

Getting Started with DordStream Studio By a developer

This tutorial teaches the basics of building a DordStream code library (Themes) for web software/ application. We recommend you to host DordStream CMS software on your local server (Computer) (internet information services (IIS)). before starting this tutorial.

Prerequisites

Install the following:

- [DordStream Studio.](#)

Languages required:

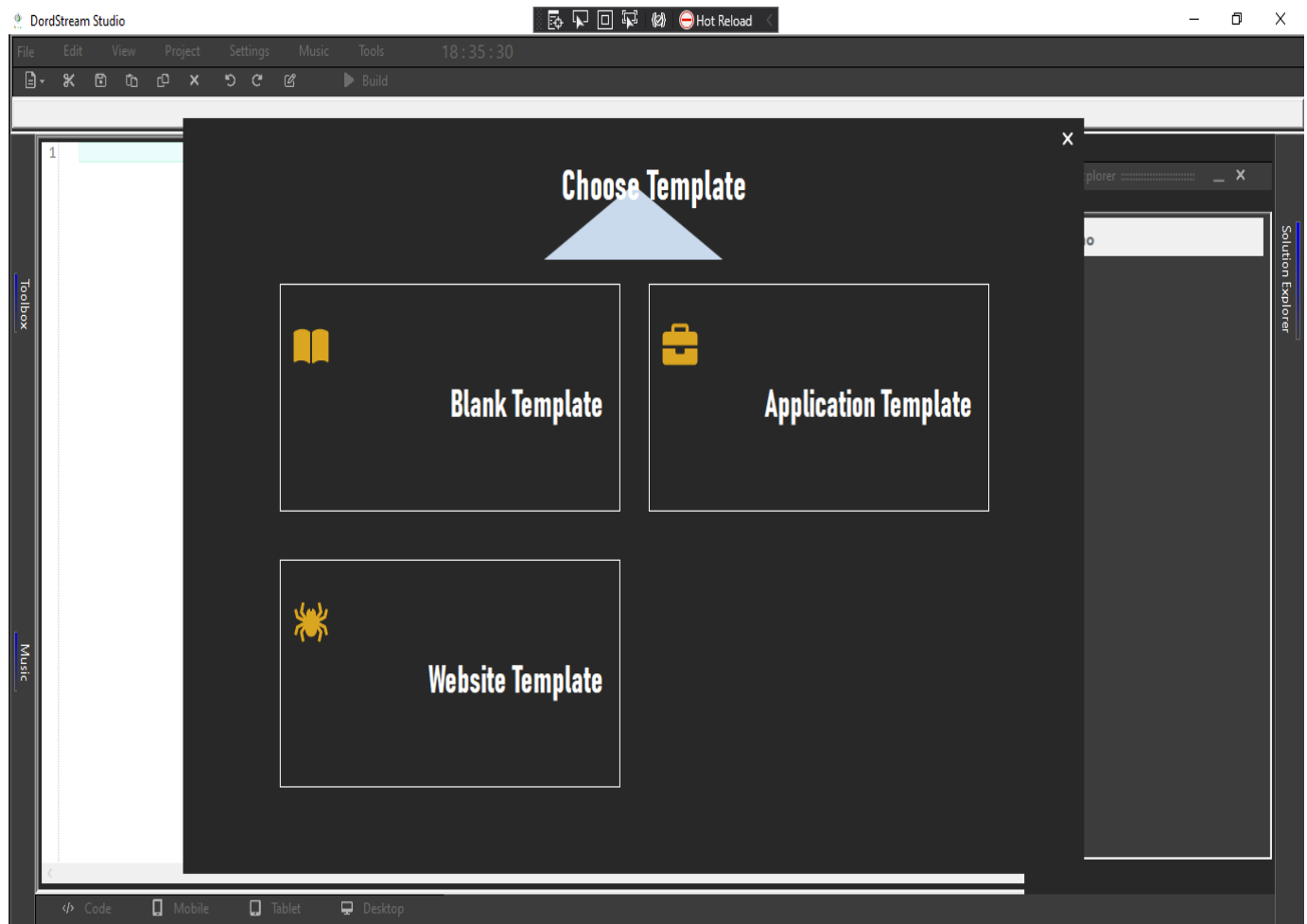
- [Html](#)
- [Css](#)
- [C#](#)

Create a Project

From the DordStream Studio **File** menu, select **New > Project**.

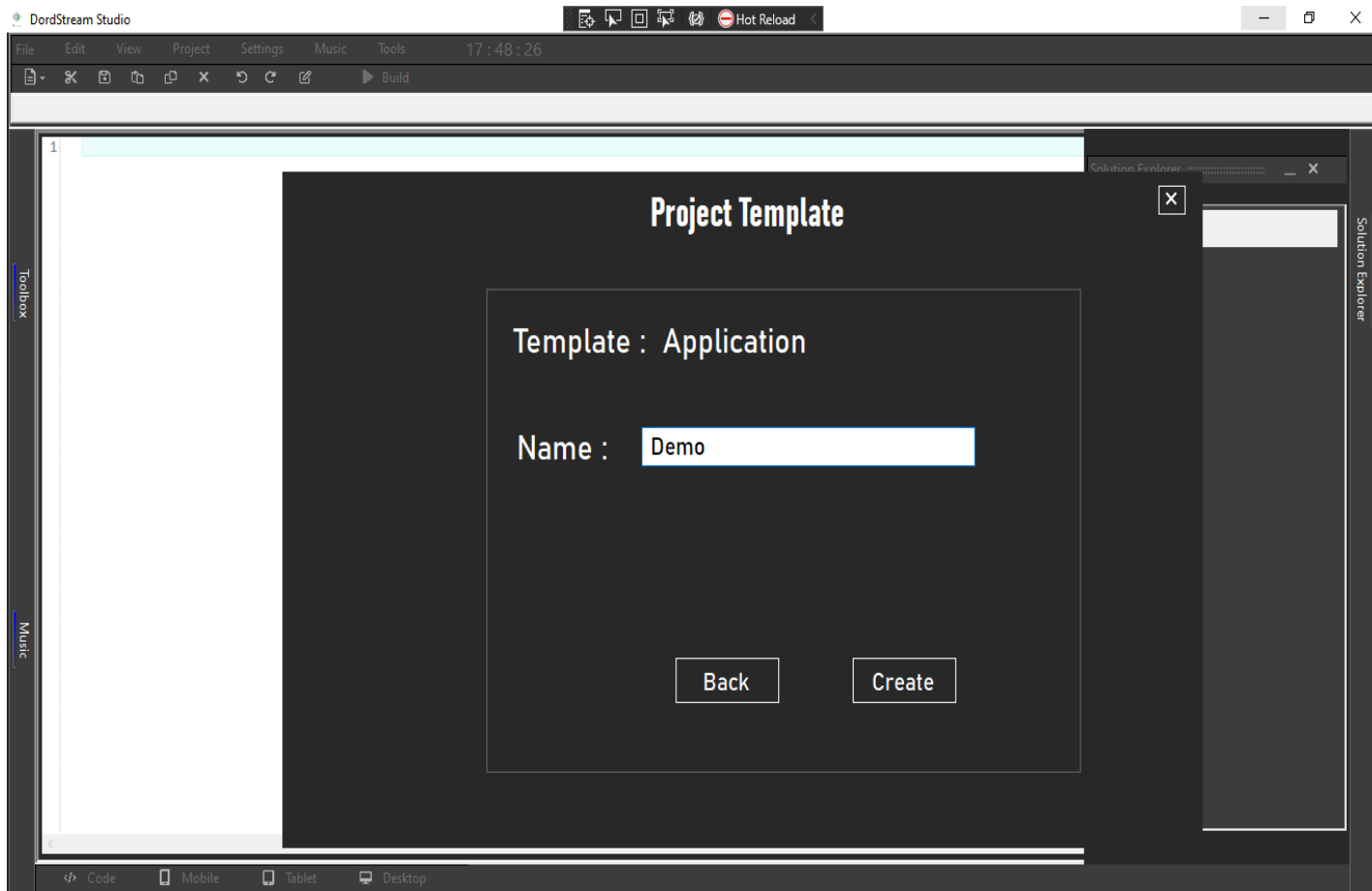
Create a new project and select *Application Template*, give the project name. In this tutorial we will Name the project as **Demo**.

- Select *Application Template*, and name the project.

