史上最全的ASP.NET MVC 路由配置，以后RouteConfig再弄不懂神仙都难救你啦~ - Cherbim

时间 2014-08-15 10:10:00 博客园-原创精华区
原文 http://www.cnblogs.com/zeusro/p/RouteConfig.html
主题 ASP.NET MVC

继续延续坑爹标题系列。其实只是把apress.pro.asp.net.mvc.4.framework里的CHAPTER 13翻译过来罢了，当做自己总结吧。内容看看就好，排版就不要吐槽了，反正我知道你也不会反对的。

XD 首先说URL的构造。 其实这个也谈不上构造，只是语法特性吧。

命名参数规范+匿名对象

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

构造路由然后添加

```
Route myRoute = new Route("{controller}/{action}", new MvcRouteHandler()); routes.Add("MyRoute", myRoute);
```

直接方法重载+匿名对象

```
routes.MapRoute("ShopSchema", "Shop/{action}", new { controller = "Home" });
```

个人觉得第一种比较易懂，第二种方便调试，第三种写起来比较效率吧。各取所需吧。本文行文偏向于第三种。

1.默认路由(MVC自带)

```
routes.MapRoute( "Default", // 路由名称 "{controller}/{action}/{id}", // 带有参数的 URL new { controller = "Home",
```

2.静态URL段

```
routes.MapRoute("ShopSchema2", "Shop/OldAction", new { controller = "Home", action = "Index" });
routes.MapRoute("ShopSchema", "Shop/{action}", new { controller = "Home" });
routes.MapRoute("ShopSchema2", "Shop/OldAction.js", new { controller = "Home", action = "Index" });
```

没有占位符路由就是现成的写死的。

比如这样写然后去访问http://localhost:XXX/Shop/OldAction.js，response也是完全没问题的。 controller ， action ， area这三个保留字就别设静态变量里面了。

3.自定义常规变量URL段（好吧这翻译暴露智商了）

```
routes.MapRoute("MyRoute2", "{controller}/{action}/{id}", new { controller = "Home", action = "Index", id = "Defaul
```

这种情况如果访问 /Home/Index 的话，因为第三段（id）没有值，根据路由规则这个参数会被设为DefaultId

这个用viewbag给title赋值就能很明显看出

```
ViewBag.Title = RouteData.Values["id"];
```

图不贴了，结果是标题显示为DefaultId。 注意要在控制器里面赋值，在视图赋值没法编译的。

4.再述默认路由

然后再回到默认路由。 UrlParameter.Optional这个叫可选URL段.路由里没有这个参数的话id为null。 照原文大致说法，这个可选URL段能用来实现一个关注点的分离。刚才在路由里直接设定参数默认值其实不是很好。照我的理解，实际参数是用户发来的，我们做的只是定义形式参数名。但是，如果硬要给参数赋默认值的话，建议用语法糖写到action参数里面。比如：

```
public ActionResult Index(string id = "abcd") { ViewBag.Title = RouteData.Values["id"]; return View(); }
```

5.可变长度路由。

```
/{id}/{*catchall}", new { controller = "Home", action = "Index", i
```

所以 /Home/Index/dabdafdaf 等效于
oeiho 等效于
/Home/Index/All/Delete/Perm/.....

6. 跨命名空间路由

这个提醒一下记得引用命名空间，开启IIS网站不然就是404。这个非常非主流，不建议瞎搞。


```
routes.MapRoute("MyRoute", "{controller}/{action}/{id}/{*catchall}", new { controller = "Home", action = "Index", i
```

但是这样写的话数组排名不分先后的，如果有多个匹配的路由会报错。 然后作者提出了一种改进写法。


```
routes.MapRoute("AddContollerRoute", "Home/{action}/{id}/{*catchall}", new { controller = "Home", action = "Index",

routes.MapRoute("MyRoute", "{controller}/{action}/{id}/{*catchall}", new { controller = "Home", action = "Index", i
```

这样第一个URL段不是Home的都交给第二个处理 最后还可以设定这个路由找不到的话就不给后面的路由留后路啦，也就不再往下找啦。

```
Route myRoute = routes.MapRoute("AddContollerRoute", "Home/{action}/{id}/{*catchall}", new { controller = "Home", a

myRoute.DataTokens["UseNamespaceFallback"] = false;
```

7. 正则表达式匹配路由

```
routes.MapRoute("MyRoute", "{controller}/{action}/{id}/{*catchall}",
 new { controller = "Home", action = "Index", id = UrlParameter.Optional },
 new { controller = "^H.*"},
new[] { "URLsAndRoutes.Controllers"});
```

