.Net MVC Routing Catchall not working

Asked 16 years ago Modified 8 years, 10 months ago Viewed 8k times

16

I can't seem to figure this out. I'm experimenting with MVC Beta and am trying to implement a catchall route such that if the user enters mysite.com/blah instead of mysite.com/home/index it will hit the "Error" route.

Unfortunately it seems that the "Default" route always catches "blah" first. In fact the only route I've been able to get to the "Error" route with is blah/blah/blah/blah.

Is this the way it's supposed to work, because I've seen other examples that have the "Default" and "Error" route set up just like this and it seems that if they were to type in a controller that doesn't exist it would hit the "Error" route.

Is there something I'm missing (very possible) or will I just have to create a specific route for each controller?

Code I'm using:

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

        routes.MapRoute(
            "Error",
            "{*catchall}",
            new { controller = "Base", action = "Error", id = "404" }
        );
```

Thank you, Jeff

asp.net-mvc routes

Share

Improve this question

Follow

edited Dec 2, 2008 at 23:52

Ricky

5,377 ● 2 ● 33 ● 30

asked Nov 25, 2008 at 21:13

Jeff Keslinke

4,358 ● 5 ● 32 ● 50

4 Answers

Sorted by: Highest score (default)

6

Your first route will catch the most urls since you have defaults for the elements, you can visualize this using the route debugger from Phil Haack, see the link:

Route Debugger

Share Improve this answer Follow

answered Nov 25, 2008 at 23:18

Scott

850 ● 7 ● 13

6

In order to handle errors I used the Application_Error event in one of my projects:

```
protected void Application_Error(object sender, EventArgs e)
{
    Exception exception = Server.GetLastError();
    HttpException httpException = exception as HttpException;
    if (httpException != null)
    {
        RouteData routeData = new RouteData();
        routeData.Values.Add("controller", "Error");
        routeData.Values.Add("action", "HttpError500");

            if (httpException.GetHttpCode() == 404)
            {
                routeData.Values["action"] = "HttpError404";
            }

        Server.ClearError();
        Response.Clear();
        IController errorController = new ErrorController();
        errorController.Execute(new RequestContext(new
HttpContextWrapper(Context), routeData));
    }
}
```

Share

Improve this answer

Follow

edited Feb 16, 2016 at 6:30

F11

3,795 ● 12 ● 51 ● 93

answered Nov 25, 2008 at 22:17

Darin Dimitrov

1.0m ● 274 ● 3.3k ● 2.9k

This is the wrong place to do exception handling, and you're also changing the meaning of an exception by blowing it off. This could potentially make debugging very, very difficult. – Jeff Putz May 27, 2009 at 5:21

1 I don't see why using Application_Error handler would make debugging difficult. If you log the exception you will have the full stack trace. Another problem with the *catchall route is that it won't be called if an exception occurs in a controller action, so if you want to handle 500 errors you will need to do it somewhere else. I prefer having all exception handling code at one place. – Darin Dimitrov May 27, 2009 at 7:05

The biggest problem is that you lose a lot of your context by the time the error makes its way to Application_Error, such as the session object, and if you do not account for those things being gone, debugging becomes a nightmare. – Nick Larsen Oct 2, 2010 at 0:33

3

MVC routes are checked in the order that they are entered.

Mysite/blah will be found by the default route. The controller will be blah, and the action is index.

When you entered the mysite/blah/blah/blah/blah route you gave it a route it could not map the default route to and then your catchall route was called.

For those other examples, did you notice if they had some error filters setup? I'm pretty sure the default asp.net mvc site has some error handling attributes on the pages already.

Share

Improve this answer

Follow

answered Nov 25, 2008 at 22:08

community wiki
Min

Thanks, that's the understanding I came to, but just didn't want to accept based on what I've seen, and what I expect. But I'll learn to deal with it. – Jeff Keslinke Nov 25, 2008 at 23:07

0

This can also help when dealing with MVC catchall problems:

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

That is, where it says `{id}`, change it to `{*id}`. This allows the final id parameter to consume as much additional path as might be passed in. The default rule accepts this:

/person/name/joe

But not this:

/products/list/sortby/name

The second URL will throw a 404 without this modification to the route.

Share Improve this answer Follow

answered Jun 25, 2012 at 6:56

Chris Moschini
37.9k ● 19 ● 164 ● 194

I've tried this and been all over the internet trying to figure it out but it doesn't work for me. If I pass 2 segments in the URL I get the 404 error whether I add the asterisk or not. Any advice? – DavidHyogo Feb 12, 2013 at 14:43

This may be best as a new question - what are the defaults you're setting (third line in this example)? A full code sample would be best which requires a separate question. – Chris Moschini Feb 12, 2013 at 17:30

Good idea Chris, but I found my silly mistake. I was defining my routes in the wrong place for MVC 4, and they were being ignored. See my answer at stackoverflow.com/questions/7515644/… I'm now happily creating all kinds of fancy route definitions! – DavidHyogo Feb 14, 2013 at 11:38