آموزش MVC (قسمت دوم)

در بخش اول نحوه ایجاد یک پروژه MVC ، ساختار آن و نحوه اضافه کردن یک اکشن به یک کنترلر توضیح داده شد. و اما ادامه کار :

-چرخه کار : MVC

به طور کلی فرآیند درخواست و پاسخ بین کلاینت و سرور در MVC به صورت زیر می باشد :

۱. یک درخواست از طرف کاربر (کلاینت) به سرو ارسال می شود (مثل مشاهده یک صفحه یا یک رویداد خاص مثل Button_click)
۲. بخشی از MVC که به آن موتور مسیر یاب (Routing Engine) می گویند، کنترلر متناسب با آن درخواست را پیدا می کند .
۳. اکشن متناسب با درخواست اجرا (Invoke) می شود و در صورت لزوم داده ها را از Model مناسب می خواند .
٤. View متناسب با اکشن ، به همراه داده های خوانده شده از Model به سمت کاربر ارسال می شود .

پس به زبان ساده تر :

اگر قصد دارید در برنامه خود رویدادی مثل Button_Click را فراخوانی کنید ، باید اول در کنترلر خود ، اکشنی را برای آن ایجاد کنید که این اکشن ممکن است داده ها را از کلاس ها یا از Model مربوطه بخواند. سپس برای این اکشن یک صفحه (View) مناسب ایجاد کنید. برای هر رویداد دیگری باید همین مراحل را تکرار کنید، یعنی صفحه ای که مثلا ٥ دکمه دارد، باید یک کنترلر که درون آن ٥ اکشن است و ٥ view متناظر با آن داشته باشد و این به شما کمک می کند تا صفحات اختصاصی ، کاراتر و خلوت تری داشته باشید.

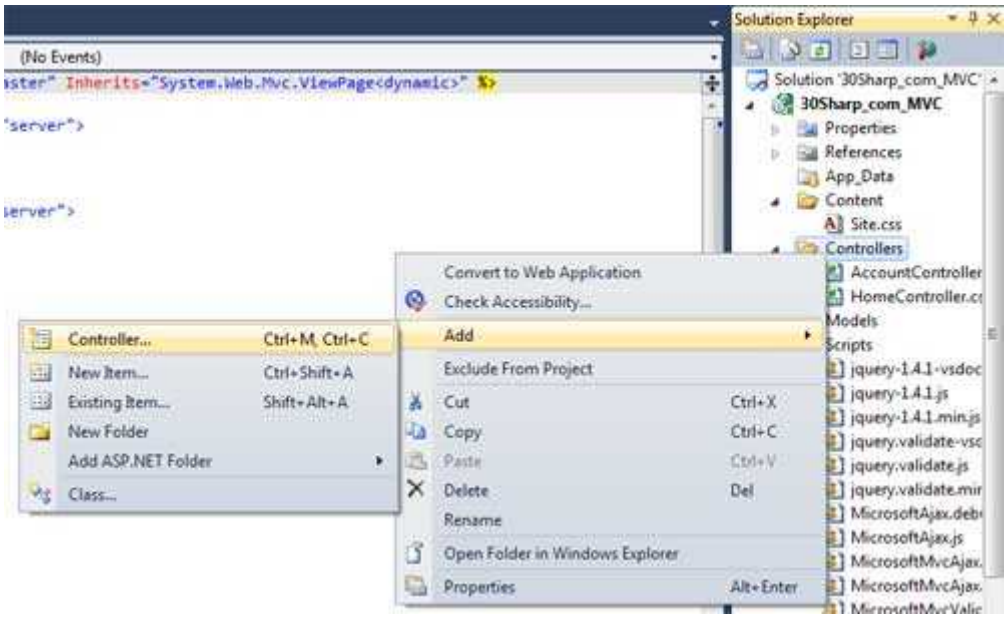
برای اینکه قالبی که Routing Engine از آن استفاده می کند را ببینید ، فایل Global.asax.cs را از درون Solution Explorer خود باز کنید و به بخش RegisterRoutes آن توجه نمایید. قالب URL و اینکه کدام پارامترها اختیاری (Optional) هستند (در اینجا id مشخص شده است) :