约束多个URL

```
routes.MapRoute("MyRoute", "{controller}/{action}/{id}/{*catchall}",
new { controller = "Home", action = "Index", id = UrlParameter.Optional },
new { controller = "^H.*", action = "^Index$|^About$"},
new[] { "URLsAndRoutes.Controllers"});
```

8. 指定请求方法

```
routes.MapRoute("MyRoute", "{controller}/{action}/{id}/{*catchall}",

new { controller = "Home", action = "Index", id = UrlParameter.Optional },

new { controller = "^H.*", action = "Index|About", httpMethod = new HttpMethodConstraint("GET") },

new[] { "URLsAndRoutes.Controllers" });
```

9.最后还是不爽的话自己写个类实现 IRouteConstraint的匹配方法。

```
using System;
  using System.Collections.Generic;
  using System.Linq;
  using System.Web;
  using System.Web.Routing;
  /// <summary>
  /// If the standard constraints are not sufficient for your needs, you can define your own custom constraints by
  /// </summary>
  public class UserAgentConstraint : IRouteConstraint
  {

    private string requiredUserAgent;
    public UserAgentConstraint(string agentParam)
    {
      requiredUserAgent = agentParam;
    }
    public bool Match(HttpContextBase httpContext, Route route, string parameterName,
    RouteValueDictionary values, RouteDirection routeDirection)
    {
      return httpContext.Request.UserAgent != null &&
      httpContext.Request.UserAgent.Contains(requiredUserAgent);
    }
  }
```

```
routes.MapRoute("ChromeRoute", "{*catchall}",

new { controller = "Home", action = "Index" },

new { customConstraint = new UserAgentConstraint("Chrome") },

new[] { "UrlsAndRoutes.AdditionalControllers" });
```

比如这个就用来匹配是否是用谷歌浏览器访问网页的。

10.访问本地文档

```
routes.RouteExistingFiles = true;

routes.MapRoute("DiskFile", "Content/StaticContent.html", new { controller = "Customer", action = "List", });
```

浏览网站，以开启 IIS Express，然后点显示所有应用程序-点击网站名称-配置（applicationhost.config）-搜索UrlRoutingModule节点

```
<add name="UrlRoutingModule-4.0" type="System.Web.Routing.UrlRoutingModule" preCondition="managedHandler,runtimeVer
```

把这个节点里的preCondition删除，变成

```
<add name="UrlRoutingModule-4.0" type="System.Web.Routing.UrlRoutingModule" preCondition="" />
```

11.直接访问本地资源，绕过了路由系统

```
routes.IgnoreRoute("Content/{filename}.html");
```

文件名还可以用 {filename}占位符。

IgnoreRoute方法是RouteCollection里面StopRoutingHandler类的一个实例。路由系统通过硬-编码识别这个Handler。如果这个规则匹配的话，后面的规则都无效了。 这也就是默认的路由里面routes.IgnoreRoute("{resource}.axd/{*pathInfo}");写最前面的原因。

