**How to use the template**

**Putting the project template where it belongs**

Shut down Visual Studio.

Drop **DPLSolutionTemplate.zip** in **C:\Users\{username}\Documents\Visual Studio {version}\Templates\ProjectTemplates**

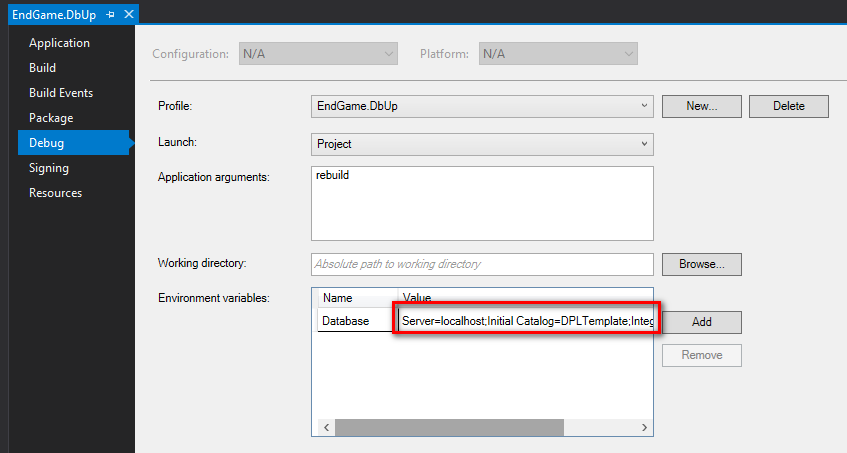
The template will show up in the new project menu in Visual Studio if you search for DPL.

**Creating a new project using the template**

Create your new project using the template. If you can't find it, search for **DPL Template Solution** in the New Project Window.

Then, you must do some modifications before it's ready to run.

Modify the connection string in the **{projectname}.DbUp** project environment variable so it contains your desired connection string:



For an example, here's mine:  
  
Server=localhost;Initial Catalog=DPLTemplate;Integrated Security=True;

Build your solution.

**Setting up the database and generating the context**

Create an empty database on whatever db platform you're using. This example uses SQL Server Developer edition with mixed mode authentication.

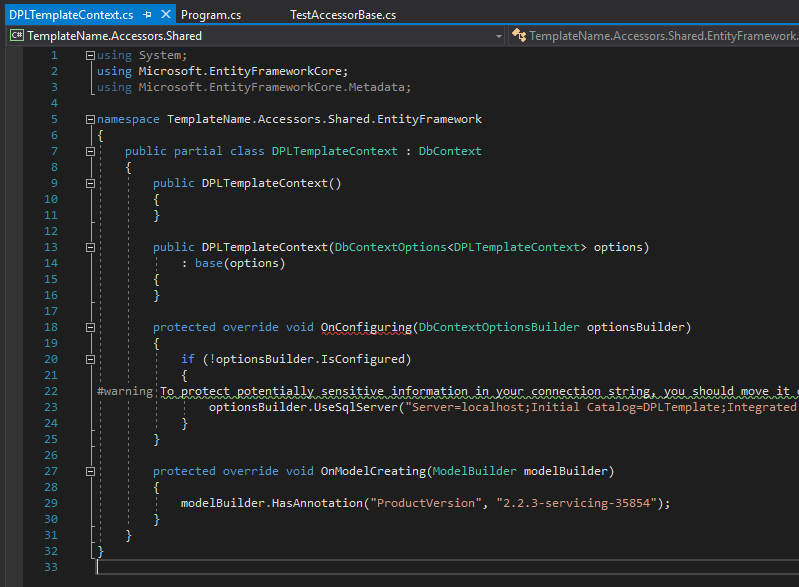
Open a command prompt and cd to your newly created solution subfolder where all the project folders live, and run this command:

**dotnet ef dbcontext scaffold --project {project.name}.Accessors.Shared --startup-project {project.name}.DbUp -o EntityFramework "Server=localhost;Database={database.name};Trusted\_Connection=True;" Microsoft.EntityFrameworkCore.SqlServer -f**

**Note:** when running this command, no news is good news. If you don't get any messages on the screen, then everything worked fine.

