MVC with controllers and views. Razor Pages is a new alternative in ASP.NET Core 2.0, a page-based programming model that makes building web UI easier and more productive. We recommend you try the [Razor Pages](https://docs.microsoft.com/aspnet/core/mvc/razor-pages) tutorial before the MVC version. The Razor Pages tutorial:Is easier to follow.Covers more features.Is the preferred approach for new application development.

This tutorial will teach you the basics of building an ASP.NET MVC 5 web app using [Visual Studio 2017](https://www.visualstudio.com/). Final Source for tutorial located on [GitHub](https://github.com/aspnet/Docs/tree/master/aspnet/mvc/overview/getting-started/introduction/sample/MvcMovie/MvcMovie).

You need an Azure account to deploy this app to Azure.

You can [open an Azure account for free](https://azure.microsoft.com/en-us/pricing/free-trial/?WT.mc_id=A443DD604) - You get credits you can use to try out paid Azure services, and even after they're used up you can keep the account and use free Azure services.

You can [activate MSDN subscriber benefits](https://azure.microsoft.com/en-us/pricing/member-offers/msdn-benefits-details/?WT.mc_id=A443DD604) - Your MSDN subscription gives you credits every month that you can use for paid Azure services.

# 2.Getting Started

Start by installing and running [Visual Studio 2017](https://www.visualstudio.com/).

Visual Studio is an IDE, or integrated development environment. Just like you use Microsoft Word to write documents, you'll use an IDE to create applications. In Visual Studio there's a list along the bottom showing various options available to you. There's also a menu that provides another way to perform tasks in the IDE. (For example, instead of selecting New Project from the Start page, you can use the menu and select File > New Project.).

# 3.Creating Your First Application

Click New Project, then select Visual C# on the left, then Web and then select ASP.NET Web Application (.NET Framework). Name your project "MvcMovie" and then click OK.

In the New ASP.NET Project dialog, click MVC and then click OK.

Visual Studio used a default template for the ASP.NET MVC project you just created, so you have a working application right now without doing anything! This is a simple "Hello World!" project, and it's a good place to start your application.

Click F5 to start debugging. F5 causes Visual Studio to start [IIS Express](https://www.iis.net/learn/extensions/introduction-to-iis-express/iis-express-overview) and run your web app. Visual Studio then launches a browser and opens the application's home page. Notice that the address bar of the browser says localhost:port# and not something like example.com. That's because localhost always points to your own local computer, which in this case is running the application you just built. When Visual Studio runs a web project, a random port is used for the web server. In the image below, the port number is 1234. When you run the application, you'll see a different port number.

Right out of the box this default template gives you Home, Contact and About pages. The image above doesn't show the Home, About and Contact links. Depending on the size of your browser window, you might need to click the navigation icon to see these links.

The application also provides support to register and log in. The next step is to change how this application works and learn a little bit about ASP.NET MVC. Close the ASP.NET MVC application and let's change some code.

# 4. Adding a Controller

MVC stands for model-view-controller. MVC is a pattern for developing applications that are well architected, testable and easy to maintain. MVC-based applications contain:Models(Classes that represent the data of the application and that use validation logic to enforce business rules for that data).Views(Template files that your application uses to dynamically generate HTML responses).Controllers(Classes that handle incoming browser requests, retrieve model data, and then specify view templates that return a response to the browser).

We'll be covering all these concepts in this tutorial series and show you how to use them to build an application.

Let's begin by creating a controller class. In Solution Explorer, right-click the Controllers folder and then click Add, then Controller.

In the Add Scaffold dialog box, click MVC 5 Controller - Empty, and then click Add.

Name your new controller "HelloWorldController" and click Add.

Notice in Solution Explorer that a new file has been created named HelloWorldController.cs and a new folder Views\HelloWorld. The controller is open in the IDE.

Replace the contents of the file with the following code.

