عنوان: Attribute Routing در ASP.NET MVC 5
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۵ ۲:۴۵
آدرس: www.dotnettips.info
گروه‌ها: MVC, ASP.Net MVC, ASP.NET MVC 5

Routing مکانیزم مسیریابی ASP.NET MVC است، که یک URI را به یک اکشن متد نگاشت می‌کند. MVC 5 نوع جدیدی از مسیر یابی را پشتیبانی میکند که Attribute Routing یا مسیریابی نشانه ای نام دارد. همانطور که از نامش پیداست، مسیریابی نشانه ای از Attributeها برای این امر استفاده میکند. این روش به شما کنترل بیشتری روی URIهای اپلیکیشن تان می‌دهد.
مدل قبلی مسیریابی (conventional-routing) هنوز کاملا پشتیبانی می‌شود. در واقع می‌توانید هر دو تکنیک را بعنوان مکمل یکدیگر در یک پروژه استفاده کنید.
در این پست قابلیت‌ها و گزینه‌های اساسی مسیریابی نشانه ای را بررسی میکنیم.
چرا مسیریابی نشانه ای؟
فعال سازی مسیریابی نشانه ای
پارامترهای اختیاری URI و مقادیر پیش فرض
پیشوند مسیر ها
مسیر پیش فرض
محدودیت‌های مسیر ها محدودیت‌های سفارشی

نام مسیر ها
ناحیه‌ها (Areas)

چرا مسیریابی نشانه ای

برای مثال یک وب سایت تجارت آنلاین بهینه شده اجتماعی، می‌تواند مسیرهایی مانند لیست زیر داشته باشد:

{productId:int}/{productTitle}

نگاشت می‌شود به: ProductsController.Show(int id)

{username}

نگاشت می‌شود به: ProfilesController.Show(string username)

{username}/catalogs/{catalogId:int}/{catalogTitle}

نگاشت می‌شود به: CatalogsController.Show(string username, int catalogId)

در نسخه قبلی ASP.NET MVC، قوانین مسیریابی در فایل RouteConfig.cs تعریف می‌شدند، و اشاره به اکشن‌های کنترلرها به نحو زیر انجام می‌شد:

```
routes.MapRoute(
    name: "ProductPage",
    url: "{productId}/{productTitle}",
    defaults: new { controller = "Products", action = "Show" },
    constraints: new { productId = "\\d+" }
);
```

هنگامی که قوانین مسیریابی در کنار اکشن متدها تعریف می‌شوند، یعنی در یک فایل سورس و نه در یک کلاس پیکربندی خارجی، درک و فهم نگاشت URIها به اکشن‌ها واضح‌تر و راحت می‌شود. تعریف مسیر قبلی، می‌تواند توسط یک attribute ساده بدین صورت نگاشت شود:

```
[Route("{productId:int}/{productTitle}")]
public ActionResult Show(int productId) { ... }
```

فعال سازی Attribute Routing

برای فعال سازی مسیریابی نشانه ای، متد MapMvcAttributeRoutes را هنگام پیکربندی فراخوانی کنید.

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

        routes.MapMvcAttributeRoutes();
    }
}
```

همچنین می‌توانید مدل قبلی مسیریابی را با تکنیک جدید تلفیق کنید.

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

    routes.MapMvcAttributeRoutes();

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

پارامترهای اختیاری URI و مقادیر پیش فرض

می توانید با اضافه کردن یک علامت سوال به پارامترهای مسیریابی، آنها را optional یا اختیاری کنید. برای تعیین مقدار پیش فرض هم از فرمت parameter=value استفاده می‌کنید.

