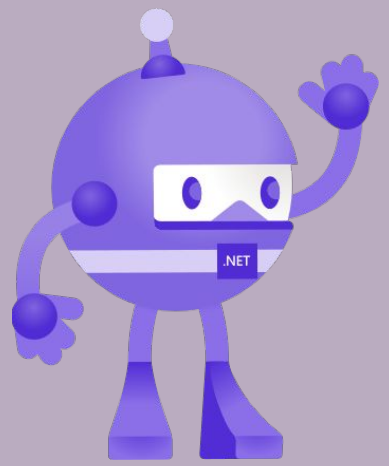
Asp.Net Core – Razor Pages

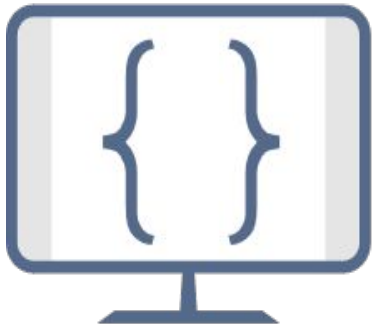
ASP.NET CORE WEB DEVELOPER PROGRAM @ AITRICH ACADEMY



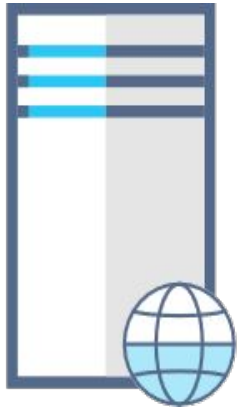
Aitrich

What are Razor Pages?

Razor Pages is a server-side, page-centric programming model for building web UI with ASP.NET Core.



Client Content



Server Code

Razor Pages

An ASP.NET Razor page has the ".cshtml" (e.g., "Index.cshtml") file extension.

This file contains a mix of HTML and Razor syntax. The Razor syntax is actually C# code mixed together with the HTML code.

The Razor parts of the file are executed on the web server before the page is sent to the client (your web browser).

The Razor page may also have a C# code file linked to it, this file has the extension

In Razor with Page Model each Razor page is a pair of files:

Benefits of Razor Pages

Makes it easy to get started building dynamic web apps with html,css and c#

Encourages organization of files by feature, which eases maintenance of your app

Html + server-side c# code by using Razor syntax

When to use Razor Pages

Blazor

Appropriate for client-heavy SPA(single page app) scenarios as React and Angular

MVC

Server-side,organized by controllers,more structured scenarios,original basis of Razor pages

Razor Pages

Server-side applications that are organized by feature,Easy to start,scales to large applications.

Razor Pages

Razor page use a markup language called Razor for embedding server-based code into webpages.

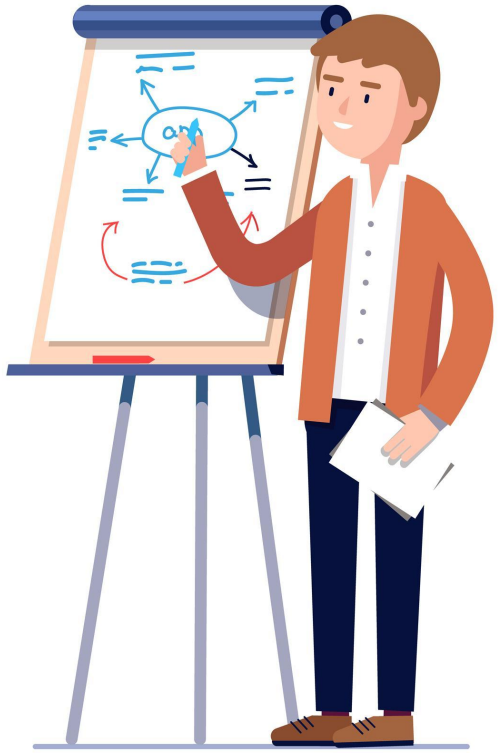
Razor syntax is combination of html and c# where the c# code defines the dynamic rendering logic for the page.

