بسم اللّه الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

آموزش افزودن یک controller

مدرس : مهندس افشین رفوآ

آموزش افزودن یک controller

MVC سرواژه ی model-view-controller یا به فارسی مدل_ نما_ کنترلگر می باشد. MVC یک الگو ویژه ی طراحی و توسعه ی برنامه هایی است که به بهترین نحو تعبیه شده، قابل آزمایش می باشد و نگهداشت آن نیز بسیار سهل می باشد. در واقع MVC بر روی معماری های چند لایه ای جهت جداسازی قسمت های مختلف برنامه و به طور دقیق تر جدا کردن قسمت های منطقی برنامه از جمله داده ، مجوزها ، اعتبارسنجی داده ها و .... از لایه ی Presentation یا در واقع همان لایه ای که مستقیم با کاربر در ارتباط است ،قرار می گیرد. برنامه های تحت وب مبتنی بر MVC دربردارنده ی:

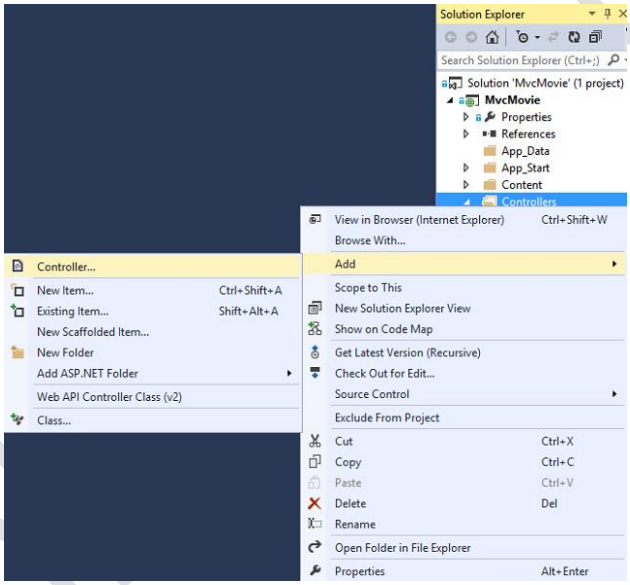
مدل: کلاس هایی که نشانگر داده ها و اطلاعات برنامه بوده و با استفاده از منطق اعتبارسنجی قوانین مربوط به لایه ی business را برای داده های برنامه اعمال می کند. در واقع بار اصلی معماری MVC بر عهده این بخش است . این بخش می تواند با داده ها در ارتباط باشد . منظور از داده لزوما ارتباط با بانک های اطلاعاتی مانند MSSQL ، Access و ... نیست ، حتی منبع داده ها در بخش Model می تواند یک آرایه از اعداد و یا هر چیز دیگری باشد . همان طور که گفته شد Model وظیفه ی بررسی اعتبار و درستی داده ها را هم بر دوش دارد.

نما: فایل های template که برنامه ی شما با استفاده از آن پاسخ های HTML را به صورت پویا ایجاد می کنند. این بخش که در واقع همان بخش Presentation Layer در معماری 3 لایه می باشد وظیفه ی بر قراری ارتباط با کاربر و دریافت داده از کاربر و نمایش داده های آماده با کاربر از طریق برقراری ارتباط با دو بخش دیگر یعنی Model و controller است. در حقیقت این بخش با داده های خام کار می کند.