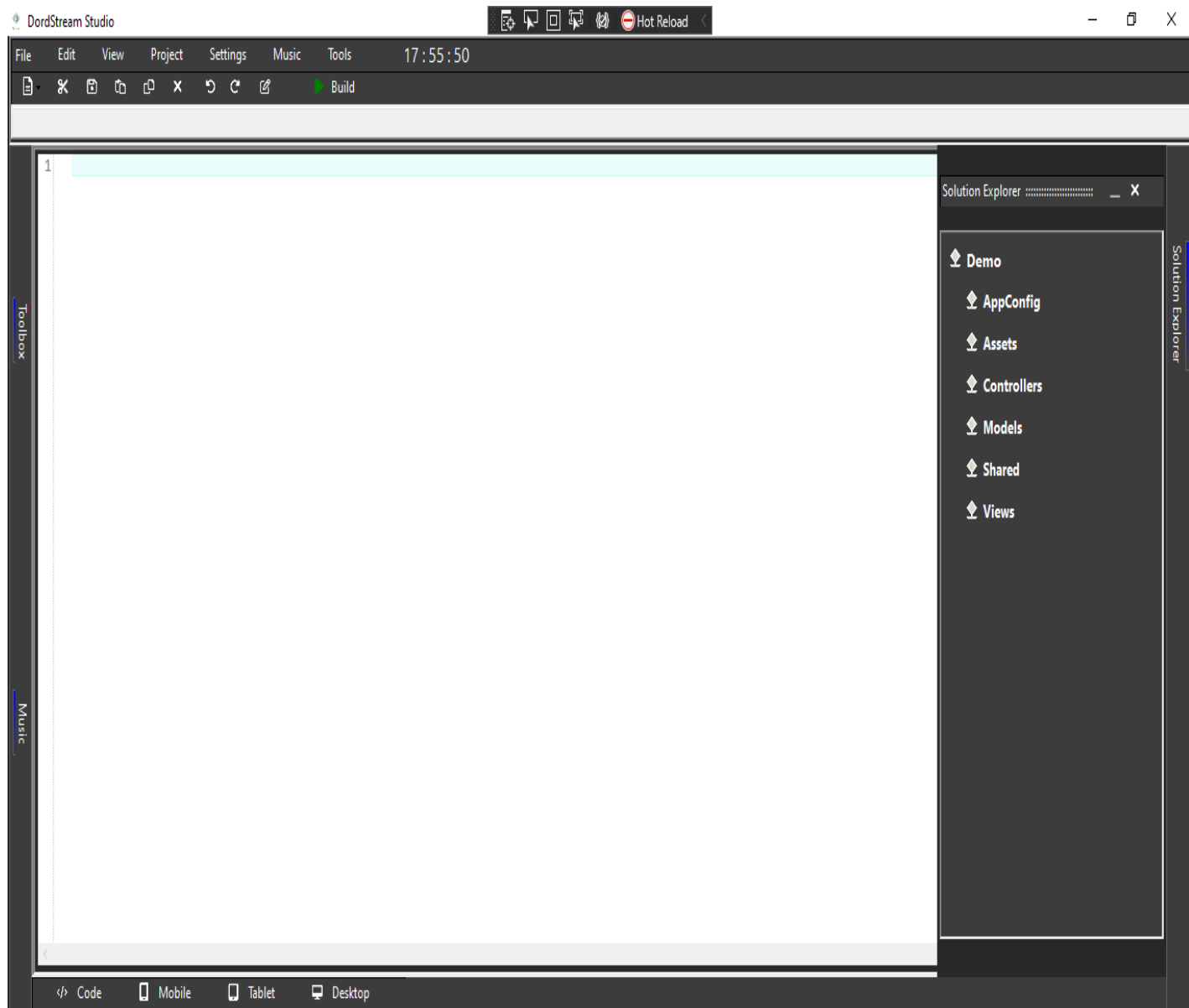


- In the picture above, there are three types of template which are Blank Template, Application Template and Website Template.
- **Blank Template:** This is an empty template which has no default files and folders. the template is used when creating a project from scratch. it can be used by a designer and a developer.
- **Application Template:** This is a template with default files and folders which is mostly used developers for developing a web application. Before using this template, the programming language needed are, C#, Html and CSS. this template make uses of MVC architectural design pattern. the examples of a web application that can be developed using this template are ecommerce store, hospital software, library software, Attendance Sheet software (Facial, Fingerprint or input), Web API etc.

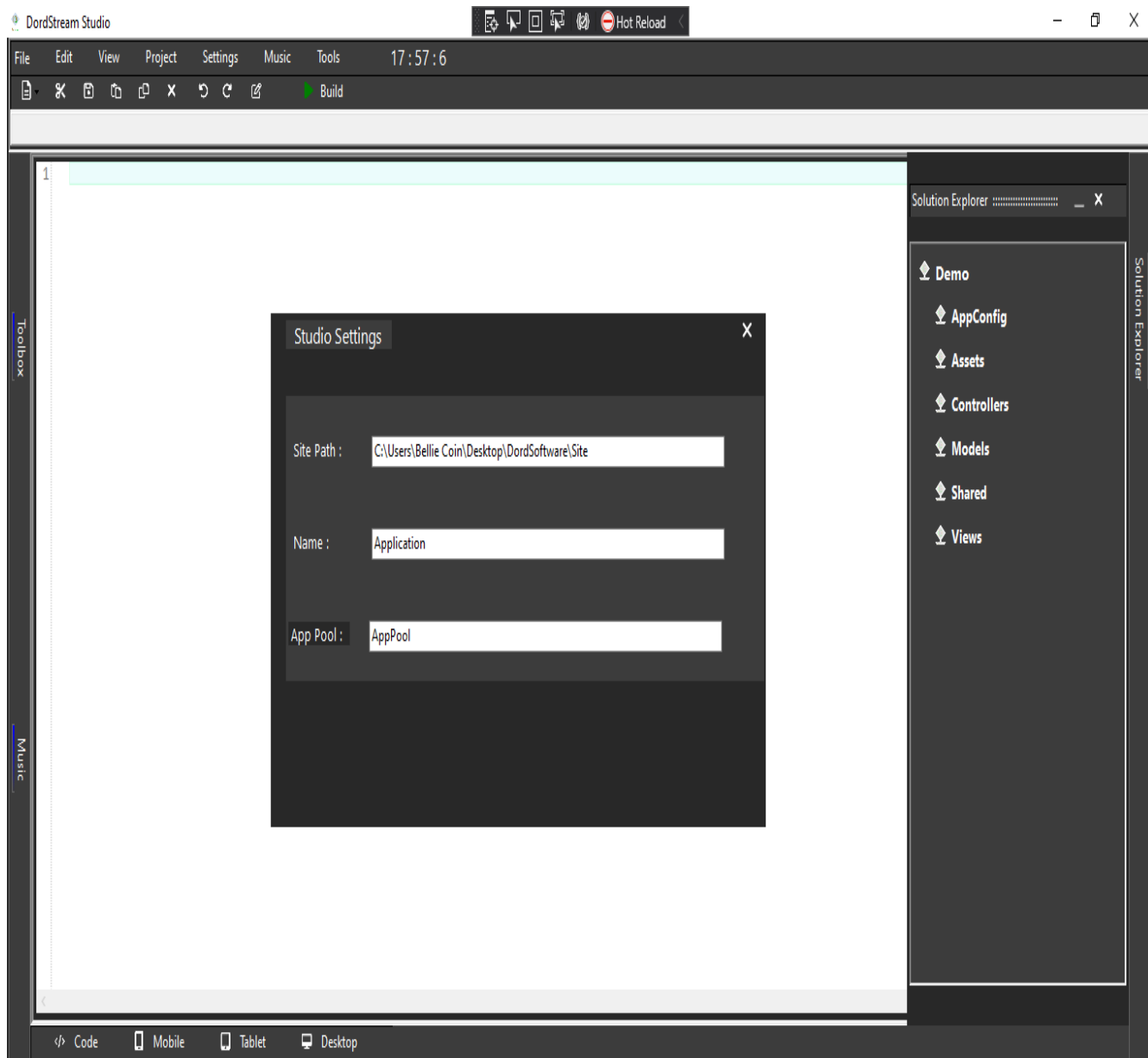
- **Website Template:** This is a template with default files and folders which is mostly used by a designer for developing a website. Before using this template, the programming language needed are Html, CSS and Dshtml. the examples of the website that can be developed using this template are blog, company website, presentation website, etc.



- **Note:** The project name must not contain any whitespace (spacing), special character (e.g. @\$&*) and digit (number)
- Click on **Create** to create the project

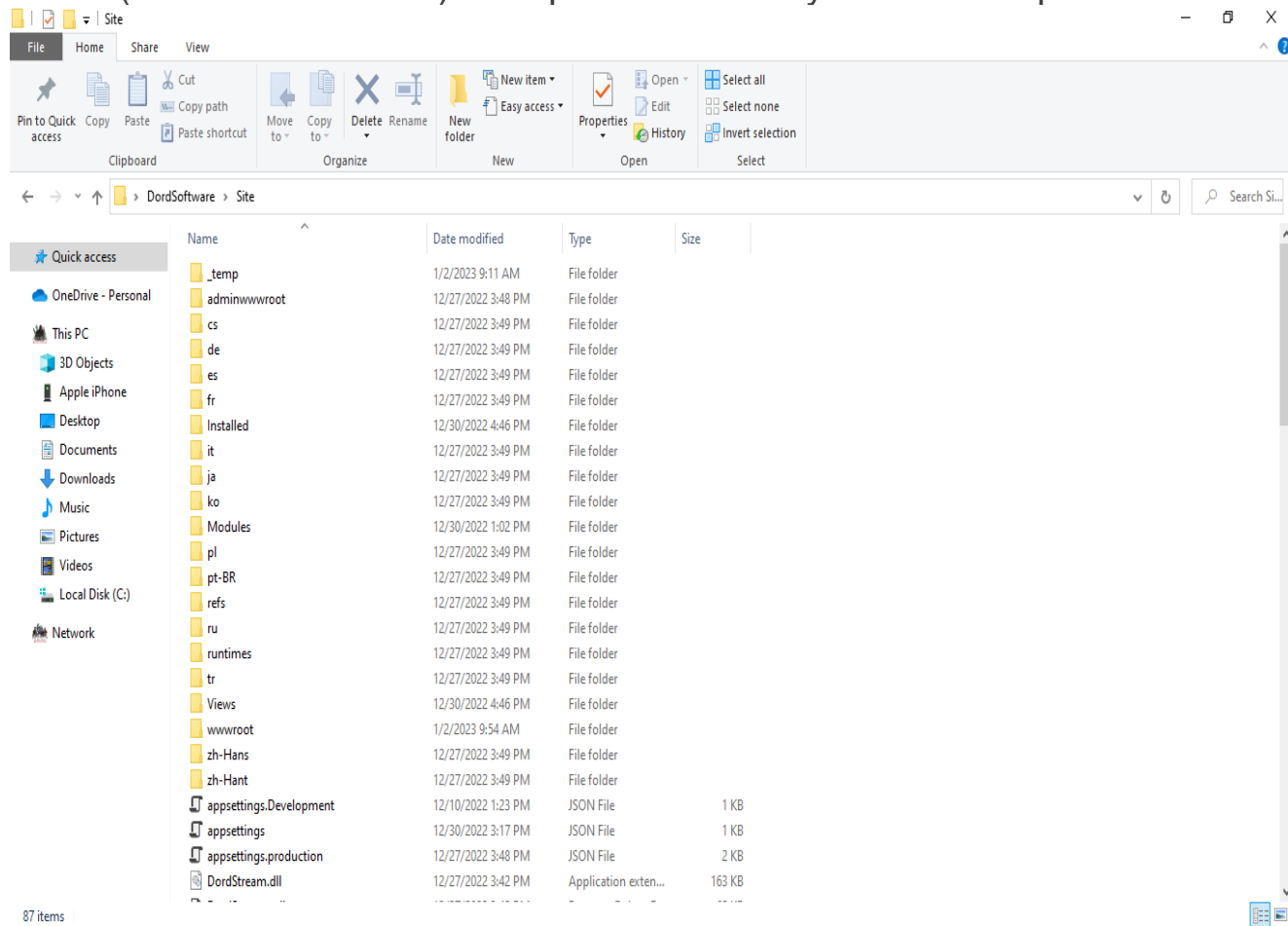


- Connect DordStream Studio with the DordStream CMS hosted on Internet Information Service (IIS). From the DordStream Studio **Settings** menu

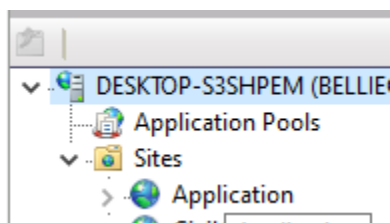


- **Site Path** specifies the path to the folder that contain the CMS files and folders (DordStream CMS).for example, My DordStream CMS was located at Desktop > DordSoftware > Site. the **Site** consist of my DordStream CMS file and Folder. **double click** on the textbox to select the path to the

folder (DordStream CMS) then press Enter Key to save the path.
