In a webpage that uses the razor syntax,there can be two kind of content:Client content and Server Code

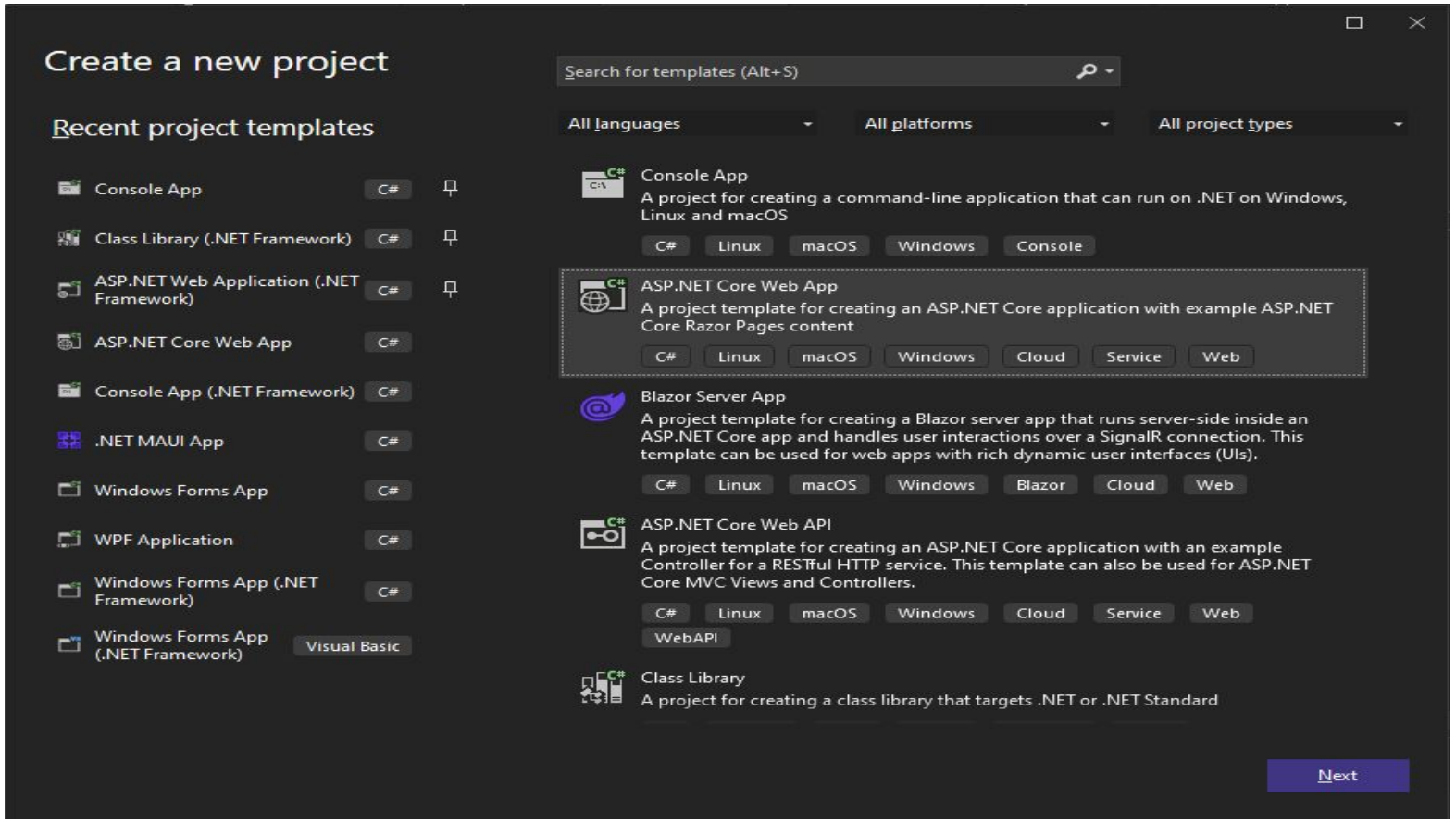
Razor Pages

Client Content : Contains what you are used to in webpages.For example,HTML markup (elements),style information such as CSS, maybe some client script such as javascript , and plain text

