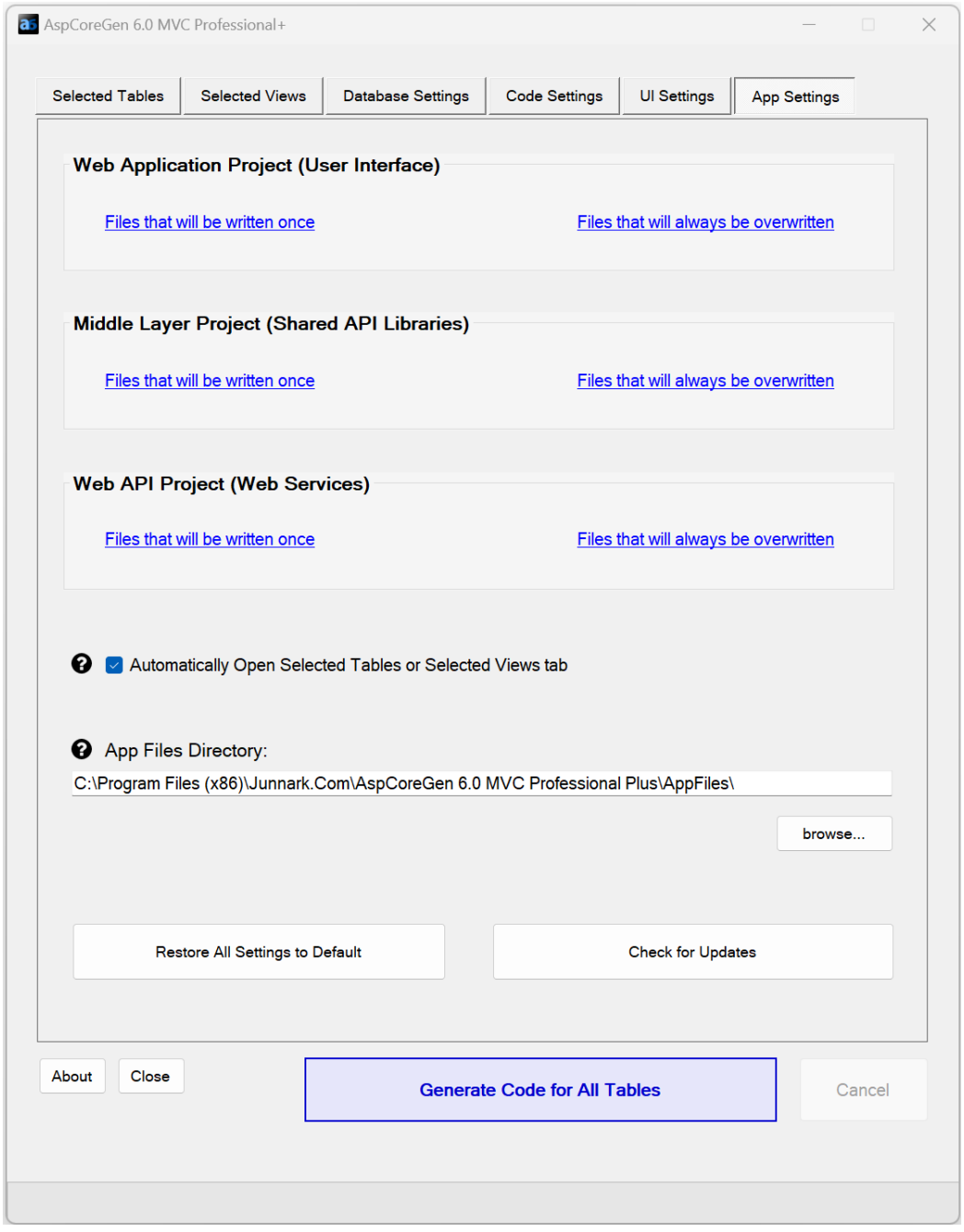
App Settings Tab

1 CONTENTS

1 Generated Projects 3
1.1 Web Application Project 4
1.1.1 Files That Will Be Written Once 5
1.1.2 Files That Will Always Be Overwritten 5
1.2 Middle Layer Project (Class Library) 6
1.2.1 Files That Will Be Written Once 6
1.2.2 Files That Will Always Be Overwritten 7
1.3 Web API Project 7
1.3.1 Files That Will Be Written Once 8
1.3.2 Files That Will Always Be Overwritten 8
1.4 Why is it Important to Know Which Files Will be Written Once or Always Overwritten 8
1.4.1 Files That Will Be Written Once 8
1.4.2 Files That Will Always Be Overwritten 9
2 Other Settings 9
2.1 Automatically Open Selected Tables or Selected Views Tab 9
2.2 App Files Directory 10
2.3 Restore All Settings to Default (Button) 10
2.4 Check for Updates (Button) 10