```
public class BooksController : Controller
{
    // eg: /books
    // eg: /books/1430210079
    [Route("books/{isbn?}")]
    public ActionResult View(string isbn)
    {
        if (!String.IsNullOrEmpty(isbn))
        {
            return View("OneBook", GetBook(isbn));
        }
        return View("AllBooks", GetBooks());
    }

    // eg: /books/lang
    // eg: /books/lang/en
    // eg: /books/lang/he
    [Route("books/lang/{lang=en}")]
    public ActionResult ViewByLanguage(string lang)
    {
        return View("OneBook", GetBooksByLanguage(lang));
    }
}
```

در این مثال، هر دو مسیر books/ و books/1430210079/ به اکشن متد "View" نگاشت می‌شوند، مسیر اول تمام کتاب‌ها را لیست میکند، و مسیر دوم جزئیات کتابی مشخص را لیست می‌کند. هر دو مسیر books/lang/ و books/lang/en/ به یک شکل نگاشت می‌شوند، چرا که مقدار پیش فرض این پارامتر en تعریف شده.

پیشوند مسیرها (Route Prefixes)

برخی اوقات، تمام مسیرها در یک کنترلر با یک پیشوند شروع می‌شوند. بعنوان مثال:

```
public class ReviewsController : Controller
{
    // eg: /reviews
```

```
    [Route("reviews")]
    public ActionResult Index() { ... }
    // eg: /reviews/5
    [Route("reviews/{reviewId}")]
    public ActionResult Show(int reviewId) { ... }
    // eg: /reviews/5/edit
    [Route("reviews/{reviewId}/edit")]
    public ActionResult Edit(int reviewId) { ... }
}
```

همچنین می‌توانید با استفاده از خاصیت [RoutePrefix] یک پیشوند عمومی برای کل کنترلر تعریف کنید:

```
[RoutePrefix("reviews")]
public class ReviewsController : Controller
{
    // eg.: /reviews
    [Route]
    public ActionResult Index() { ... }
    // eg.: /reviews/5
    [Route("{reviewId}")]
    public ActionResult Show(int reviewId) { ... }
    // eg.: /reviews/5/edit
    [Route("{reviewId}/edit")]
    public ActionResult Edit(int reviewId) { ... }
}
```

در صورت لزوم، می‌توانید برای بازنویسی (override) پیشوند مسیرها از کاراکتر ~ استفاده کنید:

```
[RoutePrefix("reviews")]
public class ReviewsController : Controller
{
    // eg.: /spotlight-review
    [Route("~/spotlight-review")]
    public ActionResult ShowSpotlight() { ... }

    ...
}
```

مسیر پیش فرض

می توانید خاصیت [Route] را روی کنترلر اعمال کنید، تا اکشن متد را بعنوان یک پارامتر بگیرید. این مسیر سپس روی تمام اکشن متدهای این کنترلر اعمال می‌شود، مگر آنکه یک [Route] بخصوص روی اکشن‌ها تعریف شده باشد.

```
[RoutePrefix("promotions")]
[Route("{action=index}")]
public class ReviewsController : Controller
{
    // eg.: /promotions
    public ActionResult Index() { ... }

    // eg.: /promotions/archive
    public ActionResult Archive() { ... }

    // eg.: /promotions/new
    public ActionResult New() { ... }

    // eg.: /promotions/edit/5
    [Route("edit/{promoId:int}")]
    public ActionResult Edit(int promoId) { ... }
}
```

محدودیت‌های مسیر ها

با استفاده از Route Constraints می‌توانید نحوه جفت شدن پارامترها در قالب مسیریابی را محدود و کنترل کنید. فرمت کلی {parameter:constraint} است. بعنوان مثال:

```
// eg: /users/5
[Route("users/{id:int}"]
public ActionResult GetUserById(int id) { ... }