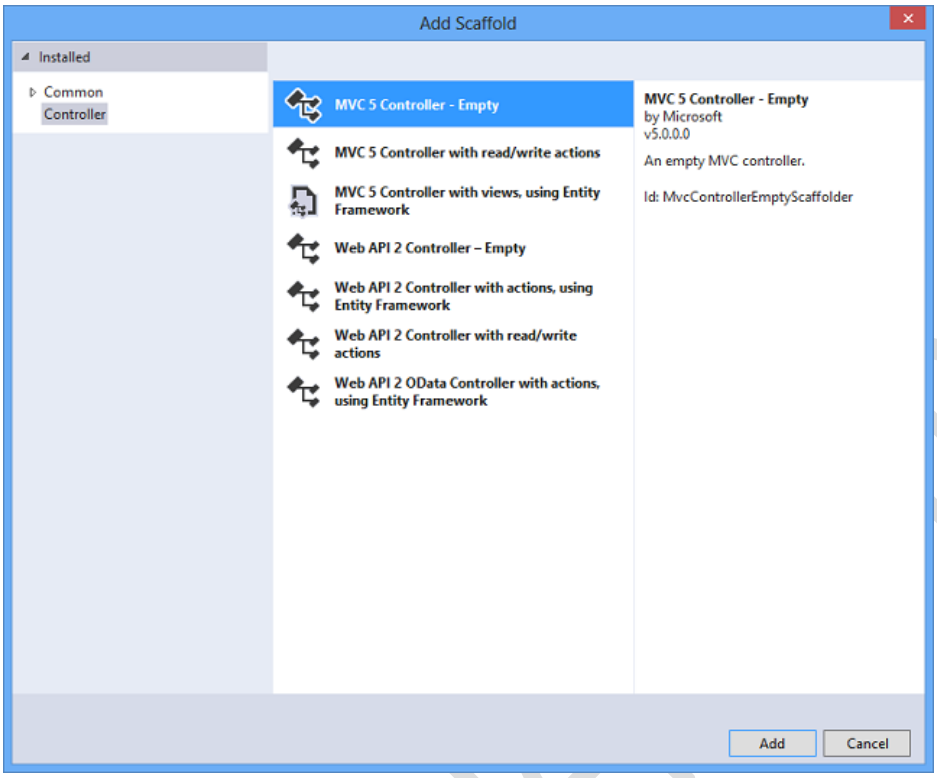
کلاس هایی که درخواست های مرورگر را مدیریت می کند، داده ها را از بخش مدل بازیابی کرده، سپس قالب های view که پاسخی به مرورگر باز می گرداند را مشخص می کند. همان طور که از اسم آن مشخص است controller یک بخش کنترل کننده می باشد و در واقع واسطی بین دو بخش Model و View محسوب می شود.

در مقاله ی آموزشی حاضر تمامی مفاهیم فوق را با جزئیات بیشتر برای شما شرح داده و نحوه ی استفاده از آن ها برای ساخت یک برنامه ی تحت وب را به شما آموزش می دهیم.

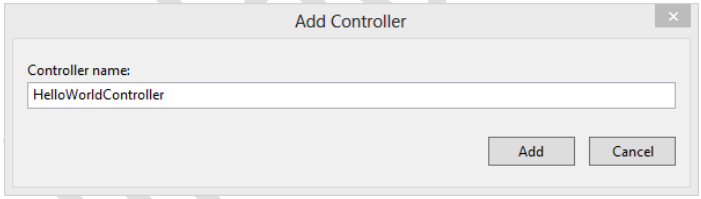
کار خود را با ایجاد یک کلاس controller آغاز می کنیم. در پنجره ی Solution Explorer، روی پوشه ی Controllers راست کلیک کرده، سپس Add و پس از آن Controller را کلیک کنید.



در پنجره ی محاوره ی Add Scaffold، Empty – MVC 5 Controller را کلیک کرده، سپس Add را کلیک نمایید.

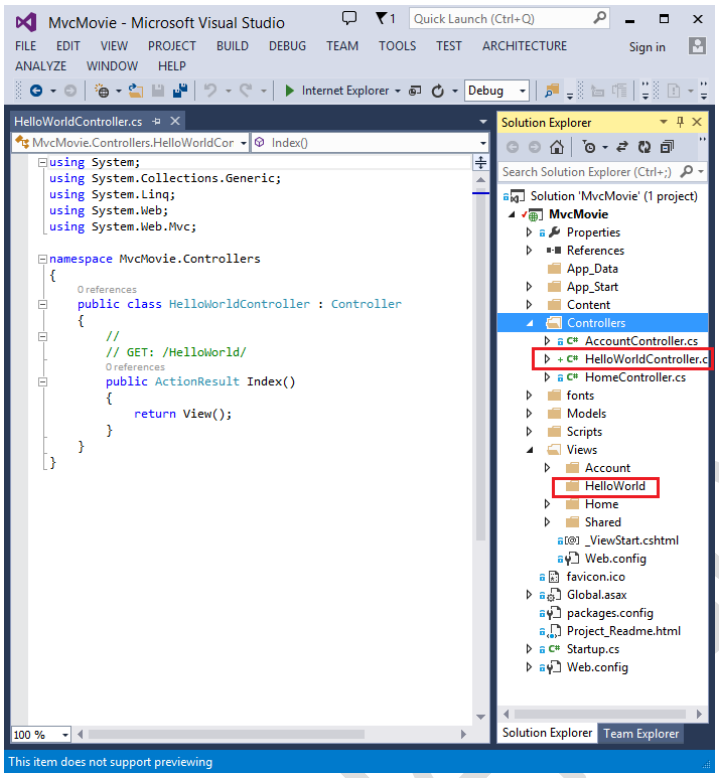


Controller خود را "HelloWorldController" نام گذاری کرده و Add را کلیک کنید:



مشاهده می کنید که در پنجره ی Solution Explorer یک فایل جدید به نام HelloWorldController.cs و یک پوشه ی جدید به نام Views\HelloWorld ایجاد شده است.

controller در محیط برنامه نویسی ویژوال باز می باشد.