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

```
    routes.MapRoute(
        "Default", // Route name
        "{controller}/{action}/{id}", // URL with parameters
        new { controller = "Home", action = "Index", id =
UrlParameter.Optional } // Parameter defaults
    );

}
```

اما مثالهای عملی :

مثال اول: در این مثال ، چند اکشن مختلف با پارامتر و بدون پارامتر آورده شده تا تاثیر آن را در URL صفحه ببینید. در این مثال چون خروجی اکشن ها از نوع string است نیازی به ساختن View نیست . برای اینکار با کلیک راست روی فولدر Controller یک کنترلر جدید به نام SecondController ایجاد کنید :



سپس در این کنترلر ، سه اکشن زیر را ایجاد کنید :

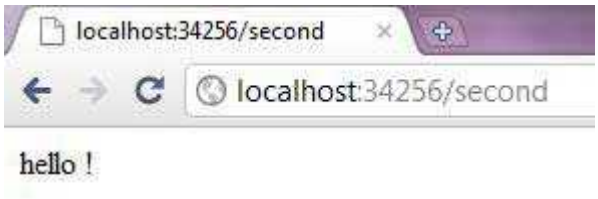
```
using System;

using System.Collections.Generic;
```

```
using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace _30Sharp_com_MVC.Controllers

{

    public class SecondController : Controller

    {

        //          // GET: /Second/

        public string Index()

        {

            string message = "hello !";

            return Server.HtmlEncode(message);

        }

        public string testAction(string id)

        {

            string message = "Hello " + id;

            return Server.HtmlEncode(message);

        }

        public string SayHello(string name)

        {

            string message = "Say hello to: " + name + " " +

Server.HtmlEncode(Request.QueryString["family"]);

            return Server.HtmlEncode(message);

        }

    }

}
```

در هنگام نوشتن اکشن ها به خروجی اکشن و پارامترهای آن توجه کنید. در اکشن Index فقط یک متن ساده نوشته شده است. در اکشن دوم، متغیر id از ورودی خوانده می شود و به همراه پیام نمایش داده می شود . در اکشن سوم علاوه بر متغیر nameمتغیر دیگری از طریق QueryString خوانده می شود.

در اینجا چون خروجی یک رشته ساده است ، Viewبرای آن تعریف نمی کنیم و خروجی آن را از طریق خط آدرس URL مشاهده می کنیم (به ترتیب خروجی اکشن اول تا سوم(

)فالب آدرس را که فراموش نکرده اید (**ControllerName/ActionName/Parameter** :

