Routing in ASP.NET Core MVC | Conventional Based Routing

.

What is Routing?

- The Routing in ASP.NET Core MVC application is a mechanism in which it will inspect the incoming Requests (i.e. URLs) and then mapped that request to the controllers and their action methods. This mapping is done by the routing rules which are defined for the application. We can do this by adding the Routing Middleware to the request processing pipeline.
- So, the ASP.NET Core Framework maps the incoming Requests i.e. URLs to the Controllers action methods based on the routes configured in your application. You can configure multiple routes for your application and for each route you can also set some specific configurations such as default values, constraints, message handlers, etc.

Types of Routing supported by ASP.NET Core MVC

- In ASP.NET Core MVC application, you can define routes in two ways. They are as follows:
  - **Convention Based Routing**
  - **Attribute-Based Routing.**

Conventional Based Routing?

- In Conventional Based Routing, the route is determined based on the conventions defined in the route templates which will map the incoming Requests (i.e. URLs) to controllers and their action methods.
- In ASP.NET Core MVC application, the Convention based Routes are defined within the Configure method of the Startup.cs class file.

Understanding Conventional Based Routing in ASP.NET Core MVC:

- In ASP.NET Core MVC application, it is the controller action method that is going to handle the incoming Requests i.e. URLs. For example, if we issue a request to the "**/Home/Index**" URL, then it is the Index action method of Home Controller class which is going to handle the request as shown in the below image

http://localhost:52190/Home/Index

```
public class HomeController : Controller
{
    public ViewResult Index()
    {
        return View();
    }
}
```

Understanding the Default Route in ASP.NET Core MVC Application:

- As we already discussed in our previous lecture that we can add the required MVC middleware into the request processing pipeline either by calling the **UseMvcWithDefaultRoute()** method or by calling the **UseMvc()** method within in the Configure() method of the **Startup.cs** class file as shown in the below image.
- As of now, we are using the **UseMvcWithDefaultRoute()** middleware.

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

    app.UseMvcWithDefaultRoute();
}
```

Adding MVC middleware to the Request Processing Pipeline

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Ex4_Routings Sign in

Debug Any CPU IIS Express Live Share

Toolbox

Index.cshtml | HomeController.cs | Startup.cs | Ex4_Routings | Object Browser

Ex4_Routings | Ex4_Routings.Startup | ConfigureServices(IServiceCollection service

```
10  namespace Ex4_Routings
11  {
12      public class Startup
13      {
14          // This method gets called by the runtime. Use this method to add servic
15          // For more information on how to configure your application, visit http
16          public void ConfigureServices(IServiceCollection services)
17          {
18              services.AddMvc();
19          }
20
21          // This method gets called by the runtime. Use this method to configure
22          public void Configure(IApplicationBuilder app, IHostingEnvironment env)
23          {
24              if (env.IsDevelopment())
25              {
26                  app.UseDeveloperExceptionPage();
27              }
28              app.UseMvcWithDefaultRoute();//Right click and go to defination  pip
29              app.Run(async (context) =>
30              {
31                  await context.Response.WriteAsync("Hello World!");
32              });
33          }
34      }
```

120 % No issues found Ln: 18 Ch: 31 SPC CRLF

Output

```
\Ex4_Routings\bin\Debug\netcoreapp2.1\Ex4_Routings.dll
1>Ex4_Routings -> C:\Users\Lenovo\Desktop\Asp.net_Core\Examples\Ex4_Routings\bin\Debug\netcoreapp2.1\Ex4_Routings.Views.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'Ex4_Routings' (1 of 1 project)
  - Ex4_Routings
    - Connected Services
    - Dependencies
    - Properties
    - wwwroot
    - Controllers
      - HomeController.cs
    - Views
      - Home
        - Index.cshtml
    - Program.cs
    - Startup.cs

Build succeeded Add to Source Control

Type here to search ENG 10:36 20-01-2021

Understanding The Route Template:

- The default route template maps most URLs that have the following pattern.
  - **http://localhost:52190/Student/Details/2**
- The first segment path of the URL i.e. "**/Student**" is mapped to the "**StudentController**". As you can see in the URL we do not have the word Controller with the first segment path of the URL. But it maps to the StudentController, this is because when ASP.NET Core MVC Framework finds the word /Student as the first segment path of URL, then it internally appends the word Controller.
- The second segment path of the URL i.e. "**/Details**" is mapped to the "**Details(int id)**" action method of the HomeController class and the third segment path of the URL i.e. "**2**" is mapped to the "**id**" parameter of the Details(int id) action method.
- As you can see in the default route template **"{controller=Home}/{action=Index}/{id?}"**, we have a question mark at the end of the id parameter which makes the parameter id as optional.

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Ex4_Routings Sign in

Debug Any CPU IIS Express Live Share

Index.cshtml | TestController.cs | Index.cshtml | HomeController.cs | Startup.cs | Ex4_Routings

Ex4_Routings | Ex4_Routings.Controllers.TestController | Details1(int id)

```
using Microsoft.AspNetCore.Mvc;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace Ex4_Routings.Controllers
{
    public class TestController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }
        //http://localhost:60153/Test/Details?name=raj
        public string Details(string name)
        {
            return "Name=" + name;
        }
        //http://localhost:60153/Test/Details1/101
        public string Details1(int id)
        {
            return "ID=" + id;
        }

```

120 % No issues found Ln: 20 Ch: 51 SPC CRLF

Output

```
\Ex4_Routings\bin\Debug\netcoreapp2.1\Ex4_Routings.dll
1>Ex4_Routings -> C:\Users\Lenovo\Desktop\Asp.net_Core\Examples\Ex4_Routings\bin\Debug\netcoreapp2.1\Ex4_Routings.Views.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'Ex4_Routings' (1 of 1 project)
  - Ex4_Routings
    - Connected Services
    - Dependencies
    - Properties
    - wwwroot
    - Controllers
      - HomeController.cs
      - TestController.cs
    - Views
      - Home
        - Index.cshtml
      - Shared
      - Test
    - Program.cs
    - Startup.cs

Item(s) Saved Add to Source Control

Type here to search ENG 10:46 20-01-2021

Thanks