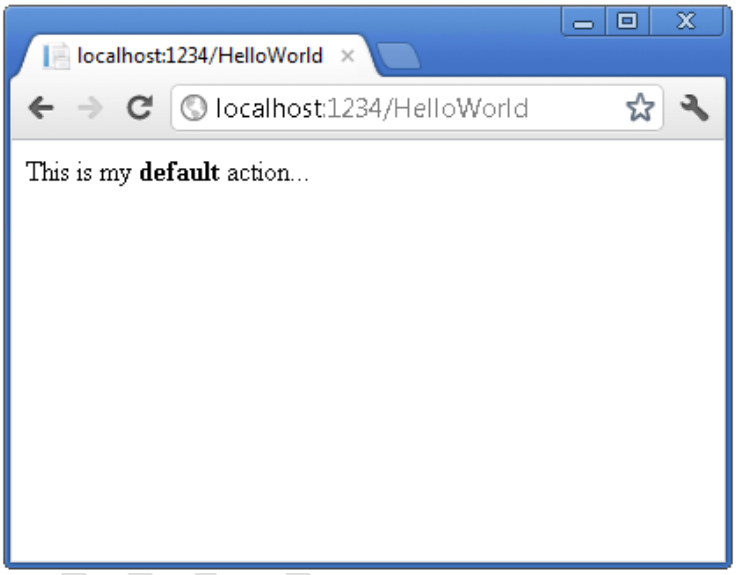


کد زیر را جایگزین محتویات فایل کنید.

```
using System.Web;
using System.Web.Mvc;

namespace MvcMovie.Controllers
{
public class HelloWorldController : Controller
{
//
// GET: /HelloWorld/
public string Index()
{
return "This is my <b>default</b> action...";
}
//
// GET: /HelloWorld/Welcome/
public string Welcome()
{
return "This is the Welcome action method...";
}
}
}
```

متدهای controller یک رشته ی HTML به عنوان نمونه برمی گردانند. Controller مربوطه HelloWorldController و اولین متد نیز Index نام گذاری شده است. حال آن را از یک مرورگر فراخوانی می کنیم. با زدن دکمه ی F5 یا Ctrl+F5 به طور همزمان، برنامه را اجرا کنید. در مرورگر، "HelloWorld" را به path موجود در نوار آدرس ضمیمه کنید. صفحه ی برنامه ی مورد نظر ظاهری شبیه به تصویر زیر خواهد داشت. در متد بالا، کد مورد نظر یک رشته را به طور مستقیم برگرداند. شما به مرورگر دستور دادید مقداری HTML برگرداند و طبق درخواست شما مرورگر این کار را انجام داد.



بسته به URL دریافتی، ASP.NET MVC کلاس های controller متفاوت (و action method های مختلفی درون آن ها) را فرامی خواند. منطق مسیریابی پیش فرض URL که توسط ASP.NET MVC بکار برده می شود، از فرمت زیر برای تشخیص اینکه کدام کد باید فراخوانی شود، بهره می گیرد:

```
/[Controller]/[ActionName]/[Parameters]
```

فرمت مسیریابی را داخل فایل App_Start/RouteConfig.cs تنظیم نمایید.

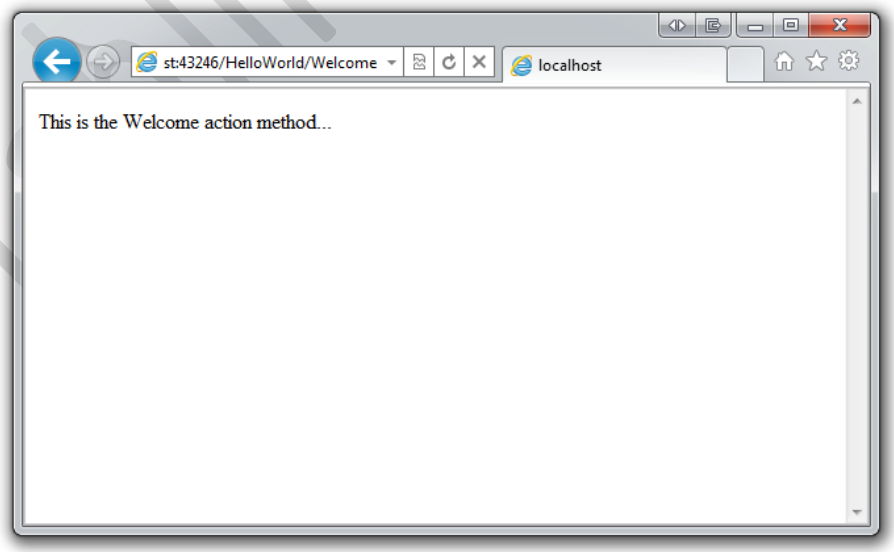
```
public static void RegisterRoutes(RouteCollection routes)
{
routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

routes.MapRoute(
name: "Default",
url: "{controller}/{action}/{id}",
defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
```

```
);
}
```