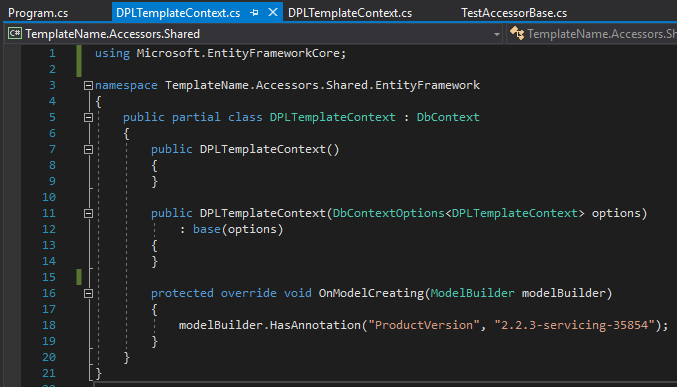
<rewrite process dealing with the context>

You will see a **DatabaseContext.cs** partial class in **{projectname}.Accessors.Shared/EntityFramework** that looks like this:



You'll have to clean that up slightly. It must be renamed to DatabaseContext to match the existing partial class, or the partial class must be renamed to match your new generated context.

Then, remove the **OnConfiguring** override. The template includes the code you need there in the other half of this partial class, currently called **DatabaseContext**, one folder up from the **EntityFramework/{databasename}Context** class you just generated. When you're done, it'll look like this:

At This point, the project will build, DbUp will run, and all tests will pass. You're ready to get coding!

**How to modify the template and export a new version**

**Updating the source code**

The **WorkingSolution** folder contains the working copy of the solution.

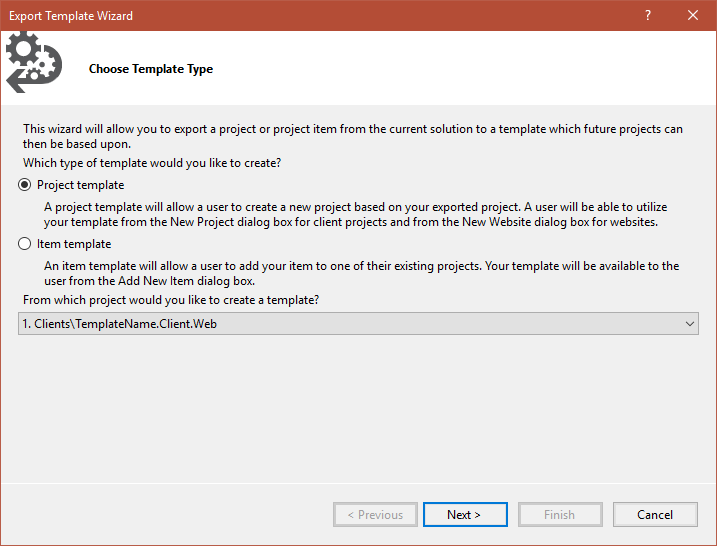
In order to modify this template and create a new version of the template, you must open that solution and follow the steps above for **Setting up the database and generating the context**.

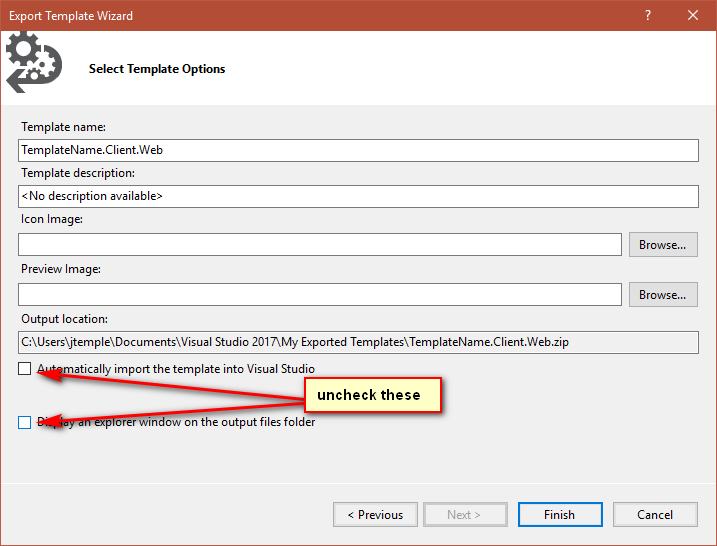
Once finished, it will build and run, and all tests will pass. This is what you modify in order to build a new version of the solution template.