Server Code : Razor enables you to add server code to your client contrent.if there is server code in the page,the server runs that code first , before it sends the page to the browser



Creating a ASP.NET Core Web App



Project Structure

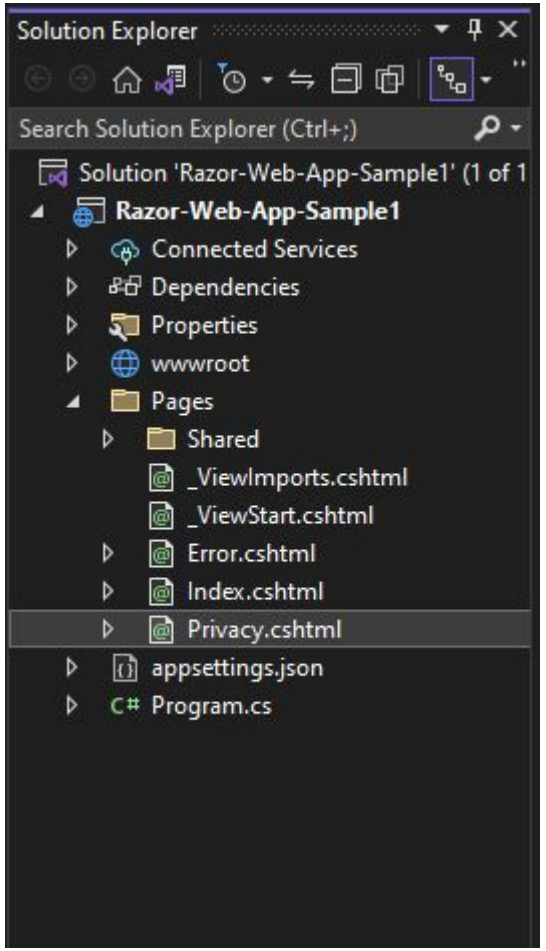
appSettings.json – This file contains configuration data, such as connection strings.

Program.cs – This file contains the entry point for the program. ,it configures app behavior.

wwwroot folder - Contains static files, such as Images, HTML files, JavaScript files, and CSS files.

Pages folder – Here you are supposed to put your ASP.NET (".cshtml") web pages

_Layout.cshtml file configures UI elements common to all pages. You can use this file to set up the navigation menu at the top of the page



Razor Pages is enabled in program.cs

AddRazorPages adds services for Razor Pages to the app.

MapRazorPages adds endpoints for Razor Pages to the IEndpointRouteBuilder.

Program.cs

Razor-Web-App-Sample1

```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddRazorPages();

var app = builder.Build();

// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Error");
    // The default HSTS value is 30 days. You may want to change thi
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.MapRazorPages();

app.Run();
```

Razor page Anatomy

In Razor with Page Model each Razor page is a pair of files:

i. A ".cshtml" file that contains HTML markup with C# code using Razor syntax.

ii. A ".cshtml.cs" ("code behind" or "Page model" file) file that contains C# code that handles page events.

Razor page – .cshtml page

This code looks a lot like a Razor view file used in an ASP.NET Core app with controllers and views. What makes it different is the @page directive.

- **@page** makes the file into an MVC action, which means that it handles requests directly, without going through a controller. **@page** must be the first Razor directive on a page. **@page** affects the behavior of other Razorn constructs. Razor Pages file names have a .cshtml suffix.

