عنوان: **ASP.NET MVC #8**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۱/۱۳ ۲۲:۱۰:۰۰
آدرس: www.dotnettips.info
گروه‌ها: MVC

معرفی HTML Helpers

یک HTML Helper تنها یک متد است که رشته‌ای را بر می‌گرداند و این رشته می‌تواند حاوی هر نوع محتوای دلخواهی باشد. برای مثال می‌توان از HTML Helpers برای رندر تگ‌های HTML، مانند img و input استفاده کرد. یا به کمک HTML Helpers می‌توان ساختارهای پیچیده‌تری مانند نمایش لیستی از اطلاعات دریافت شده از بانک اطلاعاتی را پیاده سازی کرد. به این ترتیب حجم کدهای تکراری تولید رابط کاربری در Viewهای برنامه‌های ASP.NET MVC به شدت کاهش خواهد یافت، به همراه قابلیت استفاده مجدد از متدهای الحاقی HTML Helpers در برنامه‌های دیگر.
HTML Helpers در ASP.NET MVC معادل کنترل‌های ASP.NET Web forms هستند اما نسبت به آن‌ها بسیار سبک‌تر می‌باشند؛ برای مثال به همراه ViewState و همچنین Event model نیستند.
ASP.NET MVC به همراه تعدادی متد HTML Helper توکار است و برای دسترسی به آن‌ها شیء Html که وهله‌ای از کلاس توکار HtmlHelper می‌باشد، در تمام Viewها قابل استفاده است.

نحوه ایجاد یک HTML Helper سفارشی

از دات نت سه و نیم به بعد امکان توسعه اشیاء توکار فریم ورک، به کمک متدهای الحاقی (extension methods) میسر شده است. برای نوشتن یک HTML Helper نیز باید همین شیوه عمل کرد و کلاس HtmlHelper را توسعه داد. در ادامه قصد داریم یک HTML Helper را جهت رندر تگ label در صفحه ایجاد کنیم. برای این منظور پوشه‌ی جدیدی به نام Helper را به پروژه اضافه نمائید (جهت نظم بیشتر). سپس کلاس زیر را به آن اضافه کنید:

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

namespace MvcApplication4.Helpers
{
    public static class LabelExtensions
    {
        public static string MyLabel(this HtmlHelper helper, string target, string text)
        {
            return string.Format("<label for='{0}'>{1}</label>", target, text);
        }
    }
}
```

همانطور که ملاحظه می‌کنید متد Label به شکل یک متد الحاقی توسعه دهنده کلاس HtmlHelper که تنها یک رشته را بر می‌گرداند، تعریف شده است. اکنون برای استفاده از این متد در View دلخواهی خواهیم داشت:

```
@using MvcApplication4.Helpers
@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

@Html.MyLabel("firstName", "First Name:")
```

ابتدا فضای نام مرتبط با متد الحاقی باید پیوست شود و سپس از طریق شیء Html می‌توان به این متد الحاقی دسترسی پیدا کرد. اگر برنامه را اجرا کنید، این خروجی را مشاهده خواهیم کرد. چرا؟

```
Index
<label for='firstName'>First Name:</label>
```

علت این است که Razor، اطلاعات را Html encoded به مرورگر تحویل می‌دهد. برای تغییر این رویه باید اندکی متد الحاقی تعریف شده را تغییر داد:

```
using System.Web.Mvc;

namespace MvcApplication4.Helpers
{
    public static class LabelExtensions
    {
        public static MvcHtmlString MyLabel(this HtmlHelper helper, string target, string text)
        {
            return MvcHtmlString.Create(string.Format("<label for='{0}'>{1}</label>", target, text));
        }
    }
}
```

تنها تغییر صورت گرفته، استفاده از MvcHtmlString بجای string معمولی است تا Razor آن‌را encode نکند.

تعریف HTML Helpers سفارشی به صورت عمومی:

می‌توان فضای نام MvcApplication4.Helpers این مثال را عمومی کرد. یعنی بجای اینکه بخواهیم در هر View آن‌را ابتدا تعریف کنیم، یکبار آن‌را همانند تعاریف اصلی یک برنامه ASP.NET MVC، عمومی معرفی می‌کنیم. برای این منظور فایل web.config موجود در پوشه Views را باز کنید (و نه فایل web.config قرار گرفته در ریشه اصلی برنامه). سپس فضای نام مورد نظر را در قسمت namespaces صفحات اضافه نمائید:

```
<pages pageBaseType="System.Web.Mvc.WebViewPage">
      <namespaces>
        <add namespace="System.Web.Mvc" />
        <add namespace="System.Web.Mvc.Ajax" />
        <add namespace="System.Web.Mvc.Html" />
        <add namespace="System.Web.Routing" />
        <add namespace="MvcApplication4.Helpers"/>
      </namespaces>
```

به این ترتیب متدهای الحاقی تعریف شده در فضای نام MvcApplication4.Helpers، در تمام Viewهای برنامه در دسترس خواهند بود.

استفاده از کلاس TagBuilder برای تولید HTML Helpers سفارشی:

```
using System.Web.Mvc;

namespace MvcApplication4.Helpers
{
```

```
    public static class LabelExtensions
    {
        public static MvcHtmlString MyNewLabel(this HtmlHelper helper, string target, string text)
        {
            var labelTag = new TagBuilder("label");
            labelTag.MergeAttribute("for", target);
            labelTag.InnerHtml = text;
            return MvcHtmlString.Create(labelTag.ToString());
        }
    }
}
```

در فضای نام System.Web.Mvc، کلاسی وجود دارد به نام TagBuilder که کار تولید تگ‌های HTML، مقدار دهی ویژگی‌ها و خواص آن‌ها را بسیار ساده می‌کند و روش توصیه شده‌ای است برای تولید متدهای HTML Helper. یک نمونه از کاربرد آن‌را برای بازنویسی متد MyLabel ذکر شده در اینجا ملاحظه می‌کنید.
شبیه به همین کلاس، کلاس دیگری به نام HtmlTextWriter در فضای نام System.Web.UI برای انجام اینگونه کارها وجود دارد.

نوشتن HTML Helpers ویژه، به کمک امکانات Razor

نوع دیگری از این متدهای کمکی، Declarative HTML Helpers نام دارند. از این جهت هم Declarative نامیده شده‌اند که مستقیما درون فایل‌های cshtml یا vbhtml به کمک امکانات Razor قابل تعریف هستند. تولید این نوع متدهای کمکی به این شکل نسبت به مثلا روش TagBuilder ساده‌تر است، چون توسط Razor به سادگی و به نحو طبیعی‌تری می‌توان تگ‌های HTML و کدهای مورد نظر را با هم ترکیب کرد (این رفتار طبیعی و روان، یکی از اهداف Razor است).
به عنوان مثال، تعاریف همان کلاس‌های Product و Products قسمت قبل (قسمت هفتم) را در نظر بگیرید. با همان کنترلر و Viewایی که ذکر شد.
سپس برای تعریف این نوع خاص از HTML Helpers/Razor Helpers باید به این نحو عمل کرد:
الف) در ریشه پروژه یا سایت، پوشه‌ی جدیدی به نام App_Code ایجاد کنید (دقیقا به همین نام. این پوشه، جزو پوشه‌های ویژه ASP.NET است).
ب) بر روی این پوشه کلیک راست کرده و گزینه Add|New Item را انتخاب کنید.
ج) در صفحه باز شده، MVC 3 Partial Page/Razor را یافته و مثلا نام ProductsList.cshtml را وارد کرده و این فایل را اضافه کنید.
د) محتوای این فایل جدید را به نحو زیر تغییر دهید:

```
@using MvcApplication4.Models
@helper GetProductsList(List<Product> products)
{
    <ul>
        @foreach (var item in products)
        {
            <li>@item.Name ($@item.Price)</li>
        }
    </ul>
}
```

در اینجا نحوه تعریف یک helper method مخصوص Razor را مشاهده می‌کنید که با کلمه @helper شروع شده است. مابقی آن هم ترکیب آشنای code و markup هستند که به کمک امکانات Razor به این شکل روان میسر شده است.
اکنون اگر Viewایی بخواهد از این اطلاعات استفاده کند تنها کافی است به نحو زیر عمل نماید:

```
@model List<MvcApplication4.Models.Product>
@{
    ViewBag.Title = "Index";
}
```

```
<h2>Index</h2>
@ProductsList.GetProductsList(@Model)
```

ابتدا نام فایل ذکر شده بعد نام متد کمکی تعریف شده در آن. Model هم در اینجا به لیستی از محصولات اشاره می‌کند.
همچنین چون در پوشه app_code قرار گرفته، تمام Viewها به اطلاعات آن دسترسی خواهند داشت. علت هم این است که ASP.NET به صورت خودکار محتوای این پوشه ویژه را همواره کامپایل می‌کند و در اختیار برنامه قرار می‌دهد.
به علاوه در این فایل ProductsList.cshtml، باز هم می‌توان متدهای helper دیگری را اضافه کرد و از این بابت محدودیتی ندارد.
همچنین می‌توان این متد helper را مستقیما داخل یک View هم تعریف کرد. بدیهی است در این حالت قابلیت استفاده مجدد از آن‌را به همراه داشتن Viewهایی تمیز و کم حجم، از دست خواهیم داد.

جهت تکمیل بحث

Turn your Razor helpers into reusable libraries

نظرات خوانندگان

نویسنده: ilia akbarifard
تاریخ: ۱۳۹۱/۰۱/۱۳ ۲۲:۴۵:۴۷

بسیار متشکر آقای نصیری ، به شدت این سلسله مباحث را دنبال می کنم. خداقوت.

نویسنده: محمدی
تاریخ: ۱۳۹۱/۰۱/۱۵ ۰۱:۱۸:۰۷

مرسی خیلی عالیه لطفا ادامه بدید...

نویسنده: بهنام یوسفی
تاریخ: ۱۳۹۱/۰۱/۱۸ ۱۴:۰۲:۵۵

استاد وحید خان نصیری
بسیار ممنون از مطالب عالیتون

نویسنده: TazhKar
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۴:۷

با سلام و تشکر فروان نسبت به اطلاعات کاملی که در اختیار ما قرار میدهد.
میشه لطفی کنید توضیحاتی در مورد MvcHtmlString و در مورد MvcHtmlString.cteate بدهید. قبلا از بابت پاسخ تشکر میکنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۶:۱۶

در قسمت پنجم توضیح دادم.

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۶/۲۴ ۱۳:۱۱

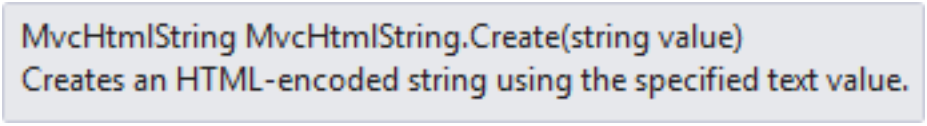
سلام
در متد الحاقی "MyNewLabel" شما مقدار text رو توسط خاصیت InnerHtml به تگ نسبت داید که در این حالت متن Encode نمیشه . می‌خواستم بدونم آیا بهتر نیست از متد SetInnerText در این حالت استفاده کنیم ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۴ ۱۳:۲۸

بسته به نیاز باید تصمیم گیری کرد. یک جایی Html.Raw مناسب است، جای دیگری نمونه encoded آن.

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۶/۲۴ ۱۴:۳۶

سلام
در توضیحات متد "MvcHtmlString.Create" این نوشته شده است :



مگه این به معنی ایجاد یک رشته‌ی encoded نیست؟
علامت "-" قبل از encoded به چه معنی است؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۴ ۱۷:۳۳

ناکافی توضیح داده شده. MvcHtmlString کارش «محصور سازی» یک رشته معمولی به صورت وهله‌ای از کلاس HtmlString است. چون محصور سازی شده، MSDN نوشته Html-encoded که باید می‌نوشت «Wraps the string in a class that implements IHtmlString»
ارائه یک رشته به صورت وهله‌ای از IHtmlString سبب خواهد شد تا زمانیکه از