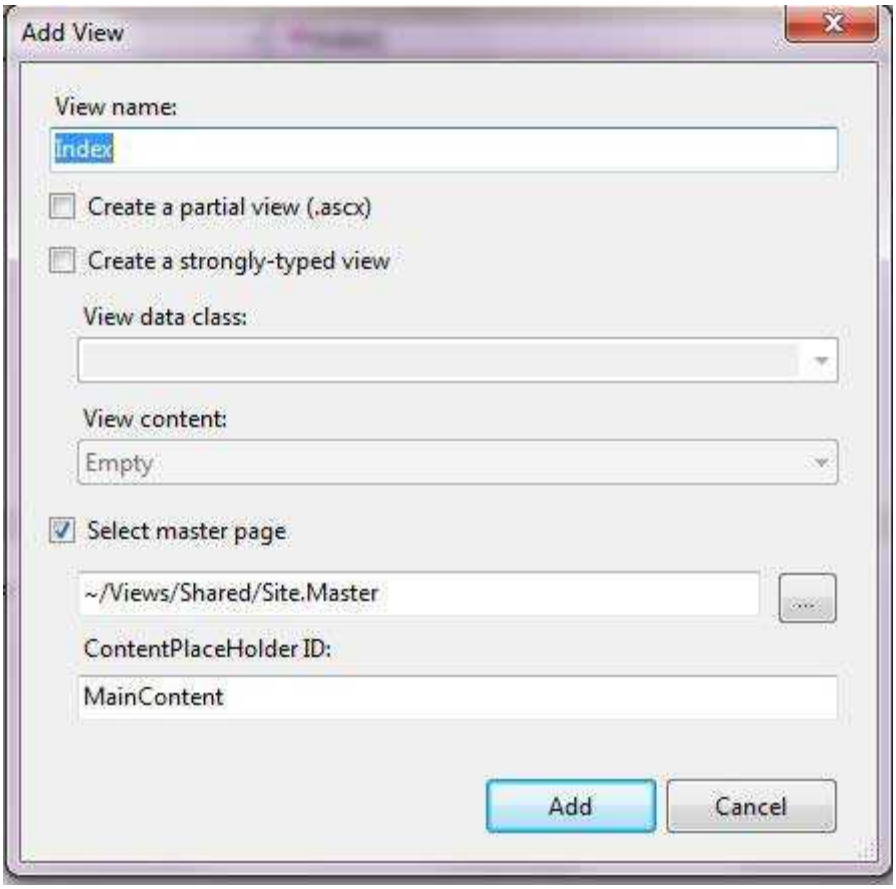


نکته : در این مورد چون نام اکشن Index است (پیش فرض) ، نیازی به تایپ آن نیست

مثال ۲ : در این مثال چگونگی انتقال یک list از نوع string را به) View خروجی (خواهیم دید. برای این کار ابتدا مانند مثال قبل یک Controller جدید به نام ThirdCintroller ایجاد کنید و درون آن لیستی از نوع string ایجاد کنید و مقدار دهی کنید :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
namespace _30Sharp_com_MVC.Controllers
{
    public class ThirdController : Controller
    {
        //          // GET: /Third/

        public ActionResult Index()
        {
            List<string> StudentName = new List<string> { "Houman", "Amir",
"Neda" };
            ViewData["SName"] = StudentName.ToList();
            return View();
        }
    }
}
```

سپس برای آن یک View جدید ایجاد کنید ، یعنی روی نام اکشن (Index) کلیک راست کنید و Add view را انتخاب و گزینه های پیش فرض را بپذیرید :



سپس در صفحه ایجاد شده ، کد زیر را وارد کنید :

```
<asp:content id="Content2" contentplaceholderid="MainContent" runat="server">

        <h2>Index</h2>

            <% foreach (var s in (IEnumerable)ViewData["Sname"]) { %>

                <%: s%>

                <br />

            <% } %>

</asp:content>
```

همانگونه که مشاهده می کنید ، در کد فوق ، یک حلقه foreach وجود دارد که لیست ذخیره شده در متغیر Sname را می خواند ، آن را به نوع IEnumerable تبدیل می کند و چاپش می کند .

توضیح اینکه علامت **<%:** را وقتی بکار می بریم که بخواهیم مقداری را چاپ کنیم (در اینجا مقدار . (s خروجی بصورت زیر خواهد بود :

-3خب برای تمرین مطالب بیان شده ، مثال بعدی را خودتان بنویسید . سعی کنید در یک کنترلر جدید ، با مقدار دادن به یک متغیر از نوع list در هنگام چاپ ، طول رشته را هم چاپ کنید .

در این جلسه با کنترلر ها بیشتر آشنا شدیم و نحوه پیمایش یک لیست را در خروجی دیدیم. در قسمت بعد ، با Model آشنا می شویم و نحوه انتقال مقادیر را از بانک اطلاعاتی به خروجی خواهیم دید.