- **Name** specifies the site name used when creating a site (DordStream CMS) on Internet information services (IIS). After inputting the name then press Enter Key to save the site name. In this tutorial I named mine as **Application**.

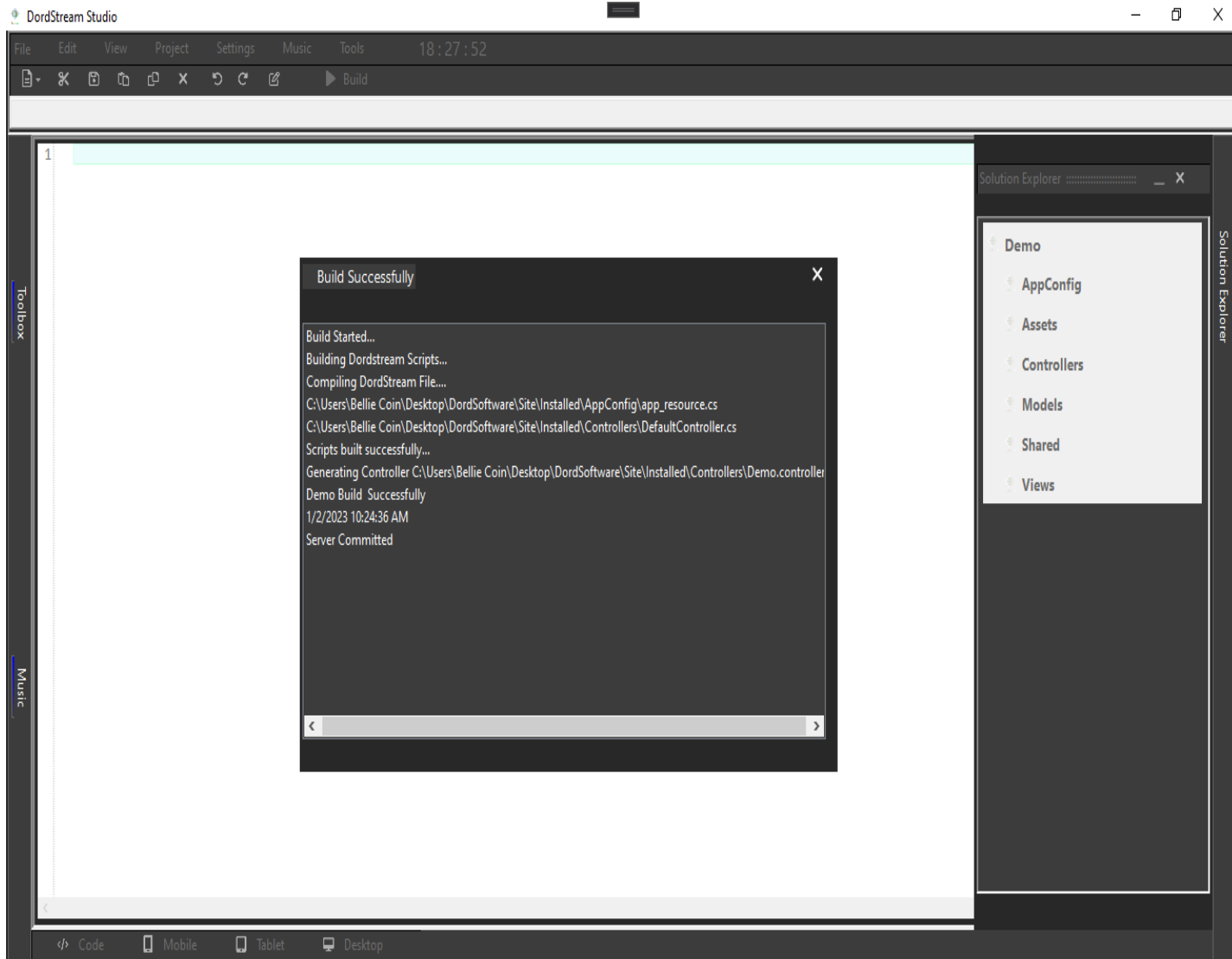


- **App Pool** specifies the application pool name used when creating a site (DordStream) on Internet information services (IIS). After inputting the application pool name then press Enter

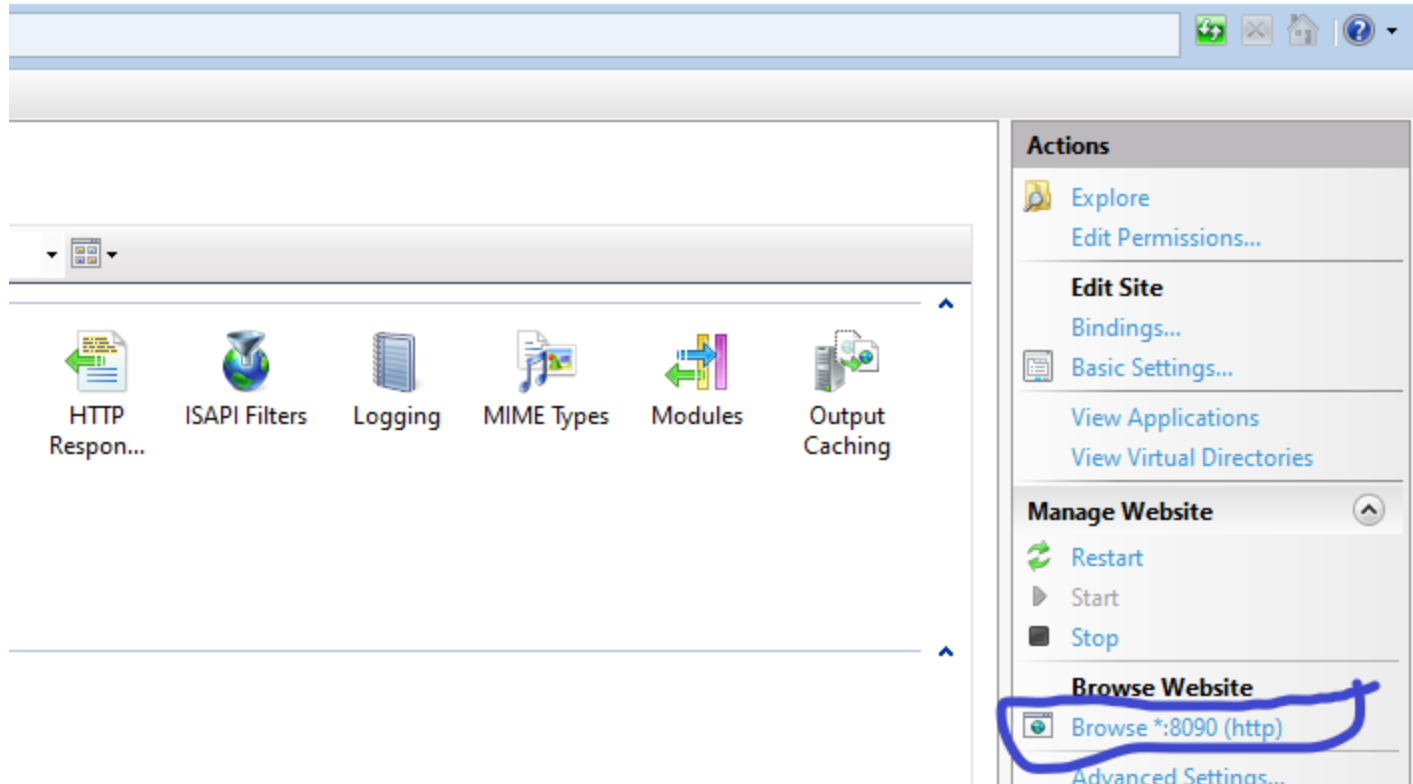
Key to save the application pool. In this tutorial is named mine as **AppPool**. **Note:** your application pool can be any name used when creating your site using DordStream CMS software on IIS.