```
<%: %>
```

و یا @ استفاده می‌شود، از حالت HTML encoded محافظت شود.
این سورس Html.Raw است:

```
public IHtmlString Raw(string value)
{
    return new HtmlString(value);
}

public IHtmlString Raw(object value)
{
    return new HtmlString(value == null ? null : value.ToString());
}
```

و این هم سورس MvcHtmlString :

```
namespace System.Web.Mvc
{
    public sealed class MvcHtmlString : HtmlString
    {
        [SuppressMessage("Microsoft.Security", "CA2104:DoNotDeclareReadOnlyMutableReferenceTypes",
Justification = "MvcHtmlString is immutable")]
        public static readonly MvcHtmlString Empty = Create(String.Empty);

        private readonly string _value;

        public MvcHtmlString(string value)
            : base(value ?? String.Empty)
        {
            _value = value ?? String.Empty;
        }

        public static MvcHtmlString Create(string value)
        {
            return new MvcHtmlString(value);
        }

        public static bool IsNullOrEmpty(MvcHtmlString value)
        {
            return (value == null || value._value.Length == 0);
        }
    }
}
```

در اینجا داریم HtmlString : MvcHtmlString یعنی MvcHtmlString از جنس همان HtmlString ایی است که متدهای MvcHtmlString.Create و Html.Raw باز می‌گردانند.

کاربرد MvcHtmlString عموما در بازگرداندن مقادیر از helper methods است؛ زمانیکه می‌دانید خروجی آن دارای تگ‌های html است. مثلا قرار است یک دکمه را نمایش دهد. به این ترتیب خروجی متد از encode شدن خودکار محافظت می‌شود.
در MVC، سیستم خروجی، رشته‌هایی از جنس IHtmlString را encode نمی‌کند. این بیشتر یک قرار داد است در اینجا.

نویسنده: جواد
تاریخ: ۱۳۹۱/۱۰/۱۴ ۱۷:۲

در قسمت " **نوشتن HTML Helpers ویژه، به کمک امکانات Razor** " چرا توی view موقع استفاده از helper پارامتر اون رو @Model دادید... توضیح هم دادید که به لیستی از محصولات اشاره می‌کنه پس چرا وقتی من به این شکل مینویسم جواب نمیده: مگه@Model چه تفاوتی داره؟

```
@model List<MvcApplication4.Models.Product>
@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

@ProductsList.GetProductsList(List<MvcApplication4.Models.Product>)

یا

@ProductsList.GetProductsList(MvcApplication4.Models.Products)
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۴ ۱۷:۲۳

همان اصول زبان سی شارپ اینجا هم برقرار است. آیا در حالت متداول می‌توانید برای صدا زدن یک متد و ارسال پارامتر به آن بنویسید؟

```
ProductsList.GetProductsList(List<MvcApplication4.Models.Product>)
```

خیر. تعریف فوق در زبان سی شارپ معتبر نیست. برای تعریف یک متد به این شکل عمل می‌شود:

```
public static void GetProductsList(List<Product> list)
{
  // ...
}
```

اما برای صدا زدن این متد استاتیک، اصول سی‌شارپ باید رعایت شود:

```
ClassName.GetProductsList(...instance...)
```

در اینجا وهله یا instance ایی باید به آن پاس شود نه syntax آن و ... @Model یک وهله است.

نویسنده: جواد
تاریخ: ۱۳۹۱/۱۰/۱۴ ۱۸:۱۷

ممنون

فقط یه نکته هنوز برام مشخص نشده که تو کدوم قسمت وهله ای به @Model پاس داده میشه...
تا اونجایی که من متوجه شدم توی فضای نام Models دو تا کلاس Product و Products وجود داره و اینکه @Model وهله ای از

لیست محصولات(که همون Products هست) به حساب میاد، درسته؟ پس فکر کنم مشکل من بر میگرده به اینکه با ساختار List<T>class آشنایی ندارم ، به این خاطره یا چیز دیگس؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۴ ۱۸:۲۴