// eg: users/ken
[Route("users/{name}"]
public ActionResult GetUserByName(string name) { ... }
```

در اینجا، مسیر اول تنها در صورتی انتخاب می‌شود که قسمت id در URI یک مقدار integer باشد. در غیر اینصورت مسیر دوم انتخاب خواهد شد.

جدول زیر constraintها یا محدودیت هایی که پشتیبانی می‌شوند را لیست می‌کند.

مثال	توضیحات	محدودیت
{x:alpha}	کاراکترهای الفبای لاتین را تطبیق (match) می‌دهد (a-z, A-Z).	alpha
{x:bool}	یک مقدار منطقی را تطبیق می‌دهد.	bool
{x:datetime}	یک مقدار **DateTime** را تطبیق می‌دهد.	datetime
{x:decimal}	یک مقدار پولی را تطبیق می‌دهد.	decimal
{x:double}	یک مقدار اعشاری 64 بیتی را تطبیق می‌دهد.	double
{x:float}	یک مقدار اعشاری 32 بیتی را تطبیق می‌دهد.	float
{x:guid}	یک مقدار GUID را تطبیق می‌دهد.	guid
{x:int}	یک مقدار 32 بیتی integer را تطبیق می‌دهد.	int
{x:length(6)} {x:length(1,20)}	رشته ای با طول تعیین شده را تطبیق می‌دهد.	length
{x:long}	یک مقدار 64 بیتی integer را تطبیق می‌دهد.	long
{(x:max(10}	یک مقدار integer با حداکثر مجاز را تطبیق می‌دهد.	max
{(x:maxlength(10}	رشته ای با حداکثر طول تعیین شده را تطبیق می‌دهد.	maxlength
{(x:min(10}	مقداری integer با حداقل مقدار تعیین شده را تطبیق می‌دهد.	min
{(x:minlength(10}	رشته ای با حداقل طول تعیین شده را تطبیق می‌دهد.	minlength
{x:range(10,50)}	مقداری integer در بازه تعریف شده را تطبیق می‌دهد.	range
{($\{x:regex(^\d{3}-\d{3}-\d{4}	یک عبارت با قاعده را تطبیق می‌دهد.	regex

توجه کنید که بعضی از constraint ها، مانند "min" آرگومان‌ها را در پرانتز دریافت می‌کنند.
می توانید محدودیت‌های متعددی روی یک پارامتر تعریف کنید، که باید با دونقطه جدا شوند. بعنوان مثال:

```
// eg: /users/5
// but not /users/10000000000 because it is larger than int.MaxValue,
// and not /users/0 because of the min(1) constraint.
[Route("users/{id:int:min(1)}")]
public ActionResult GetUserById(int id) { ... }
```

مشخص کردن اختیاری بودن پارامتر ها، باید در آخر لیست constraints تعریف شود:

```
// eg: /greetings/bye
// and /greetings because of the Optional modifier,
// but not /greetings/see-you-tomorrow because of the maxlength(3) constraint.
[Route("greetings/{message:maxlength(3)?}")]
public ActionResult Greet(string message) { ... }
```

محدودیت‌های سفارشی

با پیاده سازی قرارداد IRouteConstraint می‌توانید محدودیت‌های سفارشی بسازید. بعنوان مثال، constraint زیر یک پارامتر را به لیستی از مقادیر قابل قبول محدود می‌کند:

```
public class ValuesConstraint : IRouteConstraint
{
    private readonly string[] validOptions;
    public ValuesConstraint(string options)
    {
        validOptions = options.Split('|');
    }

    public bool Match(HttpContextBase httpContext, Route route, string parameterName,
RouteValueDictionary values, RouteDirection routeDirection)
    {
        object value;
        if (values.TryGetValue(parameterName, out value) && value != null)
        {
            return validOptions.Contains(value.ToString(), StringComparer.OrdinalIgnoreCase);
        }
        return false;
    }
}
```

قطعه کد زیر نحوه رجیستر کردن این constraint را نشان می‌دهد:

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

        var constraintsResolver = new DefaultInlineConstraintResolver();

        constraintsResolver.ConstraintMap.Add("values", typeof(ValuesConstraint));

        routes.MapMvcAttributeRoutes(constraintsResolver);
    }
}
```

حالا می‌توانید این محدودیت سفارشی را روی مسیرها اعمال کنید:

```
public class TemperatureController : Controller
{
    // eg: temp/celsius and /temp/fahrenheit but not /temp/kelvin
    [Route("temp/{scale:values(celsius|fahrenheit)}")]
    public ActionResult Show(string scale)
    {
```

```
        return Content("scale is " + scale);
    }
}
```

نام مسیر ها

می توانید به مسیرها یک نام اختصاص دهید، با این کار تولید URIها هم راحت‌تر می‌شوند. بعنوان مثال برای مسیر زیر:

```
[Route("menu", Name = "mainmenu")]
public ActionResult MainMenu() { ... }
```

می‌توانید لینکی با استفاده از Url.RouteUrl تولید کنید:

```
<a href="@Url.RouteUrl("mainmenu")">Main menu</a>
```

ناحیه‌ها (Areas)