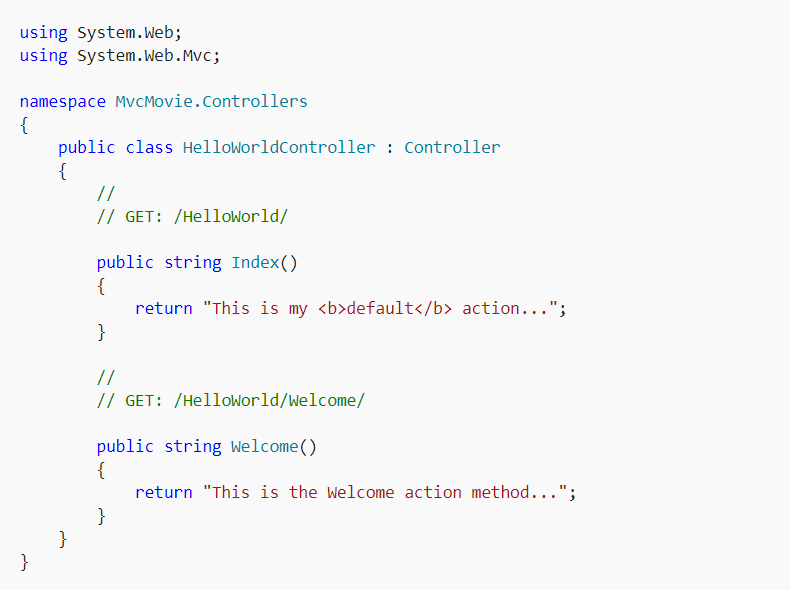


Figure 1

The controller methods will return a string of HTML as an example. The controller is named HelloWorldController and the first method is named Index. Let's invoke it from a browser. Run the application (press F5 or Ctrl+F5). In the browser, append "HelloWorld" to the path in the address bar. (For example, in the illustration below, it's http://localhost:1234/HelloWorld.) The page in the browser will look like the following screenshot. In the method above, the code returned a string directly. You told the system to just return some HTML, and it did!

ASP.NET MVC invokes different controller classes (and different action methods within them) depending on the incoming URL. The default URL routing logic used by ASP.NET MVC uses a format like this to determine what code to invoke:/[Controller]/[ActionName]/[Parameters].

You set the format for routing in the App\_Start/RouteConfig.cs file.

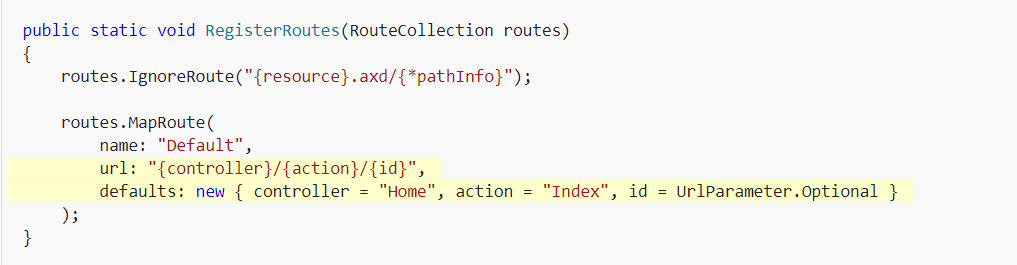


Figure 2

When you run the application and don't supply any URL segments, it defaults to the "Home" controller and the "Index" action method specified in the defaults section of the code above.

The first part of the URL determines the controller class to execute. So /HelloWorld maps to the HelloWorldController class. The second part of the URL determines the action method on the class to execute. So /HelloWorld/Index would cause the Index method of the HelloWorldController class to execute. Notice that we only had to browse to /HelloWorld and the Index method was used by default. This is because a method named Index is the default method that will be called on a controller if one is not explicitly specified. The third part of the URL segment ( Parameters) is for route data. We'll see route data later on in this tutorial.

Browse to http://localhost:xxxx/HelloWorld/Welcome. The Welcome method runs and returns the string "This is the Welcome action method...". The default MVC mapping is /[Controller]/[ActionName]/[Parameters]. For this URL, the controller is HelloWorld and Welcome is the action method. You haven't used the [Parameters] part of the URL yet.

