Sql server, .net and c# video tutorial

Free C#, .Net and Sql server video tutorial for beginners and intermediate programmers.

Centralised 404 error handling in ASP.NET Core

Suggested Videos

Part 55 - Edit view in asp.net core mvc | Text | Slides
Part 56 - httppost edit action in asp.net core mvc | Text | Slides
Part 57 - Handling 404 not found in asp.net core mvc | Text | Slides

In this video we will discuss how to **handle 404 errors** i.e Page Not Found errors in a centralised way in asp.net core.

Along the way, we will discuss the following 3 middleware components that deal with status code pages in asp.net core

- UseStatusCodePages
- UseStatusCodePagesWithRedirects
- UseStatusCodePagesWithReExecute

Types of 404 errors

In ASP.NET Core there are 2 types of 404 errors that could happen

Complete Tutorials

JavaScript tutorial

Bootstrap tutorial

Angular tutorial for beginners

Angular 5 Tutorial for beginners

Important Videos

The Gift of Education

Web application for your business

Type 1 : Resource with the specified ID does not exit. We discussed how to handle this type of 404 errors and provide a more meaningful and a more customised error view in Part 57 of ASP.NET core tutorial.

Type 2 : The provided URL does not match any route in our application. In this video, we will discuss how to handle this type of 404 error in a centralised way.

404 error example in ASP.NET Core

The following is code in Configure() method of Startup class in Startup.cs file. As you might already know, this Configure() method configures the HTTP request processing pipeline for our asp.net core application.

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }

    app.UseStaticFiles();

    app.UseMvc(routes =>
    {
        routes.MapRoute("default", "{controller=Home}/{action=Index}/{id?}");
    });
}
```

Default 404 error page in ASP.NET Core

At the moment we do not have anything configured in this http request processing pipeline to handle 404 errors. So if we navigate to http://localhost/foo/bar, we see the following default 404 error page. This is because the URL /foo/bar does not match any routes in our application.

How to become .NET developer

Resources available to help you

Dot Net Video Tutorials

ASP.NET Core Tutorial

Angular 6 Tutorial

Angular CRUD Tutorial

Angular CLI Tutorial

Angular 2 Tutorial

Design Patterns

SOLID Principles

ASP.NET Web API

Bootstrap

AngularJS Tutorial

jQuery Tutorial

JavaScript with ASP.NET Tutorial

JavaScript Tutorial

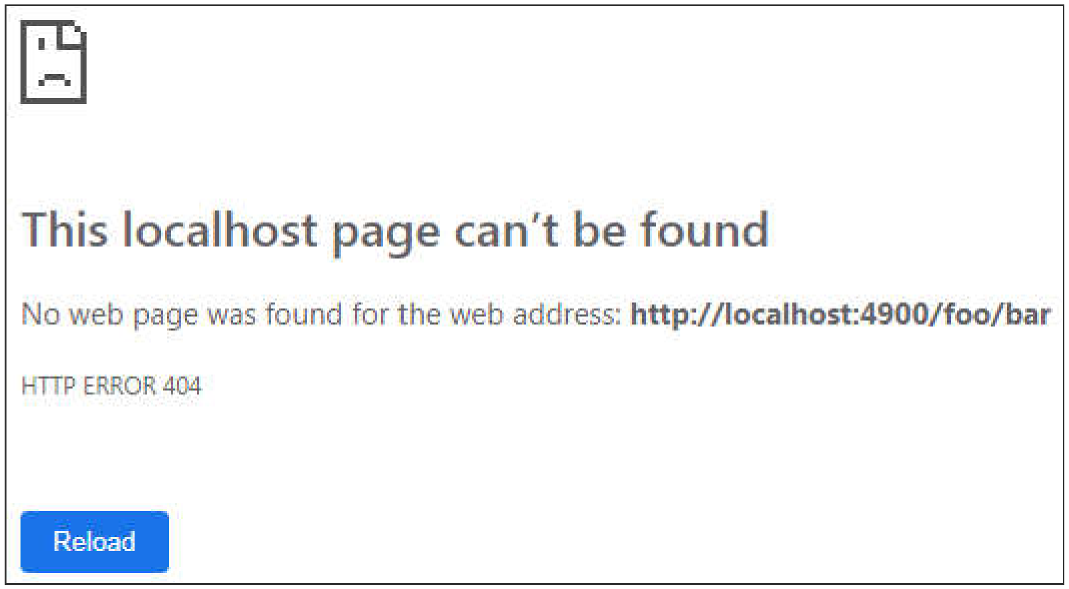
Charts Tutorial

LINQ

LINQ to SQL

LINQ to XML

Entity Framework



This localhost page can't be found

No web page was found for the web address: **http://localhost:4900/foo/bar**

HTTP ERROR 404

Handling non-success http status codes

To handle non-success http status codes such as 404 for example, we could use the following 3 built-in asp.net core middleware components.

UseStatusCodePages
UseStatusCodePagesWithRedirects
UseStatusCodePagesWithReExecute

UseStatusCodePages Middleware

This is the least useful of the 3 status code middleware components. For this reason, we rarely use it in a real world production application. To use it in an application and see what it can do, plug it into the http processing pipeline as shown below.

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
```