برای مشخص کردن ناحیه ای که کنترلر به آن تعلق دارد می‌توانید از خاصیت [RouteArea] استفاده کنید. هنگام استفاده از این خاصیت، می‌توانید با خیال راحت کلاس AreaRegistration را از ناحیه مورد نظر حذف کنید.

```
[RouteArea("Admin")]
[RoutePrefix("menu")]
[Route("{action}")]
public class MenuController : Controller
{
    // eg: /admin/menu/login
    public ActionResult Login() { ... }

    // eg: /admin/menu/show-options
    [Route("show-options")]
    public ActionResult Options() { ... }

    // eg: /stats
    [Route("~/stats")]
    public ActionResult Stats() { ... }
}
```

با این کنترلر، فراخوانی تولید لینک زیر، رشته " */Admin/menu/show-options* " را بدست میدهد:

```
Url.Action("Options", "Menu", new { Area = "Admin" })
```

به منظور تعریف یک پیشوند سفارشی برای یک ناحیه، که با نام خود ناحیه مورد نظر متفاوت است می‌توانید از پارامتر AreaPrefix استفاده کنید. بعنوان مثال:

```
[RouteArea("BackOffice", AreaPrefix = "back-office")]
```

اگر از ناحیه‌ها هم بصورت مسیریابی نشانه ای، و هم بصورت متداول (که با کلاس‌های AreaRegistration پیکربندی می‌شوند) استفاده می‌کنید باید مطمئن شوید که رجیستر کردن نواحی اپلیکیشن پس از مسیریابی نشانه ای پیکربندی می‌شود. به هر حال رجیستر کردن ناحیه‌ها پیش از تنظیم مسیرها بصورت متداول باید صورت گیرد. دلیل آن هم مشخص است، برای اینکه درخواست‌های ورودی بدرستی با مسیرهای تعریف شده تطبیق داده شوند، باید ابتدا attribute routes، سپس area registration و در آخر default route رجیستر شوند. بعنوان مثال:

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

```
    routes.MapMvcAttributeRoutes();

    AreaRegistration.RegisterAllAreas();

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

نظرات خوانندگان

نویسنده: منصور جعفری
تاریخ: ۱۳۹۲/۱۲/۲۹ ۰:۲۷

سلام
الان من در قسمت route.config به این صورت کدهام تعریف شده

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

و اومدم توی کنترل Home و اکشن موبوط به اون که دارای یک پارامتر دریافتی برای پیج هست این کد رو تعریف کردم:

```
[Route("Page/{page?}")]
        public ActionResult List(int page=1)
        {
```

اما وقتی برنامه رو اجرا میکنم خطای 404 رو میده. ممنون میشم راهنمایی کنید مشکل از کجاست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۹ ۱:۱۱

برای ریشه سایت ویژگی زیر را هم باید اضافه کنید:

```
[Route("~/")]
```

نویسنده: منصور جعفری
تاریخ: ۱۳۹۲/۱۲/۲۹ ۱۲:۳۳

ممنونم که پاسخ دادید یعنی باید به این شکل بنویسم

```
[Route("~/Page/{page?}")]
        public ActionResult List(int page=1)
        {
```

نیازی به تعریف اکشن یا RoutePrifix نیست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۹ ۱۲:۴۵

قابلیت ترکیب دارند:

```
[Route("~/")]
[Route("Page/{page?}")]
public ActionResult List(int page=1)
{
```

نویسنده: یاشار راشدی
تاریخ: ۱۳۹۳/۰۳/۲۹ ۱۳:۲۲

در صورتی که از Areaها استفاده کنید باید بالای هر کنترلر داخل Area حتما Prefix مربوط به Area را اضافه کنید وگرنه Exception دریافت می‌کنید.

```
[RouteArea("Admin")]
[RoutePrefix("menu")]
[Route("{action}")]
public class MenuController : Controller
{
    // eg: /admin/menu/login
    public ActionResult Login() { ... }

    // eg: /admin/menu/show-options
    [Route("show-options")]
    public ActionResult Options() { ... }

    // eg: /stats
    [Route("~/stats")]
    public ActionResult Stats() { ... }
```

توضیحات بیشتر

http://blogs.msdn.com/b/webdev/archive/2013/10/17/attribute-routing-in-asp-net-mvc-5.aspx#route-areas