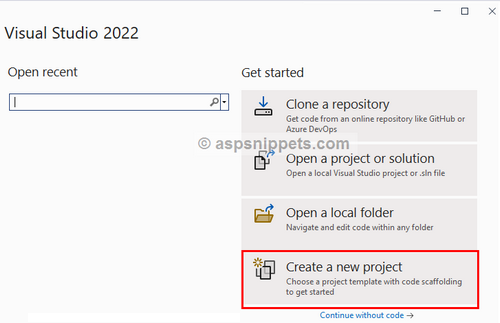
In this article I will provide a short Hello World Tutorial using a small Sample Program example on how to use and develop applications in ASP.Net Core Razor Pages 3.1 for the first time.

This article makes use of Visual Studio 2022 for developing the ASP.Net Core 3.1 Razor Pages Hello World Sample program example.

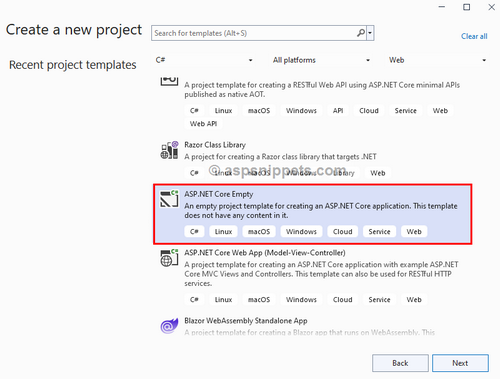
**Creating a new ASP.Net Core Razor Pages 3.1 Project**

Let’s get started with creating your first ASP.Net Core Razor Pages 3.1 Project.

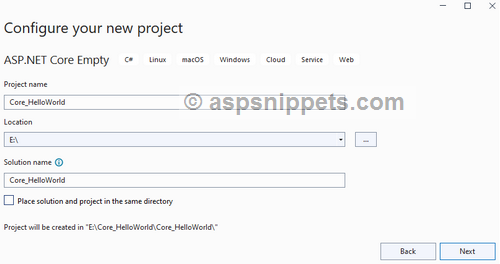
1. Open Visual Studio and from Start section click on Create a new project.



2. From the Create a new project Dialog window, select ASP.NET Core Empty option and then click on Next.



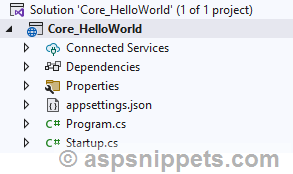
3. Then, you can set a suitable Name for your project and also you can set its Location where you want to create the Project and then click on Next.



4. From the ASP.NET Core Empty Dialog, select .NET Core 3.1 from Framework options and then click on Create.

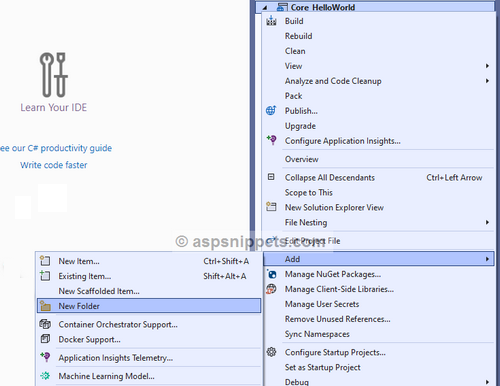


5. Your first Hello World ASP.NET Core Web Application Project is now ready and you should see the following folders and files in your Solution Explorer window.

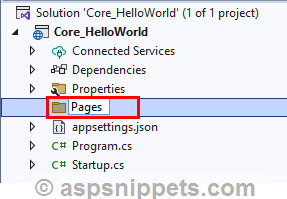


**Adding Razor Page to the ASP.Net Core 3.1 Project**

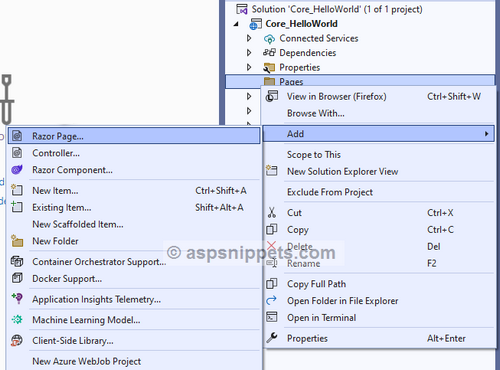
1. Inside the Solution Explorer window, Right Click on the Project and then click on Add and then New Folder option from the Context Menu.



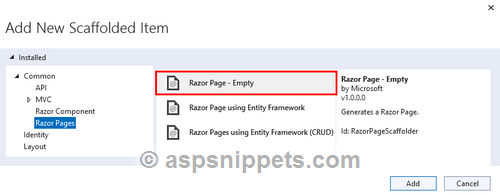
2. Name the newly added Folder as Pages.