Once you're done making your changes, you can delete the generated context class and any database models, so you can begin exporting a new template.

**Exporting a new version of the template**

Once finished, you must extract each project individually by going to the **Project -> Export Template** menu in visual studio. Use the following options to prevent Visual Studio from automatically importing each project into your new project menu:





This will extract each project into a template .zip file located in your **C:\Users\{username}\Documents\Visual Studio 2017\My Exported Templates** folder.

Delete all old project folders out of the **ExtractedTemplates** folder in this repo except the **ROOT.vstemplate** file.

Copy **TemplateName.sln** from the **WorkingSolution** folder into the **ExtractedTemplates** folder.

Move all the zip files from the **My Exported Templates** folder into the **ExtractedTemplates** repo folder and unzip them. Optionally delete the zip files.

**Altering the extracted solution to make the template work**

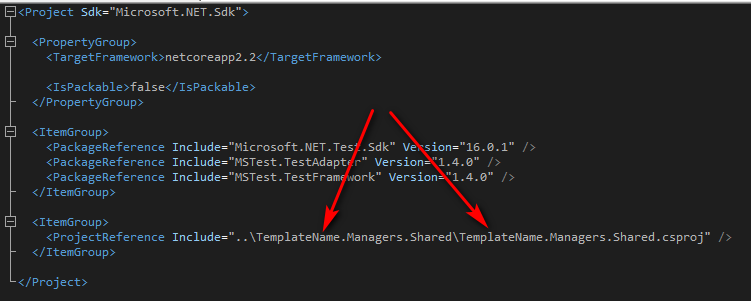
The **ExtractedTemplates** folder now contains the projects after extraction from Visual Studio and unzipping, a solution file, and a vstemplate file.

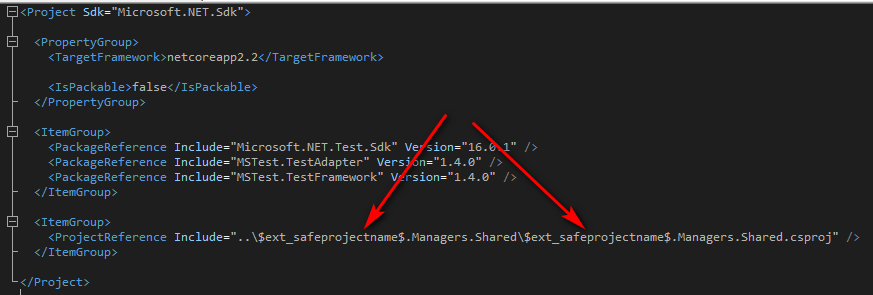
Open the **ExtractedTemplates\TemplateName.sln** in Visual Studio.

Do a find/replace in all \*.cs files in the **ExtractedTemplates\TemplateName.sln**, replacing **TemplateName** with **$ext\_projectname$**

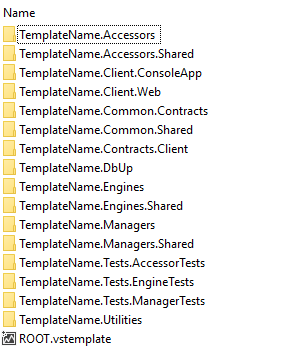
Update the **ROOT.vstemplate** file to match any projects you've added/removed as part of your new template build.

Update each .csproj file and replace every instance of TemplateName with $ext\_safeprojectname$. This is what preserves your project dependencies when the project is created.

**Before:**

**After:**

Now, create a new zip file in the **ExtractedTemplates** called **DPLSolutionTemplate.zip**. Add the unzipped project folders and their contents, and the **ROOT.vstemplate** file to it, so it looks like this inside:



**Note:** Do not include the solution file.