Filter:	Go	Show All	Group by:	No Grouping	
Name	Status	.NET CLR V...	Managed Pipel...	Identity	Application
 .NET v2.0	Started	v2.0	Integrated	ApplicationPoolId...	0
 .NET v2.0 Classic	Started	v2.0	Classic	ApplicationPoolId...	0
 .NET v4.5	Started	v4.0	Integrated	ApplicationPoolId...	0
 .NET v4.5 Classic	Started	v4.0	Classic	ApplicationPoolId...	0
 AppPool	Started	No Manag...	Integrated	ApplicationPoolId...	2
 Beauty	Started	v4.0	Integrated	ApplicationPoolId...	0
 BeautyPalor	Started	v4.0	Integrated	ApplicationPoolId...	0
 BeautyParlor	Started	v4.0	Integrated	ApplicationPoolId...	0
 BeautyParlor Ap...	Started	No Manag...	Integrated	ApplicationPoolId...	0
 BloggerDotCore	Started	v4.0	Integrated	ApplicationPoolId...	0
 BloogerTest	Started	v4.0	Integrated	ApplicationPoolId...	0
 Classic .NET Ap...	Started	v2.0	Classic	ApplicationPoolId...	0
 Custom	Started	No Manag...	Integrated	ApplicationPoolId...	1

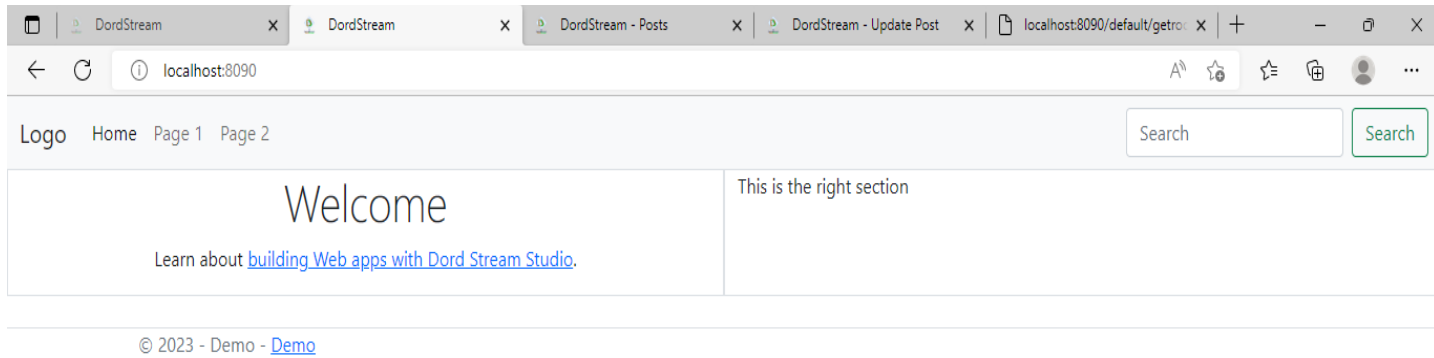
- Build your Project, Click on **Build** to complain the project.



- Display the **project** create on a web browser. open your web browser (either google chrome, Edge, opera, etc.), at the address bar (Tab bar) type the address to the hosted site on IIS. The address bar shows localhost:port# and not something like example.com. That's because localhost is the standard hostname for your local computer. Localhost only serves web requests from the local computer. Or accessing the site directly from the IIS Console. Click on **Browse** in your IIS console to display the site on a web browser.



The default template creates **Demo**, **Home**, **Page 1** and **Page 2** links and pages. Depending on the size of your browser window, you might need to click the navigation icon to show the links.



Project files and folders

The following table lists the files and folders in the project. For this tutorial

Folder	Purpose
AppConfig	<p>This is a folder that contain the project configuration and resources files. the folder consist of two files which are appsettings.json and app_resource.cs</p> <p>appsettings.json is a Json file that contain configuration settings and also use to store data. the data can be access using ConfigurationManager class.</p> <p>By default, appsetting.json consist of '"DebugMode": true' which specify configuration mode of the Application.</p> <p>Note: DebugMode must be set to false when the application is going for production. i.e. going on a web server.</p> <pre>ConfigurationManager manager = new ConfigurationManager(); object value = manager["Key Name"];</pre> <p>app_resource.cs is a c# file that consist of the libraries used for the application.</p>
Assets	<p>This is a folder that contains set of files generally text content, graphics, image, audio, video, Javascript, Css, etc. which server does not change when sending to users. this is the assets folder of the application.</p>
Shared	<p>This is a folder that contains the files which will be shared throughout the application.</p> <p>Shared folder has a required (compulsory) files and optional files. compulsory files are layout.html, landing.html, readme.txt, banner.png banner.jpg and error.html</p> <p>optional files depend on usage. they are login.html, register.html, post.html, category.html and page.html depending on the project template</p> <ul style="list-style-type: none"> Layout (layout.html). This is a file that contain the hypertext markup language which defines the project layout. Landing (landing.html). This is a file that contain the hypertext markup language which defines the landing page interface (first page to be loaded when running the application).

Folder	Purpose
AppConfig	<p>This is a folder that contain the project configuration and resources files. the folder consist of two files which are appsettings.json and app_resource.cs</p> <p>appsettings.json is a Json file that contain configuration settings and also use to store data. the data can be access using ConfigurationManager class.</p> <p>By default, appsetting.json consist of '"DebugMode": true' which specify configuration mode of the Application.</p> <p>Note: DebugMode must be set to false when the application is going for production. i.e. going on a web server.</p> <pre>ConfigurationManager manager = new ConfigurationManager(); object value = manager["Key Name"];</pre> <p>app_resource.cs is a c# file that consist of the libraries used for the application.</p>
	<ul style="list-style-type: none"> • ReadMe (readme.txt). This is a file that contain project description. • Error (error.html). This is a file that contain hypertext markup language which defines the error page interface. • Login (login.html). This is a file that contain hypertext markup language which defines the login page interface. • Register (register.html). This is a file that contain hypertext markup language which defines the register page interface. • Banner (banner.png banner.jpg). This is the project banner. (image file).

Folder	Purpose
AppConfig	<p>This is a folder that contain the project configuration and resources files. the folder consist of two files which are appsettings.json and app_resource.cs</p> <p>appsettings.json is a Json file that contain configuration settings and also use to store data. the data can be access using ConfigurationManager class.</p> <p>By default, appsetting.json consist of '"DebugMode": true' which specify configuration mode of the Application.</p> <p>Note: DebugMode must be set to false when the application is going for production. i.e. going on a web server.</p> <pre>ConfigurationManager manager = new ConfigurationManager(); object value = manager["Key Name"];</pre> <p>app_resource.cs is a c# file that consist of the libraries used for the application.</p>
	<ul style="list-style-type: none"> • Category (category.html). This is a file that contain hypertext markup language which defines the blog post category page interface. • Post (post.html). This is a file that contain hypertext markup language which defines the post page interface (display post). • Page (page.html). This is a file that contain hypertext markup language which defines the blog page interface (display blog page).
Views	<p>This is a folder that contains html file that is visible to the user. this is the visible part which user can access through web browser. the files are connected to the controller which allow users to interact with the controller.</p> <p>The files in views folder always depends on the controller.</p> <p>e.g. in DordStream Studio when you create a Controller file, the view will be generated automatically and create a connection between them.</p>

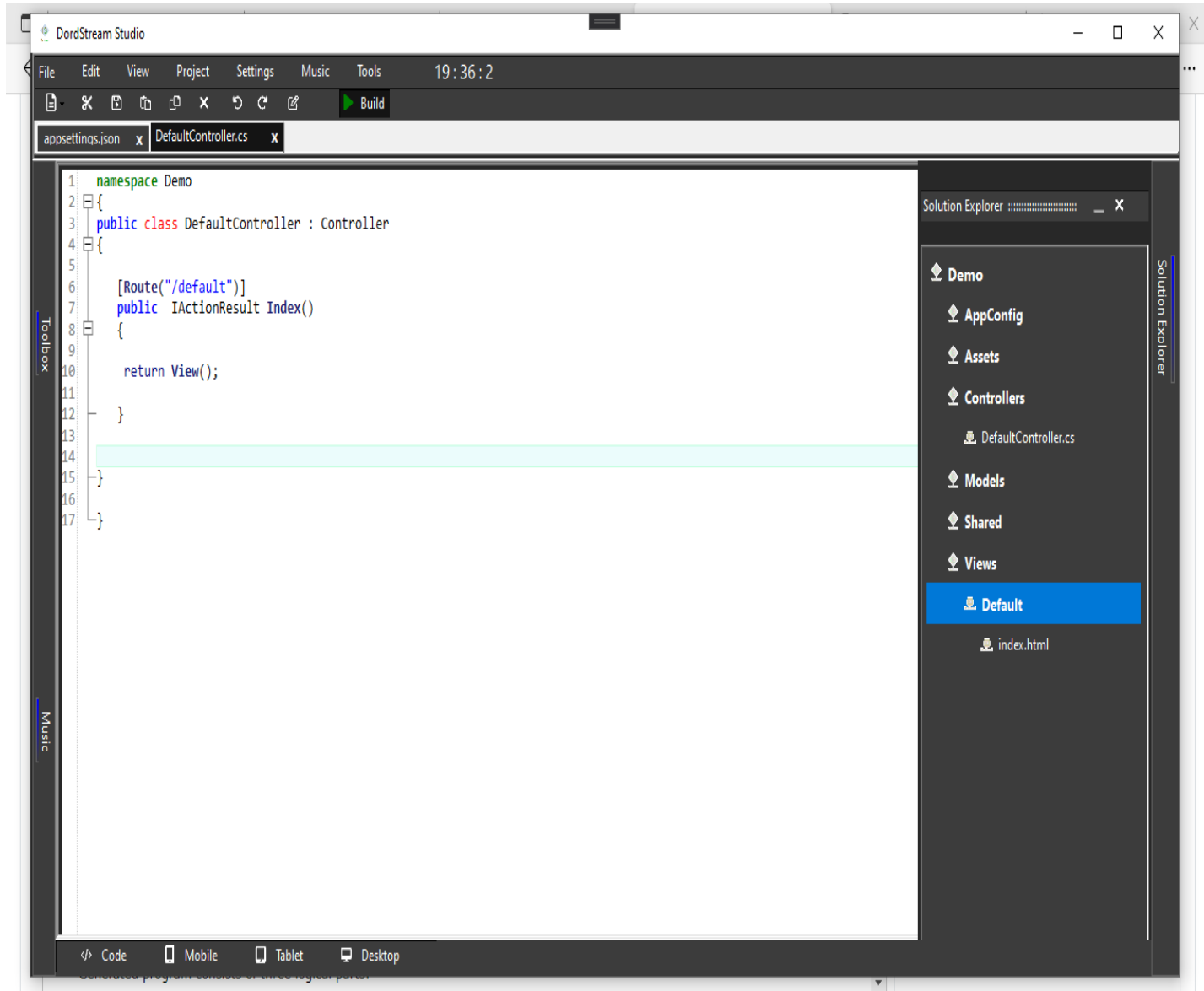
Folder	Purpose
AppConfig	<p>This is a folder that contain the project configuration and resources files. the folder consist of two files which are appsettings.json and app_resource.cs</p> <p>appsettings.json is a Json file that contain configuration settings and also use to store data. the data can be access using ConfigurationManager class.</p> <p>By default, appsetting.json consist of '"DebugMode": true' which specify configuration mode of the Application.</p> <p>Note: DebugMode must be set to false when the application is going for production. i.e. going on a web server.</p> <pre>ConfigurationManager manager = new ConfigurationManager(); object value = manager["Key Name"];</pre> <p>app_resource.cs is a c# file that consist of the libraries used for the application.</p>
	@Model, or @ViewBag or @ViewData["Key"] is used for in the html document to transfer a data from the controller to the view
Controllers	<p>This is a folder that contain the Controller files. the controller file is a c# file that consist of the application/ business logic. the file extension is .cs E.g. DefaultController.cs.</p>
Models	<p>This is the folder that contain the data model files used for the application. the model files are c# file used for transferring data from view to controller vise versa.</p> <p>The file extension is .cs (C#). E.g. DataModel.cs</p>

Controller File

In this section, you understand what is a controller.