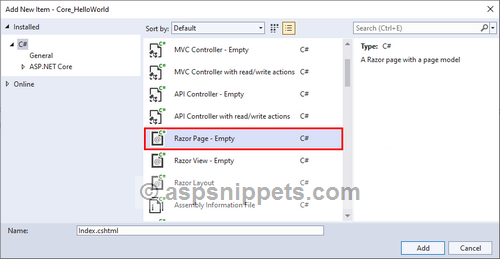
3. Now, inside the Solution Explorer window, Right Click on the Pages folder and then click on Add and then click Razor Page option from the Context Menu.



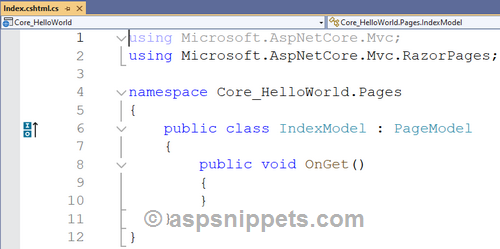
4. From the Add New Scaffolded Item Dialog window, select the “Razor Page - Empty” option and then click on Add button.



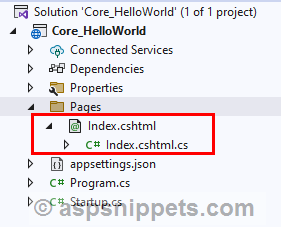
5. After adding the “Razor Page - Empty” option you can provide a suitable Name to it and then click on Add button.



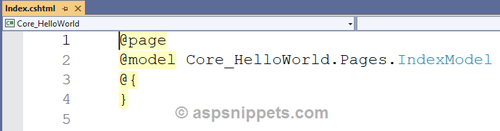
Once you click on add, the following IndexModel is created. The default Handler method is “Get”.



Inside the Solution Explorer window, the corresponding Razor HTML (Index.cshtml) Page is also shown.



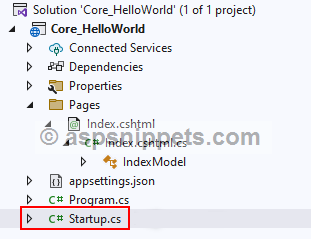
And when the Razor HTML Page is opened it will look as shown below.



**Configuring the Routes**

The last important part is to configure the Routes in the Startup.cs file.

1. Open the Startup.cs file from the Solution Explorer window.



2. Inside the Startup.cs, there are two configurations.

ConfigureServices

Inside this method, you will have to add the following code which will instruct the program to add MVC services.

public void ConfigureServices(IServiceCollection services)

{

   services.AddMvc();

}

Configure

Inside this method, the default settings for the Razor Pages are configured.

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

    if (env.IsDevelopment())

    {

        app.UseDeveloperExceptionPage();

    }

    app.UseRouting();

    app.UseEndpoints(endpoints =>

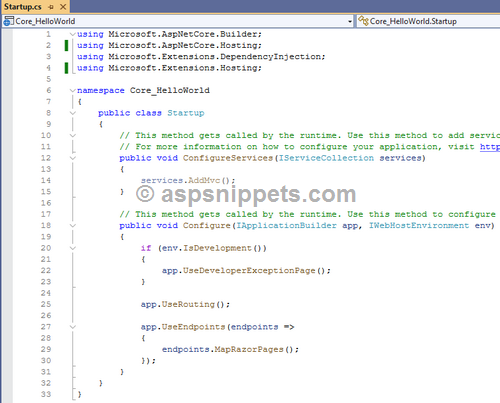
    {

        endpoints.MapRazorPages();

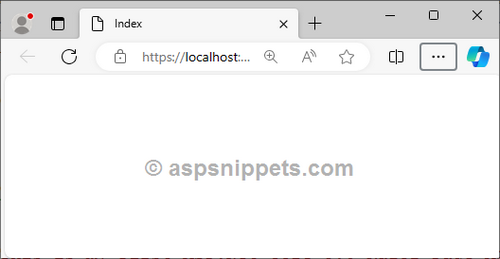
    });

}

Finally, the Startup.cs file will look as shown below.



Now press F5 to run the Application and you should see a blank page in browser.



**Displaying a Message from PageModel to Razor Page in ASP.Net Core 3.1 Project**

1. Inside the Razor PageModel's OnGet Handler method, the public property named "Message" is set.

public class IndexModel : PageModel

{

    public string Message { get; set; }

    public void OnGet()

    {

        this.Message = "This is my First ASP.Net Core 3.1 Razor Page App.";

    }

}

2. Then, inside the Razor HTML Page, the public property Message is accessed from the Razor PageModel class and displayed.

@page

@model Core\_HelloWorld.Pages.IndexModel

@{

    Layout = null;

}

<!DOCTYPE html>

<html>

<head>

    <meta name="viewport" content="width=device-width" />

    <title>Index</title>

</head>

<body>

    <h3>@Model.Message</h3>

</body>

</html>

3. Now press F5 to run the application and you should see a message displayed on page in browser.