Let's modify the example slightly so that you can pass some parameter information from the URL to the controller (for example, /HelloWorld/Welcome?name=Scott&numtimes=4). Change your Welcomemethod to include two parameters as shown below. Note that the code uses the C# optional-parameter feature to indicate that the numTimes parameter should default to 1 if no value is passed for that parameter.

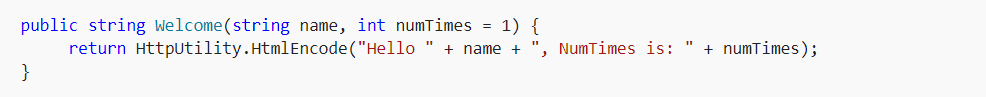


Figure 3

Run your application and browse to the example URL (http://localhost:xxxx/HelloWorld/Welcome?name=Scott&numtimes=4). You can try different values for name and numtimes in the URL. The [ASP.NET MVC model binding system](http://odetocode.com/Blogs/scott/archive/2009/04/27/6-tips-for-asp-net-mvc-model-binding.aspx) automatically maps the named parameters from the query string in the address bar to parameters in your method.

In the sample above, the URL segment ( Parameters) is not used, the name and numTimesparameters are passed as [query strings](http://en.wikipedia.org/wiki/Query_string). The ? (question mark) in the above URL is a separator, and the query strings follow. The & character separates query strings.

Replace the Welcome method with the following code:

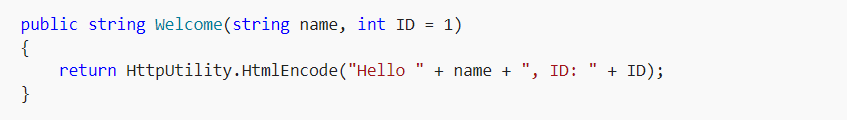


Figure 4

Run the application and enter the following URL: <http://localhost:xxx/HelloWorld/Welcome/1?name=Scott>

This time the third URL segment matched the route parameter ID. The Welcome action method contains a parameter (ID) that matched the URL specification in the RegisterRoutes method.

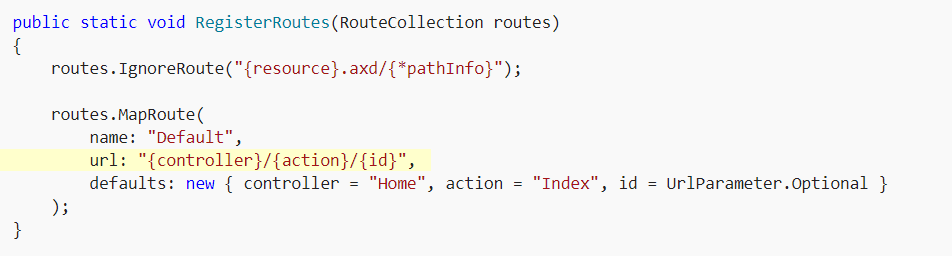


Figure 5

In ASP.NET MVC applications, it's more typical to pass in parameters as route data (like we did with ID above) than passing them as query strings. You could also add a route to pass both the name and numtimes in parameters as route data in the URL. In the App\_Start\RouteConfig.cs file, add the "Hello" route:



Figure 6

Run the application and browse to /localhost:XXX/HelloWorld/Welcome/Scott/3.

For many MVC applications, the default route works fine. You'll learn later in this tutorial to pass data using the model binder, and you won't have to modify the default route for that.

In these examples the controller has been doing the "VC" portion of MVC — that is, the view and controller work. The controller is returning HTML directly. Ordinarily you don't want controllers returning HTML directly, since that becomes very cumbersome to code. Instead we'll typically use a separate view template file to help generate the HTML response. Let's look next at how we can do this.

# 5.Adding a Model

In this section you'll add some classes for managing movies in a database. These classes will be the "model" part of the ASP.NET MVC app.