Controller file is a C# file that contain the application logic. i.e. the logic used for creating or developing the application. After creating an

Application Template Project on DordStream studio a Default Controller was created with the view.



Generated program consists of three logical parts:

- Definition of namespace **Demo**
- Definition of a class **DefaultController**;
- Definition of a method **Index**;

Defining a Class

On the first line of our program we define a namespace called **Demo**.the simplest definition of a **namespace** consists of the keyword **namespace**, followed by its name. in our case the name of the namespace is **Demo** which is used to organize the code elements inside the curly brackets **{}**.

Defining a Class

On the first line of our program we define a class called **DefaultController**. The simplest definition of a class consists of the keyword **class**, followed by its name. In our case the name of the class is **DefaultController**. The content of the class is located in a block of program lines, surrounded by curly brackets: **{}**.

```
namespace Demo
{
    public class DefaultController : Controller
    {

        [Route("/default")]
        public IActionResult Index()
        {

            return View();

        }

    }

}
```

the program above consists of single method named Index and return IActionResult.

Defining the Index() Method

we define a method with the name **Index()**, which is the starting point for our program. Every program written in DordStream. the method interact with index view which is inside the Default folder.

The method must be declared as shown above, the string or text that attribute Route take as parameter which specify the route to the view.

Contents of the Index() Method

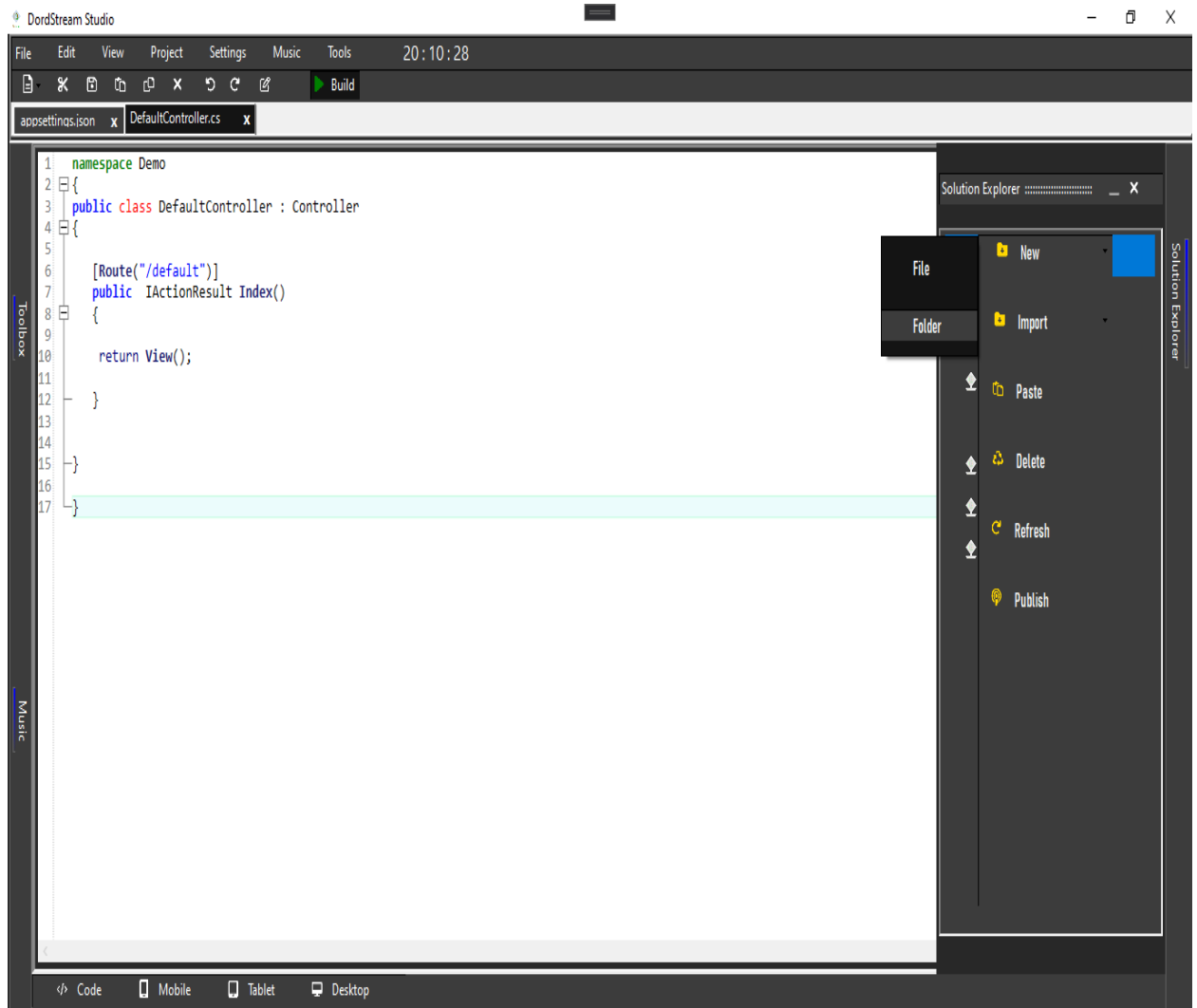
The content of every method is found after its signature, surrounded by opening and closing curly brackets. On the next line of our sample program we use *return view()* method to return the html file of the view.

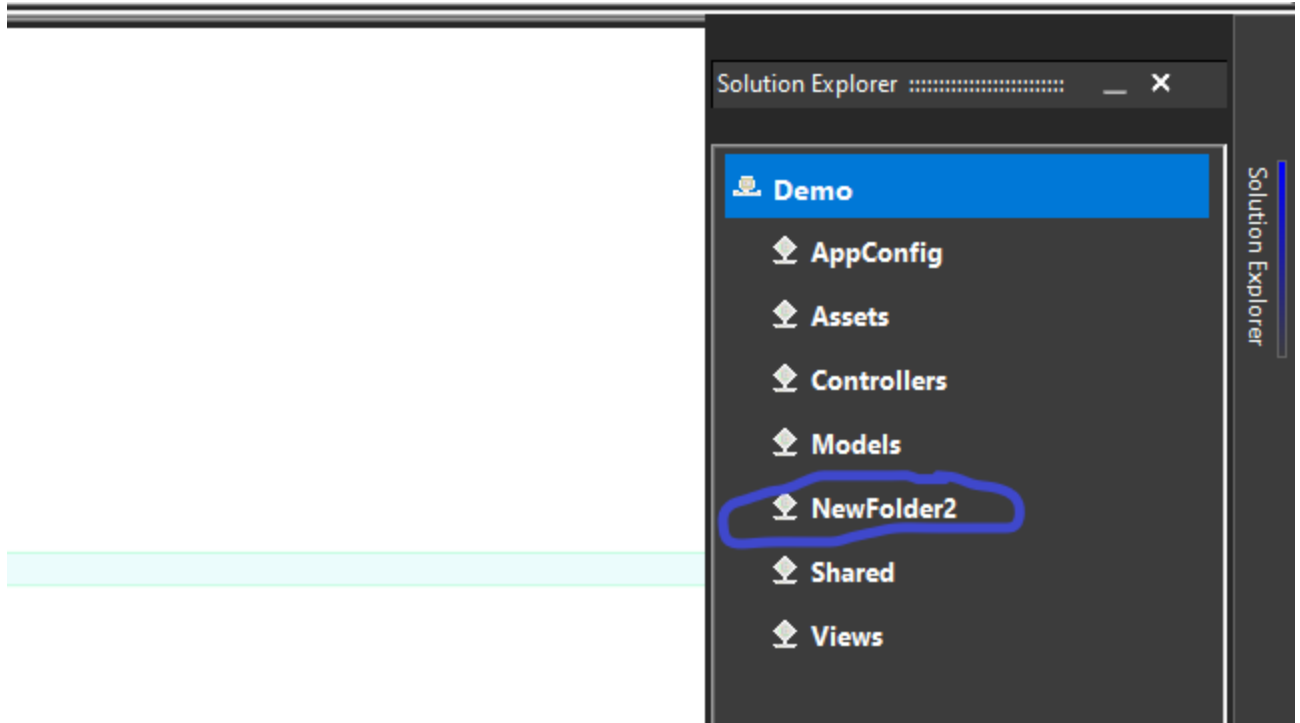
Click on Build to compile the project and debug it through browser.