路由测试（在测试项目的基础上，要装moq）

```
PM> Install-Package Moq
```

```
using System;
  using Microsoft.VisualStudio.TestTools.UnitTesting;
  using System.Web;
  using Moq;
  using System.Web.Routing;
  using System.Reflection;
  [TestClass]
  public class RoutesTest
  {
    private HttpContextBase CreateHttpContext(string targetUrl = null, string HttpMethod = "GET")
    {
      // create the mock request
      Mock<HttpRequestBase> mockRequest = new Mock<HttpRequestBase>();
      mockRequest.Setup(m => m.AppRelativeCurrentExecutionFilePath)
      .Returns(targetUrl);
      mockRequest.Setup(m => m.HttpMethod).Returns(HttpMethod);
      // create the mock response
      Mock<HttpResponseBase> mockResponse = new Mock<HttpResponseBase>();
      mockResponse.Setup(m => m.ApplyAppPathModifier(
      It.IsAny<string>())).Returns<string>(s => s);
      // create the mock context, using the request and response
      Mock<HttpContextBase> mockContext = new Mock<HttpContextBase>();
      mockContext.Setup(m => m.Request).Returns(mockRequest.Object);
      mockContext.Setup(m => m.Response).Returns(mockResponse.Object);
      // return the mocked context
      return mockContext.Object;
    }

    private void TestRouteMatch(string url, string controller, string action, object routeProperties = null, stri
    {
      // Arrange
      RouteCollection routes = new RouteCollection();
      RouteConfig.RegisterRoutes(routes);
      // Act - process the route
      RouteData result = routes.GetRouteData(CreateHttpContext(url, httpMethod));
      // Assert
      Assert.IsNotNull(result);
      Assert.IsTrue(TestIncomingRouteResult(result, controller, action, routeProperties));
    }

    private bool TestIncomingRouteResult(RouteData routeResult, string controller, string action, object propertyS
    {
      Func<object, object, bool> valCompare = (v1, v2) =>
      {
        return StringComparer.InvariantCultureIgnoreCase
        .Compare(v1, v2) == 0;
      };
      bool result = valCompare(routeResult.Values["controller"], controller)
      && valCompare(routeResult.Values["action"], action);
      if (propertySet != null)
      {
        PropertyInfo[] propInfo = propertySet.GetType().GetProperties();
        foreach (PropertyInfo pi in propInfo)
        {
          if (!(routeResult.Values.ContainsKey(pi.Name)
          && valCompare(routeResult.Values[pi.Name],
          pi.GetValue(propertySet, null))))
          {
            result = false;
```

```
            break;
          }
        }
      }
      return result;
    }

    private void TestRouteFail(string url)
    {
      // Arrange
      RouteCollection routes = new RouteCollection();
      RouteConfig.RegisterRoutes(routes);
      // Act - process the route
      RouteData result = routes.GetRouteData(CreateHttpContext(url));
      // Assert
      Assert.IsTrue(result == null || result.Route == null);
    }

    [TestMethod]
    public void TestIncomingRoutes()
    {
      // check for the URL that we hope to receive
      TestRouteMatch("~/Admin/Index", "Admin", "Index");
      // check that the values are being obtained from the segments
      TestRouteMatch("~/One/Two", "One", "Two");
      // ensure that too many or too few segments fails to match
      TestRouteFail("~/Admin/Index/Segment");//失败
      TestRouteFail("~/Admin");//失败
      TestRouteMatch("~/", "Home", "Index");
      TestRouteMatch("~/Customer", "Customer", "Index");
      TestRouteMatch("~/Customer/List", "Customer", "List");
      TestRouteFail("~/Customer/List/All");//失败
      TestRouteMatch("~/Customer/List/All", "Customer", "List", new { id = "All" });
      TestRouteMatch("~/Customer/List/All/Delete", "Customer", "List", new { id = "All", catchall = "Delete" });
      TestRouteMatch("~/Customer/List/All/Delete/Perm", "Customer", "List", new { id = "All", catchall = "Delete/Pe
    }



  }
```

最后还是再推荐一下Adam Freeman写的apress.pro.asp.net.mvc.4这本书。稍微熟悉MVC的从第二部分开始读好了。前面都是入门（对我来说是扯淡）。但总比国内某些写书的人好吧——把个开源项目的源代码下载下来帖到书上面来，然后标题起个深入解析XXXX，然后净瞎扯淡。最后一千多页的巨著又诞生了。Adam Freeman的风格我就很喜欢，都是实例写作，然后还在那边书里面专门写了大量的测试。

哎没办法啊，技术差距就是这样了。

推荐文章

- 1. ASP.NET Core CORS 简单使用
- 2. Lind.DDD.Domain.IOwnerBehavor对实体的意义
- 3. [原]ASP.NET MVC4 乱七八糟罗列
- 4. #Kotlin# 一年の使用报告 - 函数式思想
- 5. F#创建者Don Syme谈F#设计原则
- 6. Bypassing Applocker with MSBuild.exe

我来评几句

请输入评论内容...

登录后评论

已发表评论数(0)

相关站点

博客园
cnblogs.com

博客园-原创精华区
十订阅
热门文章

- 1. Lind.DDD.Domain.IOwnerBehavor对实体的意义
- 2. [原]ASP.NET MVC4 乱七八糟罗列
- 3. #Kotlin# 一年の使用报告 - 函数式思想
- 4. F#创建者Don Syme谈F#设计原则
- 5. Bypassing Applocker with MSBuild.exe

补充信息

提交

网站相关

- 关于我们
- 移动应用
- 建议反馈

关注我们

- 推酷网
- tuicool2012

友情链接

人人都是产品经理 PM256 移动信息化 行晓网 智城外包网 虎嗅 IT耳朵 创媒工场 经理人分享 市场部网 砍柴网 CocoaChina 北风网 云智慧 我赢职场 大数据时代 奇笛网 咕噜网 红联linux Win10之家 鸟哥笔记 爱游戏 投资潮 31会议网 极光推送 Teambition 硅谷网 leangoo ZEALER中国 OpenSNS 小牛学堂 handone Scrum中文网 比戈大牛 又拍云 更多链接 >>