WCF

ASP.NET Web Services

Dot Net Basics

C#

SQL Server

ADO.NET

ASP.NET

GridView

ASP.NET MVC

Visual Studio Tips and Tricks

Dot Net Interview Questions

Slides

Entity Framework

WCF

ASP.NET Web Services

Dot Net Basics

C#

SQL Server

ADO.NET

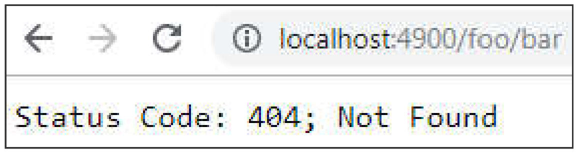
ASP.NET

```
    else
    {
        app.UseStatusCodePages();
    }

    app.UseStaticFiles();

    app.UseMvc(routes =>
    {
        routes.MapRoute("default", "{controller=Home}/{action=Index}/{id?}");
    });
}
```

With UseStatusCodePages Middleware configured, if we navigate to http://localhost/foo/bar, it returns the following simple text response.



```
Status Code: 404; Not Found
```

UseStatusCodePagesWithRedirects Middleware

In a production quality application we want to intercept these non-success http status codes and return a custom error view. To achieve this, we can either use UseStatusCodePagesWithRedirects middleware or UseStatusCodePagesWithReExecute middleware.

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseStatusCodePagesWithRedirects("/Error/{0}");
    }

    app.UseStaticFiles();
```

GridView

ASP.NET MVC

Visual Studio Tips and Tricks

Java Video Tutorials

Part 1 : Video | Text | Slides

Part 2 : Video | Text | Slides

Part 3 : Video | Text | Slides

Interview Questions

C#

SQL Server

Written Test

```
    app.UseMvc(routes =>
    {
        routes.MapRoute("default", "{controller=Home}/{action=Index}/{id?}");
    });
}
```

With the following line in place, if there is a 404 error, the user is redirected to /Error/404. The placeholder {0}, in "/Error/{0}" will automatically receive the http status code.

```
app.UseStatusCodePagesWithRedirects("/Error/{0}");
```

ErrorController

```
public class ErrorController : Controller
{
    // If there is 404 status code, the route path will become Error/404
    [Route("Error/{statusCode}")]
    public IActionResult HttpStatusCodeHandler(int statusCode)
    {
        switch (statusCode)
        {
            case 404:
                ViewBag.ErrorMessage = "Sorry, the resource you requested could not be
found";
                break;
        }

        return View("NotFound");
    }
}
```

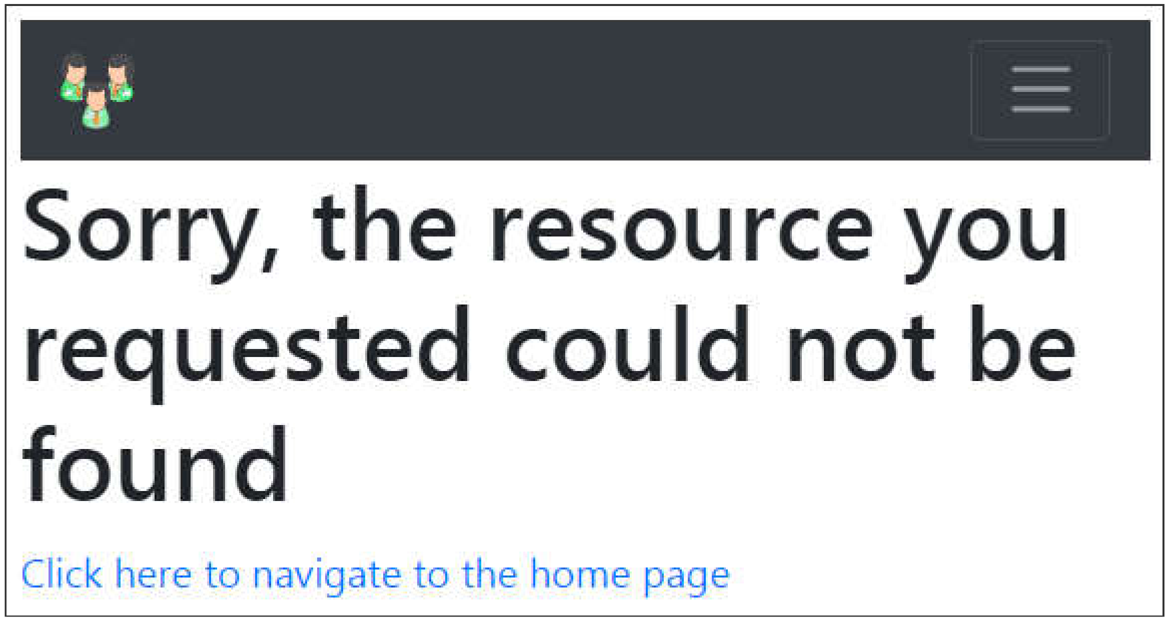
NotFound View

```
@{
    ViewBag.Title = "Not Found";
}

<h1>@ViewBag.ErrorMessage</h1>
```

```
<a asp-action="index" asp-controller="home">
    Click here to navigate to the home page
</a>
```

At this point, if we navigate to http://localhost/foo/bar we see the following custom 404 error view NotFound.cshtml as expected.



Sorry, the resource you requested could not be found

Click here to navigate to the home page

To use UseStatusCodePagesWithReExecute middleware instead of UseStatusCodePagesWithRedirects middleware

REPLACE app.UseStatusCodePagesWithRedirects("/Error/{0}");
WITH app.UseStatusCodePagesWithReExecute("/Error/{0}");

Re-run the application and navigate to http://localhost/foo/bar, we see the same custom 404 error view NotFound.cshtml.

The obvious question that comes to our mind at this point is, what's the difference between these 2 middleware components and which one should we be using. We will answer these 2 questions in our next video.

No comments:

Post a Comment

If you like this website, please share with your friends on facebook and Google+ and recommend us on google using the g+1 button on the top right hand corner.

Links to this post

Create a Link

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)