Adding a Class

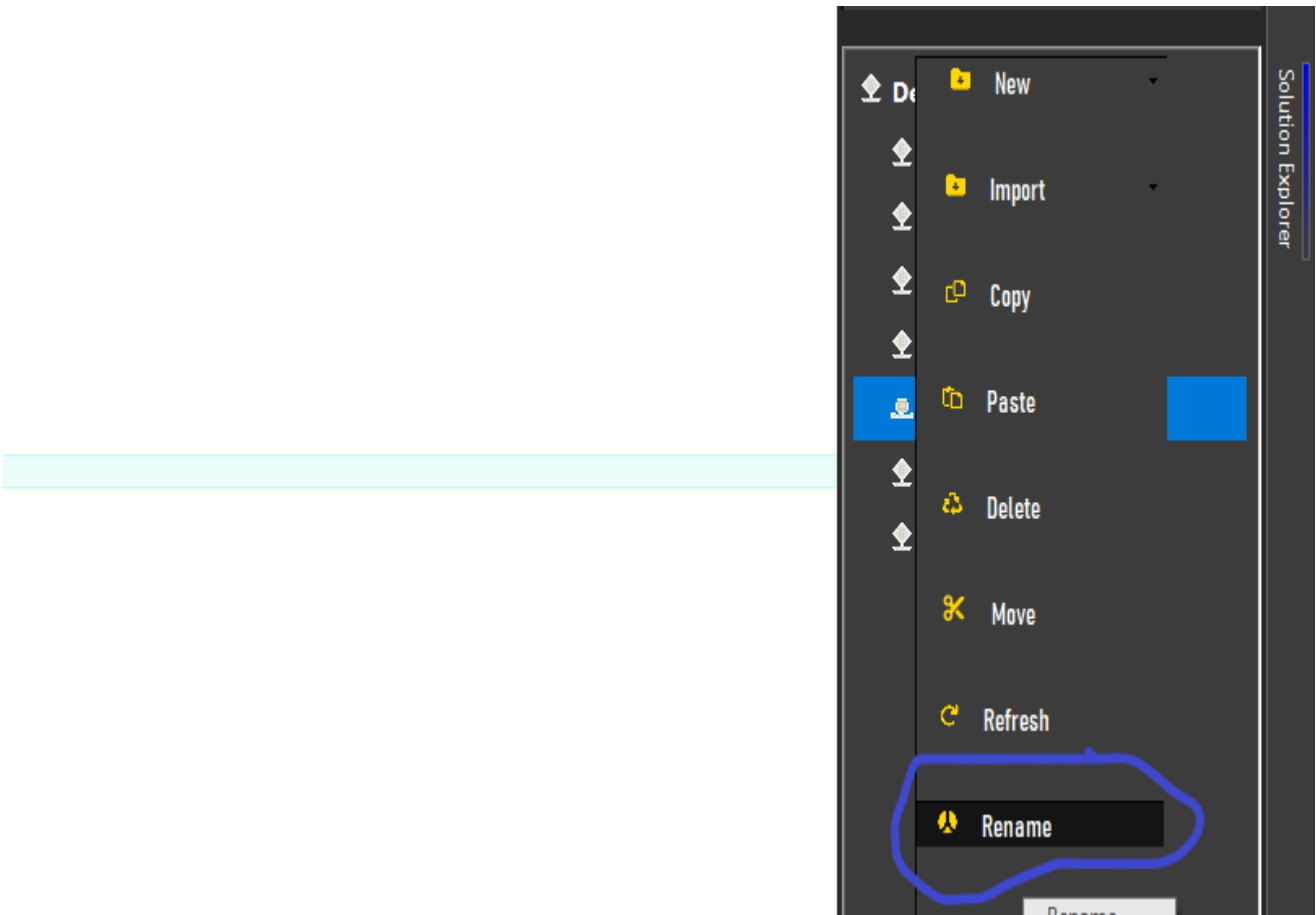
In this section, you add and reuse classes for managing the project

- Create a folder that will contain the class file
- **Right click** on the project name (in this tutorial, the project name is **Demo**) then click **New > Folder**

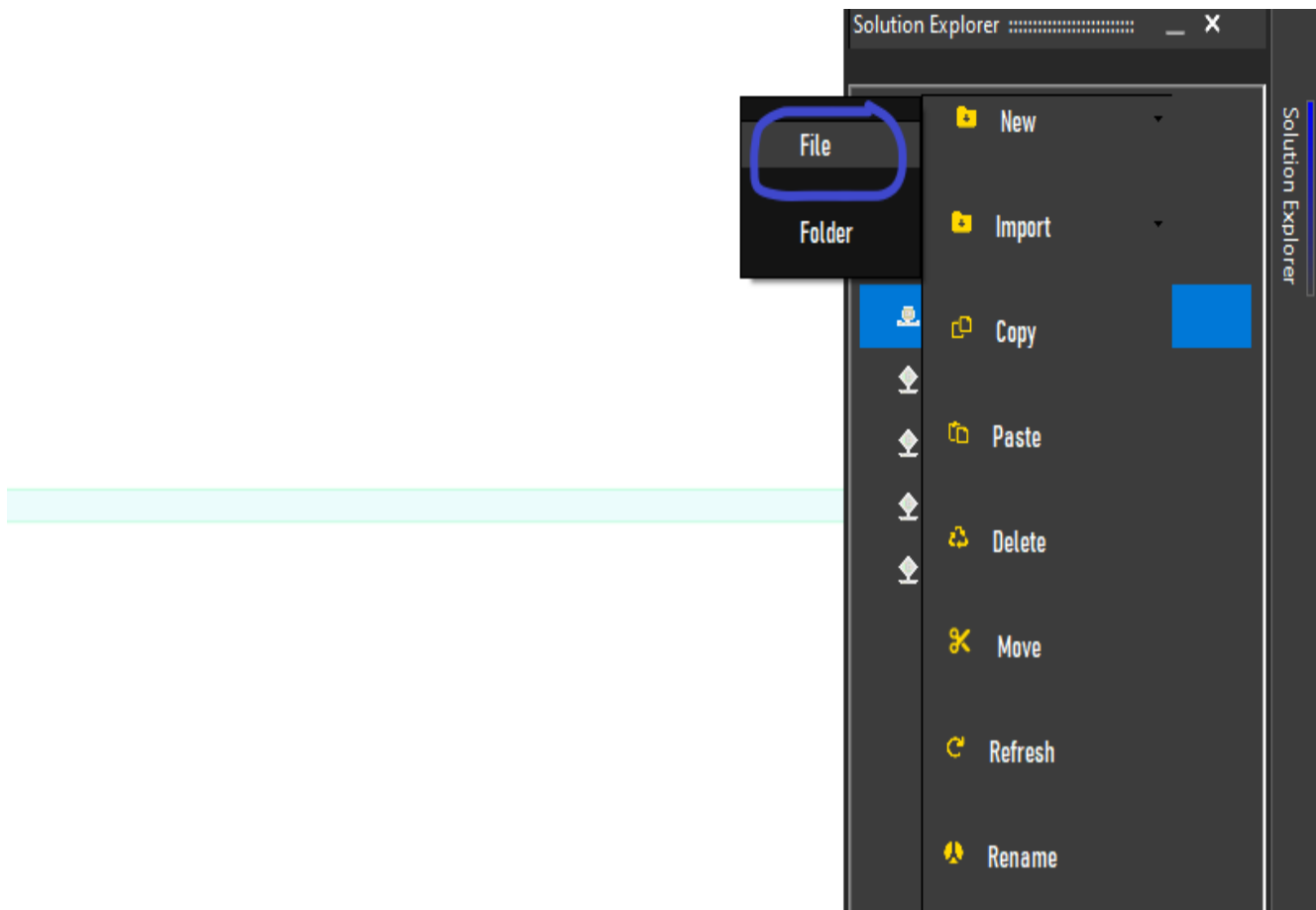




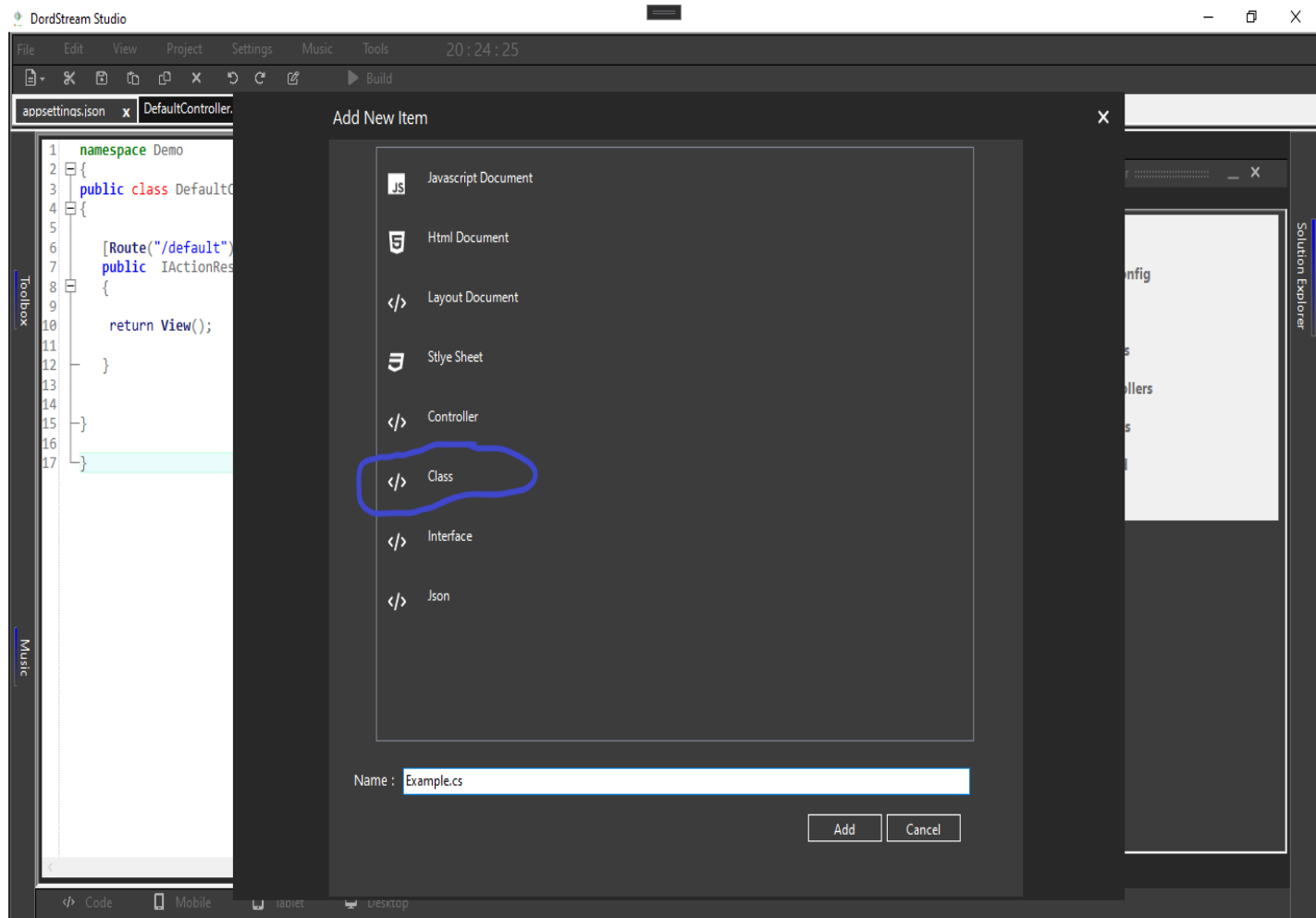
- A new folder with random number will be created by default. in this example **NewFolder2** was created
- Rename the folder to the desired name. In this tutorial we will rename the folder as **Classes**.
- **Right click** on the folder and click on **Rename**



