100 XP

Explore your Visual Studio App Project

5 minutes

You've successfully created your web app and published it to Azure, but what happens when you want to make changes? Visual Studio remembers where the app is published, which makes updating and changing your app a two-click process.

Explore the project structure

You've created an ASP.NET Core web app in Visual Studio, and now you'll need to edit and customize your website. Let's explore the project structure to see what Visual Studio has created.

Dependencies

The dependencies folder contains the ASP.NET Core internals to get your app up and running. Unless you're adding specific third-party packages, you won't need to spend much time in this folder.

Properties

The properties folder contains configuration data for where you're hosting your web app. If you expand the **PublishProfiles** folder now, you should see the URL for the Alpine Ski Hill site. Each publishing profile is an .xml file containing publishing configuration information, such as the Azure address that Visual Studio uses to upload your files.

wwwroot

The wwwroot file contains all of your static assets for your site, such as the css, js, images, and lib files. When you are ready to style and add more functionality to your site, you will work in here.

Pages

The **Pages** folder has ***Razor*** templates for the pages of your site. Razor is a markup syntax for embedding server code into ASP.NET web pages. It includes HTML, and has special conventions for displaying data and executing logic on your site.

Each page in your site is represented with two code files:

- A `.cshtml` file, which is the Razor markup file. This file contains your display HTML and some C# logic.
- A `.cs` file, which is the C# code-behind that inherits from the `PageModel` class. This file allows you to intercept HTTP requests and do some processing on that request before passing any data to the Razor file.

appsetting.json

This is a configuration file for ASP.NET Core.

Program.cs

The Program.cs file configures and launches the web host for your site.

Introduction to Razor templates

We will want to make some basic changes to our website. You'll need to have a basic understanding of how to leverage Razor templates to customize your web app.

What is Razor?

Razor is an ASP.NET syntax used to create dynamic web pages with C#. When a server reads a

Razor page, the C# code is run before it renders the HTML. This allows you to generate dynamic content quickly.

Razor uses `@` directives to tell ASP.NET how to process a page.

For example, take a look at the code in the `Privacy.cshtml` page:

ASP.NET (C#)

```
@page
@model PrivacyModel
@{
    ViewData["Title"] = "Privacy Policy";
}
<h1>@ViewData["Title"]</h1>

<p>Use this page to detail your site's privacy policy.</p>
```

- The `@page` directive is telling ASP.NET to process this file as a Razor page.
- The `@model` directive is telling ASP.NET to tie this Razor page with a C# class called `PrivacyModel`.

Razor also uses the `@` symbol to transition between code and HTML. If you look at the preceding snippet, you'll notice `@{ ... }`. This is a **Razor code block**, which is *executed but not rendered*.

To render the output of a code statement, use the `@` in front of a C# expression. There's an example in the preceding code block in the `<h1>` tag.

Creating and publishing a website are just the first steps in creating a good website. Once you start to add content, you'll need to update your site. Once you've published your site to Azure, you can update it at any time.

Next unit: Exercise - Publish an update to your site

Continue >