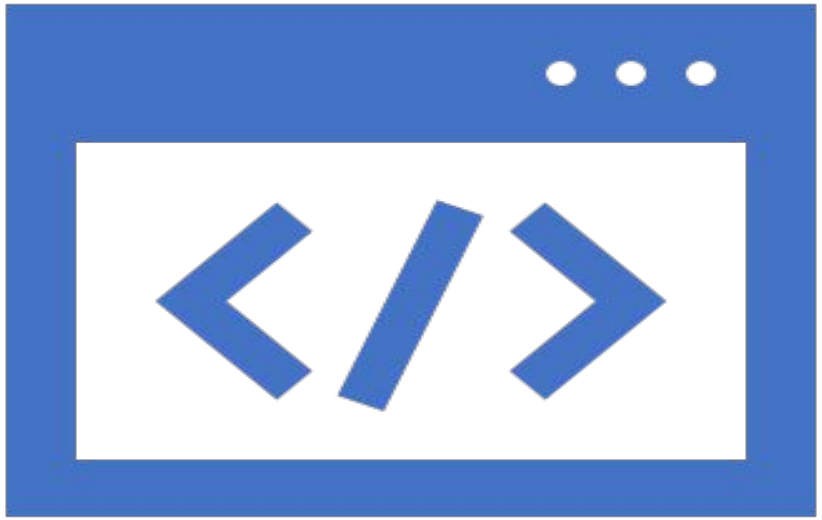
- The **@model** directive specifies the model type made available to the Razor page

Index.cshtml

```
1   @page
2   @model IndexModel
3   @{
4       ViewData["Title"] = "Home page";
5   }
6
7   <div class="text-center">
8       <h1 class="display-4">Hire Me Now</h1>
9
10  </div>
11
```

Rendering HTML

The default Razor language is HTML. Rendering HTML from Razor markup is no different than rendering HTML from an HTML file. HTML markup in .cshtml Razor files is rendered by the server unchanged.



Razor syntax

Razor is a markup syntax for embedding .NET based code into webpages. The Razor syntax consists of Razor markup, C#, and HTML. Files containing Razor generally have a **.cshtml** file extension. Razor is also found in Razor component files (**.razor**). Razor syntax is similar to the templating engines of various JavaScript single-page application (SPA) frameworks, such as Angular, React, VueJs, and Svelte.

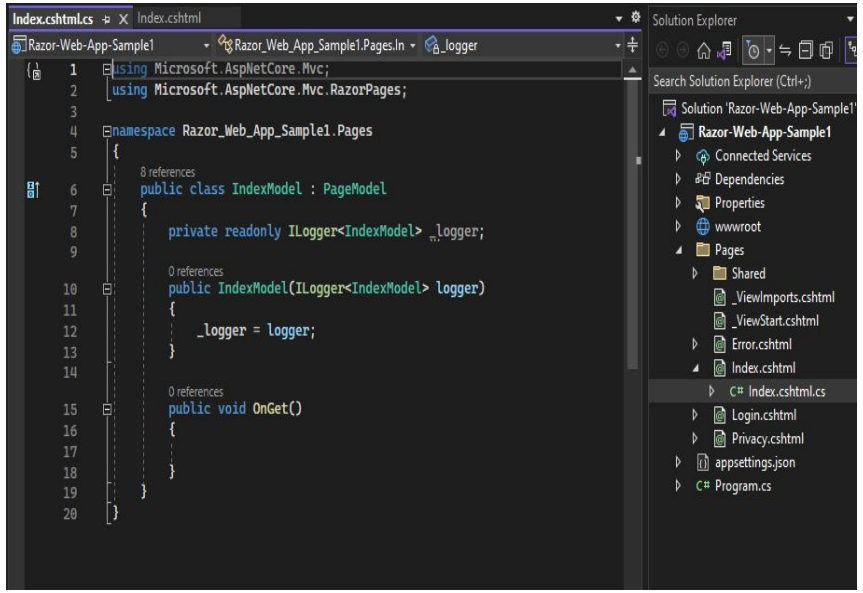
Razor supports C# and uses the @ symbol to transition from HTML to C#. Razor evaluates C# expressions and renders them in the HTML output.

When an @ symbol is followed by a Razor reserved keyword, it transitions into Razor-specific markup. Otherwise, it transitions into plain HTML.

To escape an @ symbol in Razor markup, use a second @ symbol:

PageModel - .cshtml.cs page

- OnPost to handle form submissions.
- OnGet to initialize state needed for the page.
- Defines page handlers for HTTP requests sent to the page and for the data used to render the page.
- Encapsulates the data properties and logic operations scoped just to its Razor page.
- Page events can be manage here
- This is the logic Part of the page .
- By convention, the PageModel class file has the same name as the Razor Page file with .cs appended.




thank you