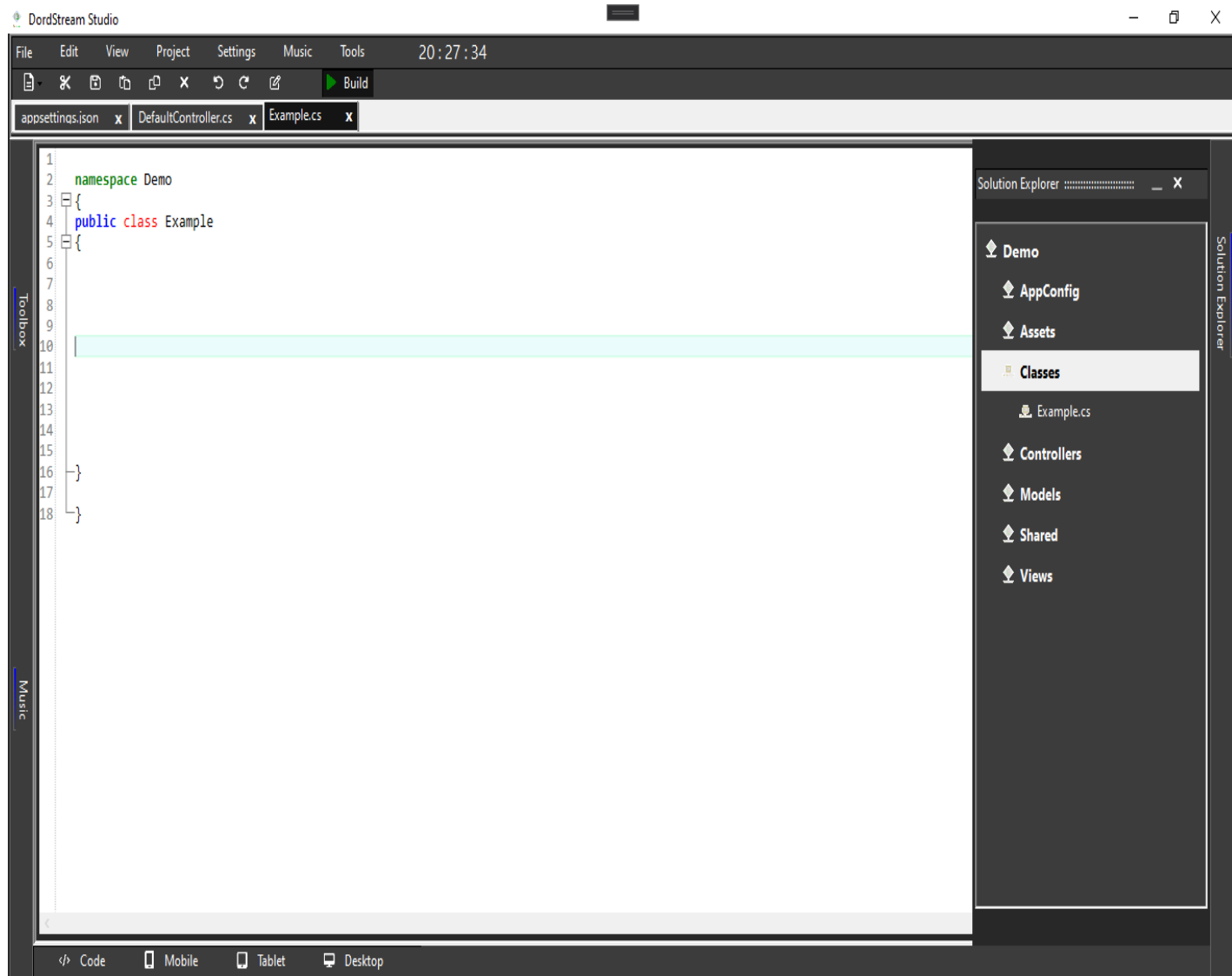
- Specify the name then press **Enter key**
- Create a class file into the folder created.
- Right click on the folder (Classes) then click **New > File**



- Click on class and give it a name. In this tutorial we are naming it **Example.cs**



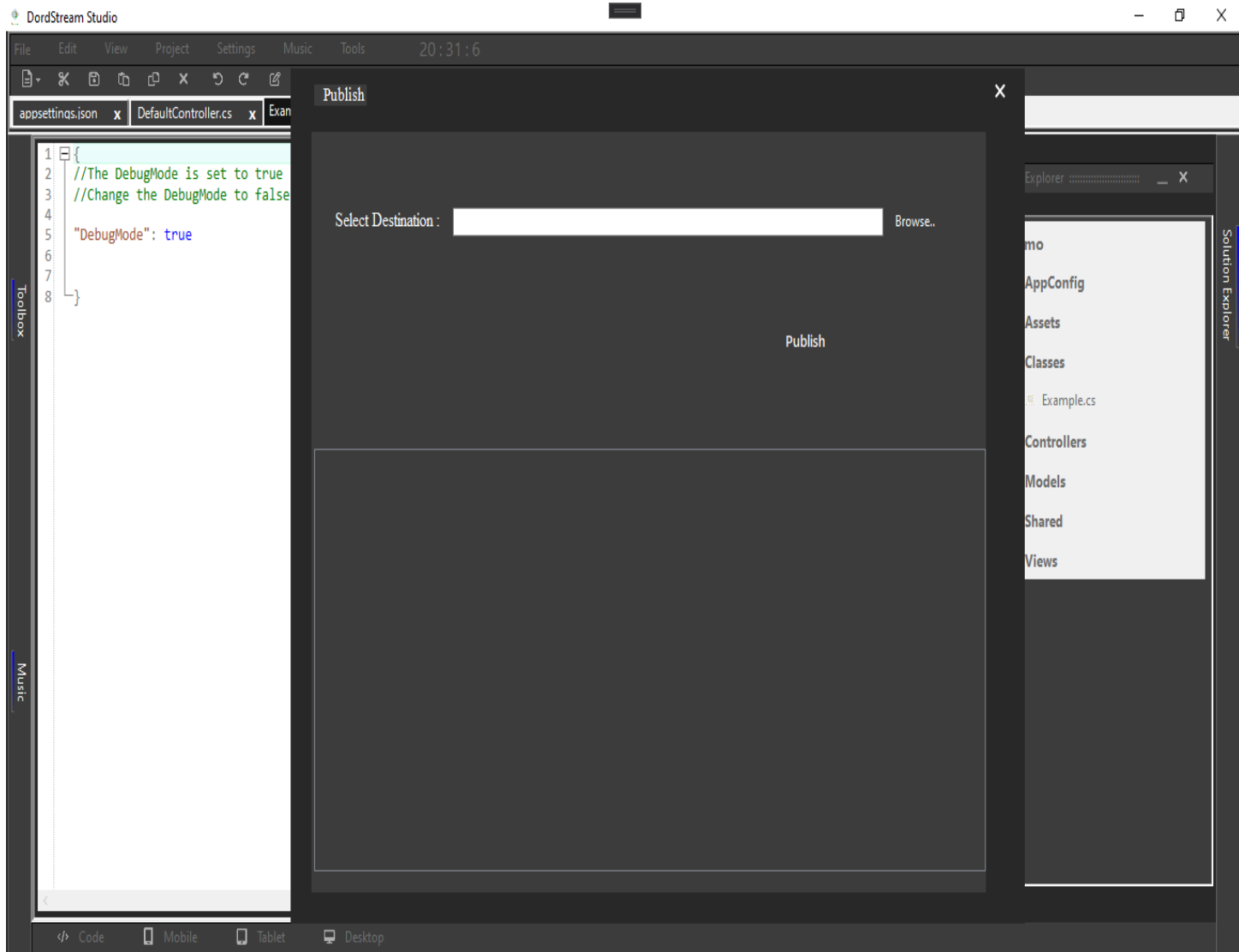
- Click on **Add** to create the Class.
- A default class will be generated. To Learn How to create a class in C#