چنانچه برنامه را اجرا کرده ولی هیچ بخش URL ای برای آن فراهم نکرده باشید، در آن صورت URL به صورت پیش فرض به controller "Home" و متد action "Index" که در بخش defaults کد بالا تعریف کردید تنظیم می شود.

اولین بخش URL کلاس controller مورد نظر را برای اجرا مشخص می کند. بنابراین /HelloWorld به HelloWorldController نگاشت (map) می شود. دوم ین بخش URL، آن متد action ای که باید روی کلاس اجرا شود را تعیین می کند. از این رو /HelloWorld/Index سبب می شود متد Index متعلق به کلاس HelloWorldController اجرا شود. تنها کاری که ما انجام دادیم مراجعه به /HelloWorld بود و متد Index به صورت پیش فرض و بدون دخالت بکار برده شد. دلیلش این است که Index، یک متد پیش فرض است که در صورت تعریف نکردن یک متد (به صورت صریح)، بر روی controller فراخوانی می شود. سومین بخش URL (پارامترها) برای route data درنظر گرفته شده است. به آدرس http://localhost:xxxx/HelloWorld/Welcome مراجعه کنید. متد Welcome اجرا شده و رشته ی "This is the Welcome action method..." را برمی گرداند. Mapping (نگاشت) پیش فرض MVC /[Controller]/[ActionName]/[Parameters] می باشد. این URL، Action method و Welcome controller آن HelloWorld می باشد. هنوز از بخش URL [Parameters] استفاده نشده است.

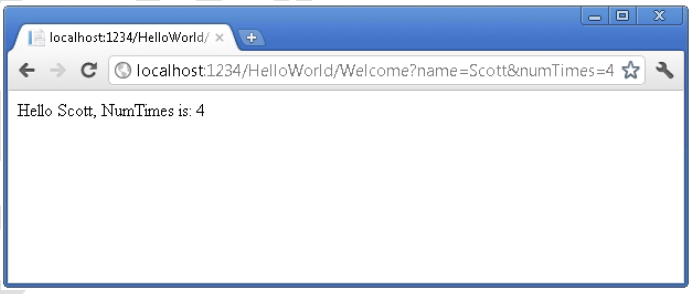


حال مثال فوق را کمی تغییر داده تا بتوان مقداری اطلاعات پارامتر از URL به controller (به عنوان نمونه /HelloWorld/Welcome?name=Scott&numtimes=4) ارسال نمود. با اصلاح متد Welcome دو پارامتر مانند مثال زیر به آن ها اضافه کنید. توجه داشته باشید که کد مورد نظر با استفاده از امکان پارامتر اختیاری (optional-parameter) C# مشخص می کند که پارامتر numTimes در صورت عدم ارسال مقدار برای پارامتر مورد نظر باید به صورت پیش فرض بر روی 1 تنظیم گردد.

```
public string Welcome(string name, int numTimes = 1) {
return HttpUtility.HtmlEncode("Hello " + name + ", NumTimes is: " + numTimes);
}
```

نکته ی امنیتی: کد بالا از HttpServerUtility.HtmlEncode برای محافظت از برنامه درمقابل ورودی مخرب (منظور جاوا اسکریپت) بهره می گیرد.

برنامه را اجرا کرده و به آدرس URL نمونه " http://localhost:xxxx/HelloWorld/Welcome?name=Scott&numtimes=4 " مراجعه کنید. می توانید مقادیر مختلفی را برای name و numtimes در URL خود امتحان کنید. ASP.NET MVC model binding system به صورت خودکار پارامترهای نام گذاری شده را از query string در نوار آدرس به پارامترهای موجود در متد شما نگاشت می کند.