سورس کامل مثال‌های این سری رو دریافت کنید: MVC_Samples.zip
جایی که وهله‌ای از اشیاء به View متناظر ارسال می‌شود در اکشن متد مشخص شده است (return View):

```
public ActionResult Index()
{
    var products = new Products();
    return View(products);
}
```

نویسنده: جواد
تاریخ: ۱۳۹۱/۱۰/۱۴ ۱۹:۵۲

یعنی @Model یه viewdatadictionary هست... که با این خط کد

```
return view(products)
```

مقادیر پارامتر ورودی در اون قرار میگیره ... درسته؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۴ ۲۰:۵۵

در اینجا Model یک خاصیت از شیء ViewData است که در ControllerBase تعریف شده .
مراجعه کنید به قسمت 5 برای توضیحات بیشتر « بررسی نحوه انتقال اطلاعات از یک کنترلر به Viewهای مرتبط با آن »

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۰۸/۱۱ ۱۰:۵

« تعریف HTML Helpers سفارشی به صورت عمومی:
برای این منظور فایل web.config موجود در پوشه Views را باز کنید (و نه فایل web.config قرار گرفته در ریشه اصلی برنامه).
سپس فضای نام مورد نظر را در قسمت namespaces صفحات اضافه نمائید: »

برای من برعکس بود و باید در فایل web.config در ریشه اصلی برنامه اضافه میشد تا بدون مشکل متد Html Helper اجرا شود
البته من با VB برنامه نویسی میکنم

نویسنده: vici
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱۹:۶

سلام؛ چرا داخل html helper نمی‌تونیم از Actionlink استفاده کنیم؟ هدفم ایجاد لینک هایی ست که به یه متد(Detail) اشاره کنن همراه با Id مورد نظر برای دیدن جزییات. ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱۹:۴۸

از یکی از دو روش زیر استفاده کنید:

```
var urlHelper = new UrlHelper(htmlHelper.ViewContext.RequestContext);
var url = urlHelper.Action("Home", "Index");
```

و یا

```
UrlHelper.GenerateUrl(null, actionName, controllerName,
                                null, null, null, routeValues, htmlHelper.RouteCollection,
                                htmlHelper.ViewContext.RequestContext, true);
```

نویسنده: ع طاهری
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۳:۹

سلام؛ میخوام از کلاس‌های بوت استرپ در input استفاده کنم. آیا میشه برای @Html.TextBoxFor(m => m.P1) از کلاس‌های بوت استرپ استفاده کرد؟