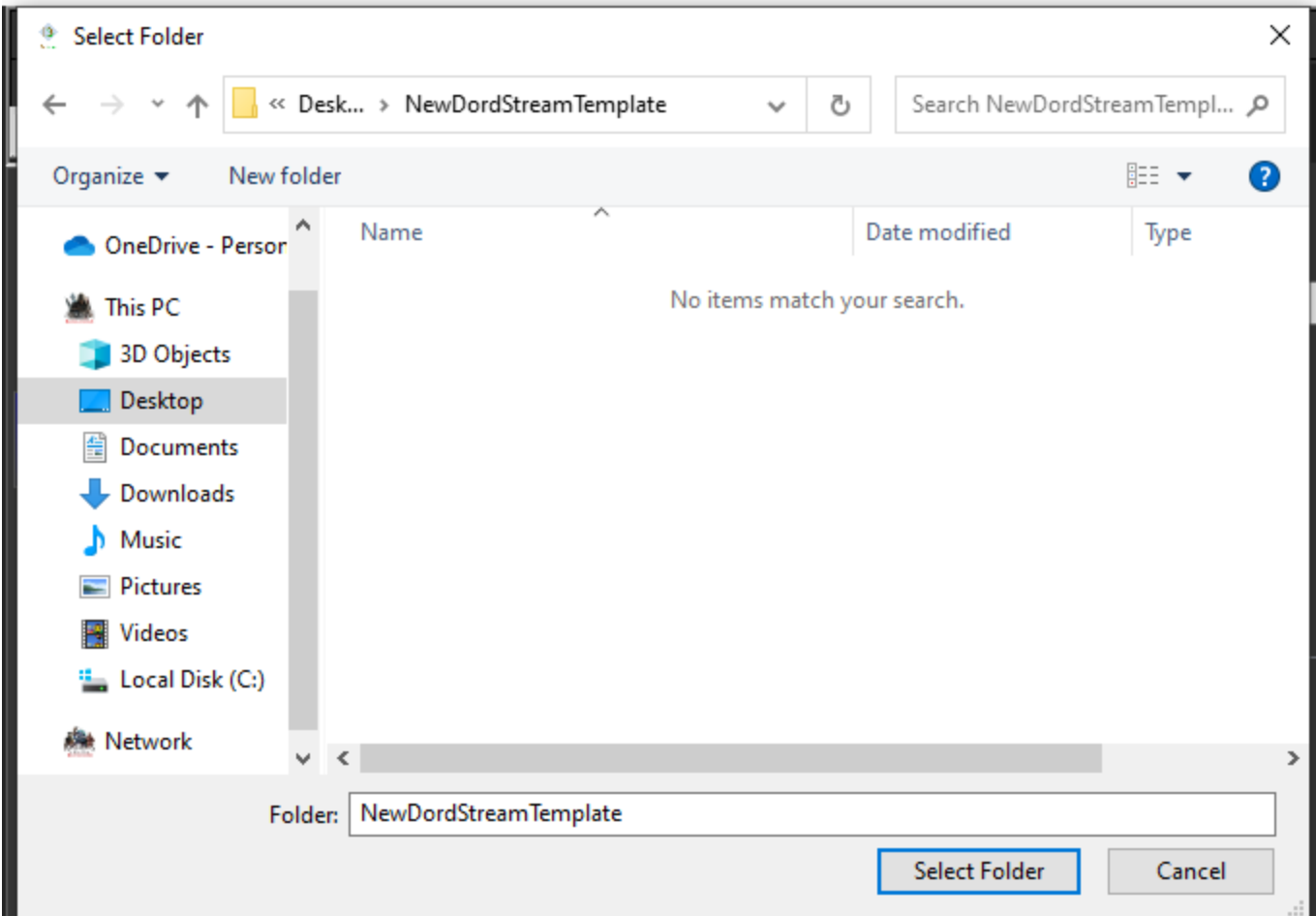
- The class can be in a controller by creating an instance of the class

Publish Project

This is the final section where your project is ready for production. (**Note:** change DebugMode to false in appsettings.json before publishing your project). From the DordStream Studio click **Project** menu.



- Click on **Browse** to select the project destination i.e. the location to save the generated file.



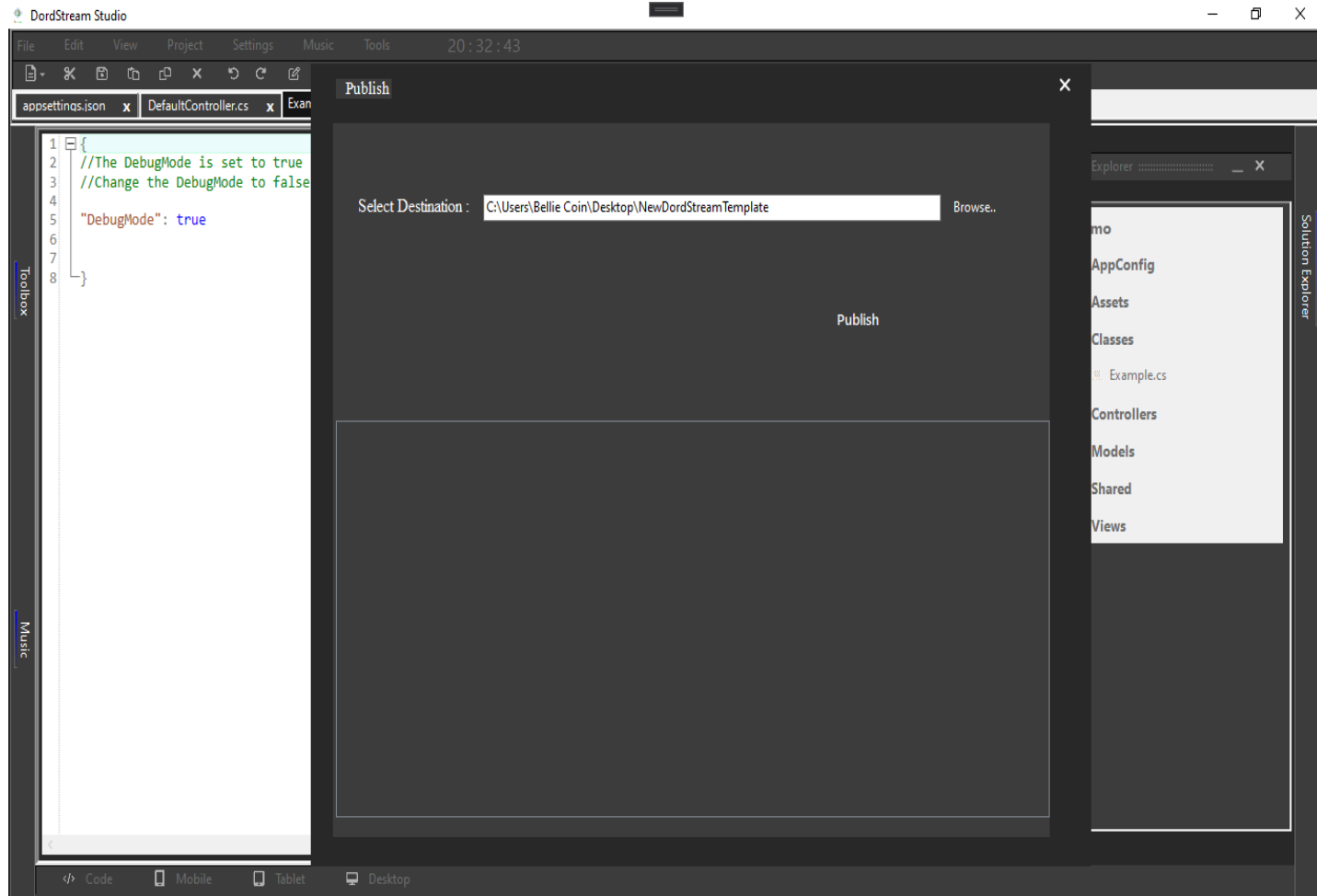
Music

</> Code

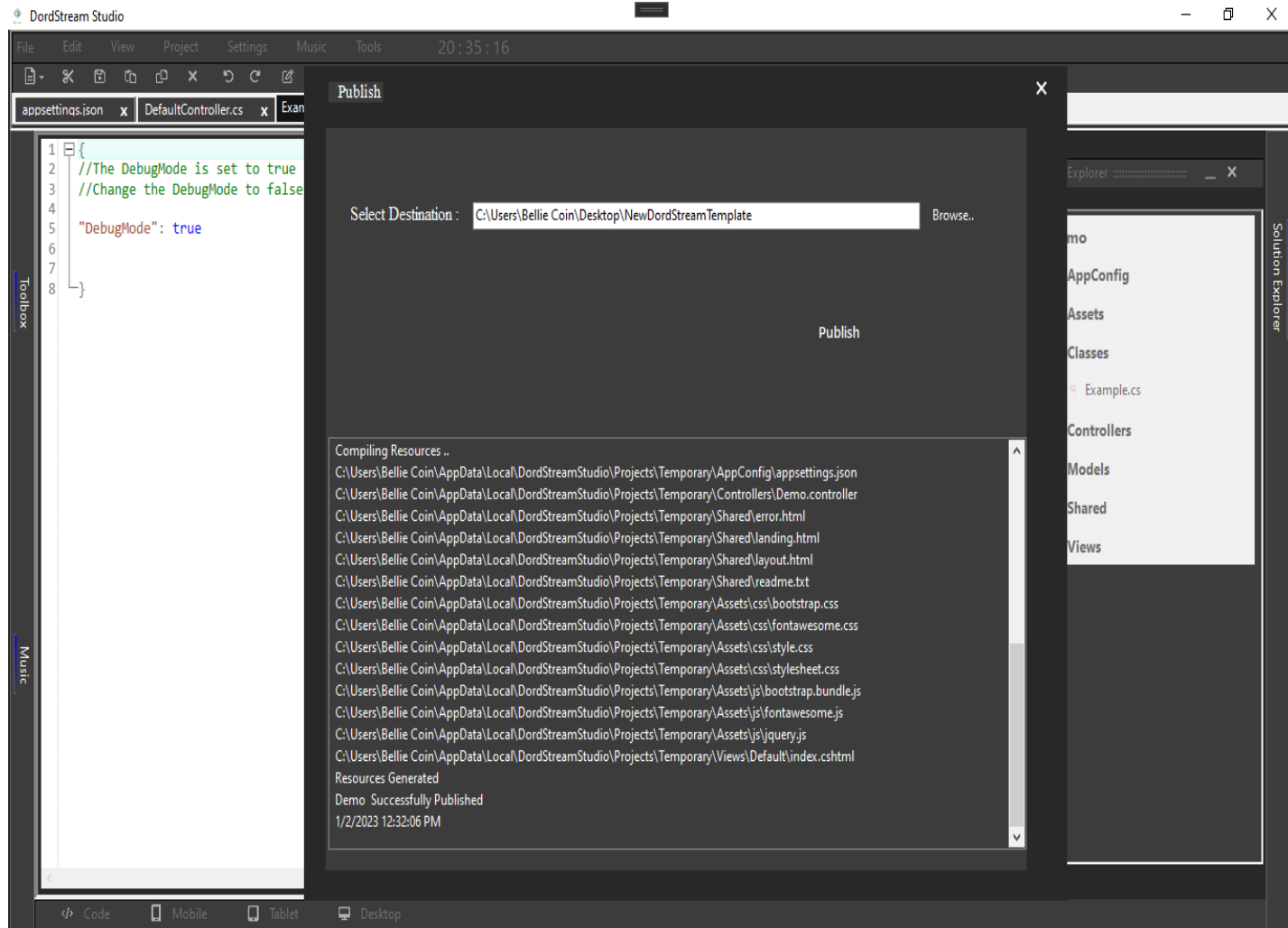
Mobile

Tablet

Desktop



- Click on **Publish** to compile and build the project i.e. ready for production.



- After publishing, then you can now upload it to the Themes section of the **CMS software**.