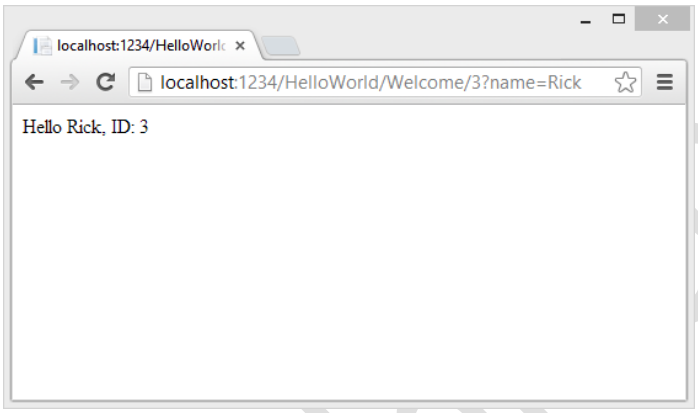


در نمونه ی بالا، بخش (parameters) URL استفاده نشده است. پارامترهای name و numTimes به عنوان query string ارسال شده اند. علامت سوال " ? " که در URL فوق مشاهده می کنید، یک تفکیک کننده (separator) است که query string ها به دنبال آن می آیند. کاراکتر " & " نیز query string ها را از یکدیگر جدا می کند.

متد Welcome را با کد زیر جایگزین کنید:

```
public  string  Welcome (string name, int ID = 1)
{
return HttpUtility.HtmlEncode("Hello " + name + ", ID: " + ID);
}
```

برنامه را اجرا کرده و آدرس http://localhost:xxx/HelloWorld/Welcome/3?name=Rick " URL " را وارد نمایید.



این بار سومین بخش URL با ID پارامتر مسیریابی (route parameter) match شد (منطبق بود).

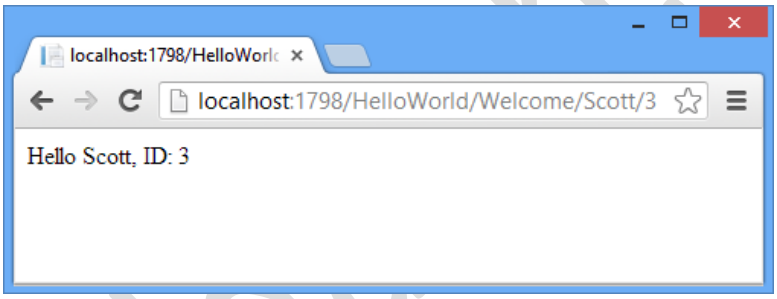
Action method که Welcome نام گذاری شده حاوی یک پارامتر (ID) است که با تعریف URL در متد RegisterRoutes مطابقت دارد (match شد).

```
public static void RegisterRoutes(RouteCollection routes)
{
routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
routes.MapRoute(
name: "Default",
url: "{controller}/{action}/{id}",
defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
);
}
```

در برنامه های تحت وب که توسط ASP.NET MVC طراحی می شوند، رایج است که پارامترها به صورت route data به action method ها ارسال شود (مانند کاری که ما با ID فوق انجام دادیم)، سپس آن ها را به صورت query string پاس دهید. همچنین می توان برای ارسال همزمان پارامترهای name و numtimes به صورت route data در URL، یک route اضافه کنید. کافی است route "Hello" را داخل فایل App_Start\RouteConfig.cs اضافه کنید:

```
public class RouteConfig
{
public static void RegisterRoutes(RouteCollection routes)
{
routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
routes.MapRoute(
name: "Default",
url: "{controller}/{action}/{id}",
defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
);
routes.MapRoute(
name: "Hello",
url: "{controller}/{action}/{name}/{id}"
);
}
}
```

برنامه را اجرا کرده و به آدرس " /localhost:XXX/HelloWorld/Welcome/Scott/3 " مراجعه کنید.



برای اغلب برنامه های MVC مسیر (route) پیش فرض مناسب می باشد. در فواصل آینده به نحوه ی ارسال داده به وسیله ی model binder خواهیم پرداخت که در آن دیگر نیازی به دستکاری مسیر (route) پیش فرض نیست. مثال هایی که در فصل بکار برده و تشریح شدند، وظایف مربوط به بخش های view و controller را انجام می دادند. Controller به طور مستقیم HTML را بازمی گرداند. به طور معمول رایج نیست که controller مسقیم HTML را بازگردانی نماید، به این خاطر که کار کد را بسیار سنگین می کند. بجای آن یک فایل قالب view مجزا ایجاد می کنیم که به ساخت پاسخ های HTML کمک می کند.