You'll use a .NET Framework data-access technology known as the [Entity Framework](https://docs.microsoft.com/ef/) to define and work with these model classes. The Entity Framework (often referred to as EF) supports a development paradigm called Code First. Code First allows you to create model objects by writing simple classes. (These are also known as POCO classes, from "plain-old CLR objects.") You can then have the database created on the fly from your classes, which enables a very clean and rapid development workflow. If you are required to create the database first, you can still follow this tutorial to learn about MVC and EF app development. You can then follow Tom Fizmakens [ASP.NET Scaffolding](https://docs.microsoft.com/en-us/aspnet/visual-studio/overview/2013/aspnet-scaffolding-overview) tutorial, which covers the database first approach.

In Solution Explorer, right click the Models folder, select Add, and then select Class.

Enter the class name "Movie".

Add the following five properties to the Movie class:

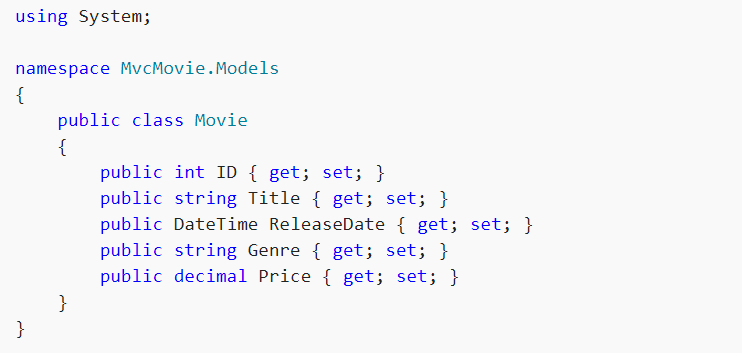


Figure 7

We'll use the Movie class to represent movies in a database. Each instance of a Movie object will correspond to a row within a database table, and each property of the Movie class will map to a column in the table.

Note: In order to use System.Data.Entity, and the related class, you need to install the [Entity Framework NuGet Package](https://www.nuget.org/packages/EntityFramework/). Follow the link for further instructions.

In the same file, add the following MovieDBContext class:

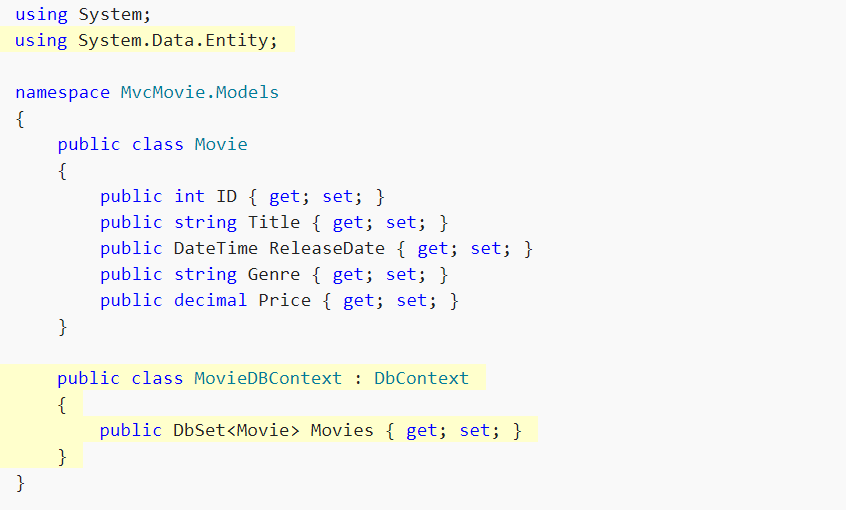


Figure 8

The MovieDBContext class represents the Entity Framework movie database context, which handles fetching, storing, and updating Movie class instances in a database. The MovieDBContext derives from the DbContext base class provided by the Entity Framework.

In order to be able to reference DbContext and DbSet, you need to add the following usingstatement at the top of the file:



Figure 9

You can do this by manually adding the using statement, or you can hover over the red squiggly lines, click Show potential fixes and click using System.Data.Entity;

Note: Several unused using statements have been removed. Visual Studio will show unused dependencies as gray. You can remove unnused dependencies by hovering over the gray dependencies, click Show potential fixes and click Remove Unused Usings.

We've finally added a model (the M in MVC).