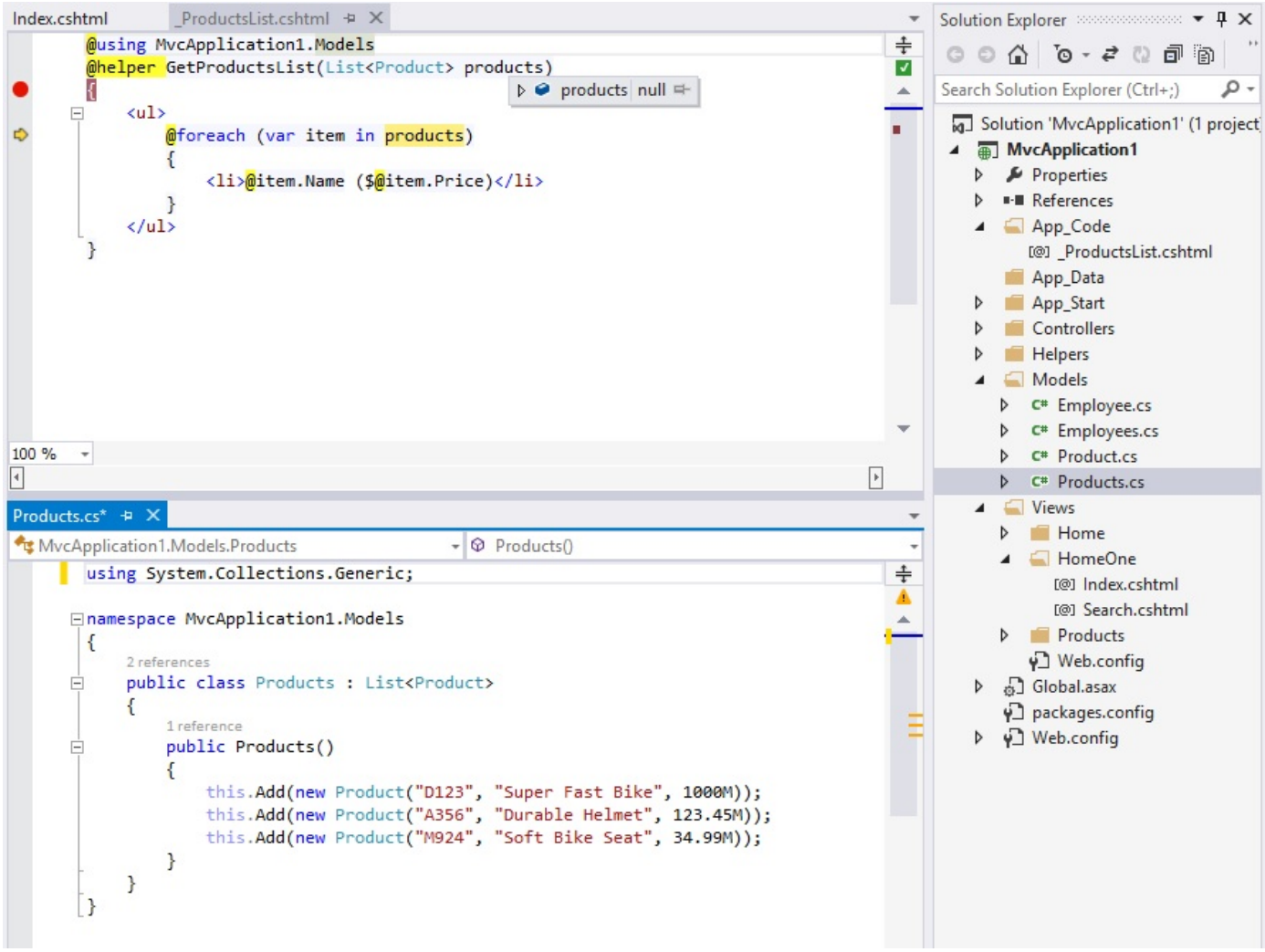
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۳:۳۶

متد TextBoxFor چندین overload دارد برای مشخص کردن ویژگی‌های HTML مانند:

```
@Html.TextBoxFor(m => m.LastName, new { @class = "class-name", placeholder = "last name" })
```

در اینجا چون نام class معادل یکی از نام‌های رزرو سی‌شارپ است، یک @ ابتدای آن قرار گرفته.

نویسنده: پوریا منفرد
تاریخ: ۱۳۹۲/۱۱/۲۲ ۱۵:۳۸



دلیل Null بودن رو متوجه نمی‌شم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۲ ۱۷:۵۵

- فراخوانی این متد کمکی به چه نحوی بوده؟ (اکشن متد آن که @Model را مقدار دهی می‌کند و همچنین کدهای View فراخوان ذکر نشده)
- مثال اصلی بحث جاری از اینجا قابل دریافت است: MVC_Samples.zip
در کدهای پوشه MVC-08 به HomeController و متد Index آن دقت کنید (برای مقدار دهی مدل). همچنین در View فراخوان، در پوشه Views\Home، در فایل Index.cshtml، نحوه‌ی تعریف نوع مدل را توسط @model و استفاده از مقدار آن‌را توسط @Model، بررسی کنید. (نوع مدل با m کوچک شروع می‌شود و مقدار مدل با M بزرگ)

نویسنده: حامد
تاریخ: ۱۳۹۲/۱۲/۱۳ ۱۱:۵۰

میشه لطف کنید بگید اگر helper رو جایی دیگه غیر از App_code داشته باشیم، مثلا توی یک فولدر، چه جوری باید ازش استفاده کرد؟

نویسنده: وحید نصیری

تاریخ: ۱۳۹۲/۱۲/۱۳ ۱۳:۲۳

از razor generator می‌توانید استفاده کنید.