App Settings Tab

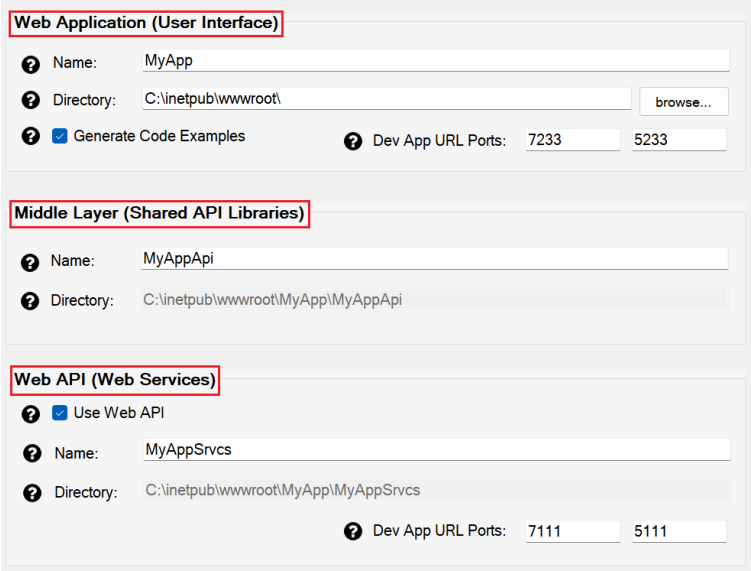
The App Settings Tab is where we set AspCoreGen 6.0 MVC application-specific information such as the directory where the *App Files* are located. Whether to open the *Selected Tables* or *Selected Views* tab when the respective option is chosen in the *Code Settings* tab. You can also *Restore All Settings to Default* from here with a click of a button. And most importantly, it shows in a list the generated *Files* that you can customize per Project and the generated *Files* that you should not add code to.



1 GENERATED PROJECTS

These are the *Generated Projects* as shown in the *Code Settings Tab*.

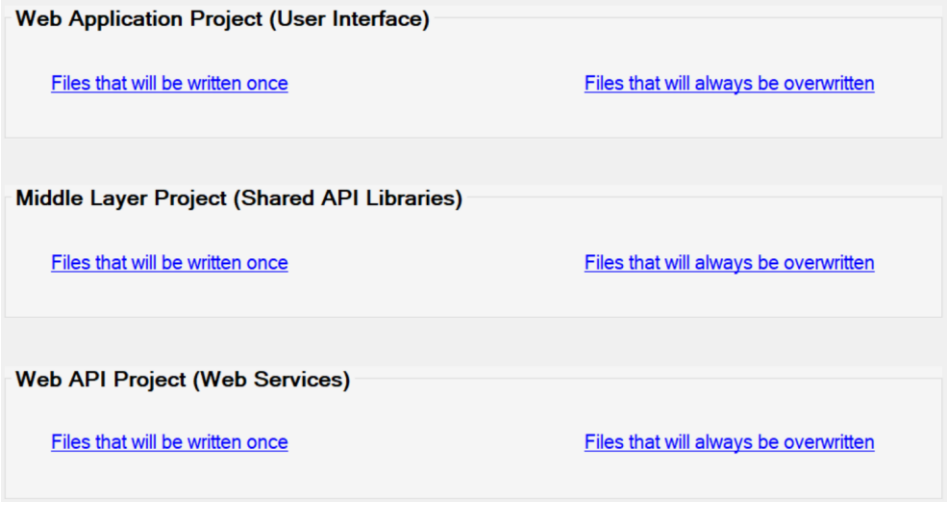
- Web Application Project (User Interface/Front End)
- Middle Layer Project (Class Library Project – Business Layer, Data Repository, and Shared Libraries)
- Web API Project (Web Services - Optional)



Code Settings Tab – Generated Projects

Each application/project are comprised of generated *Files*. You can add your own code in some of the generated *Files* per project, and when you generate code over the same projects again these *Files* will NOT be overwritten. There are also *Files* that you should not add your own code to.

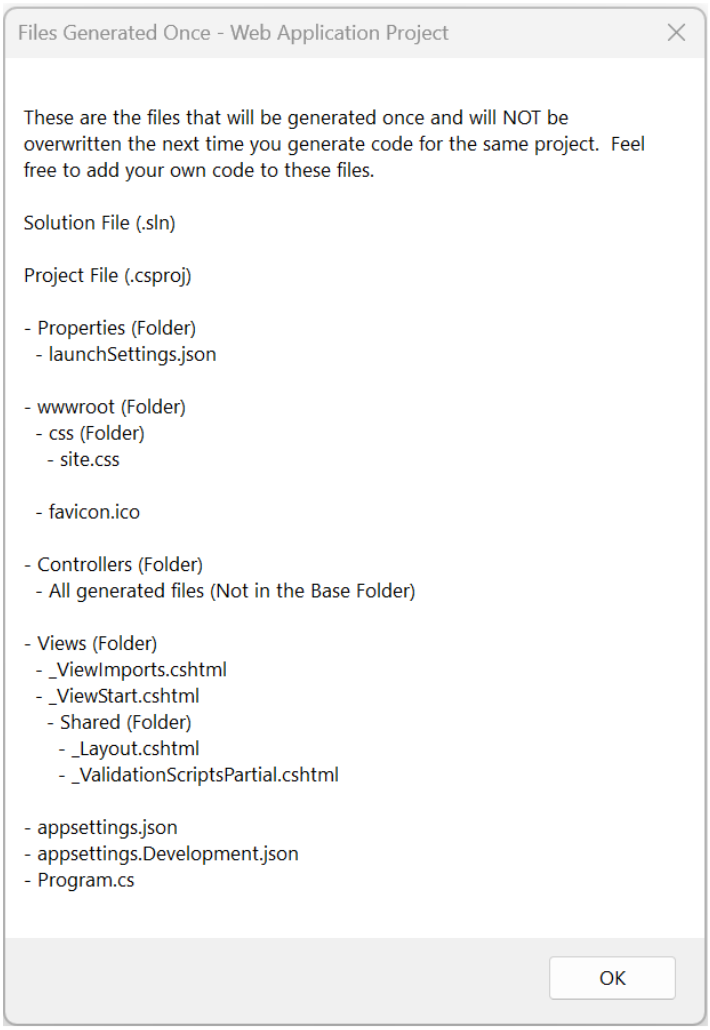
- **Files that will be written once:** You can add your own custom here.
- **Files that will always be overwritten:** Do not add your code here, it will be overwritten.

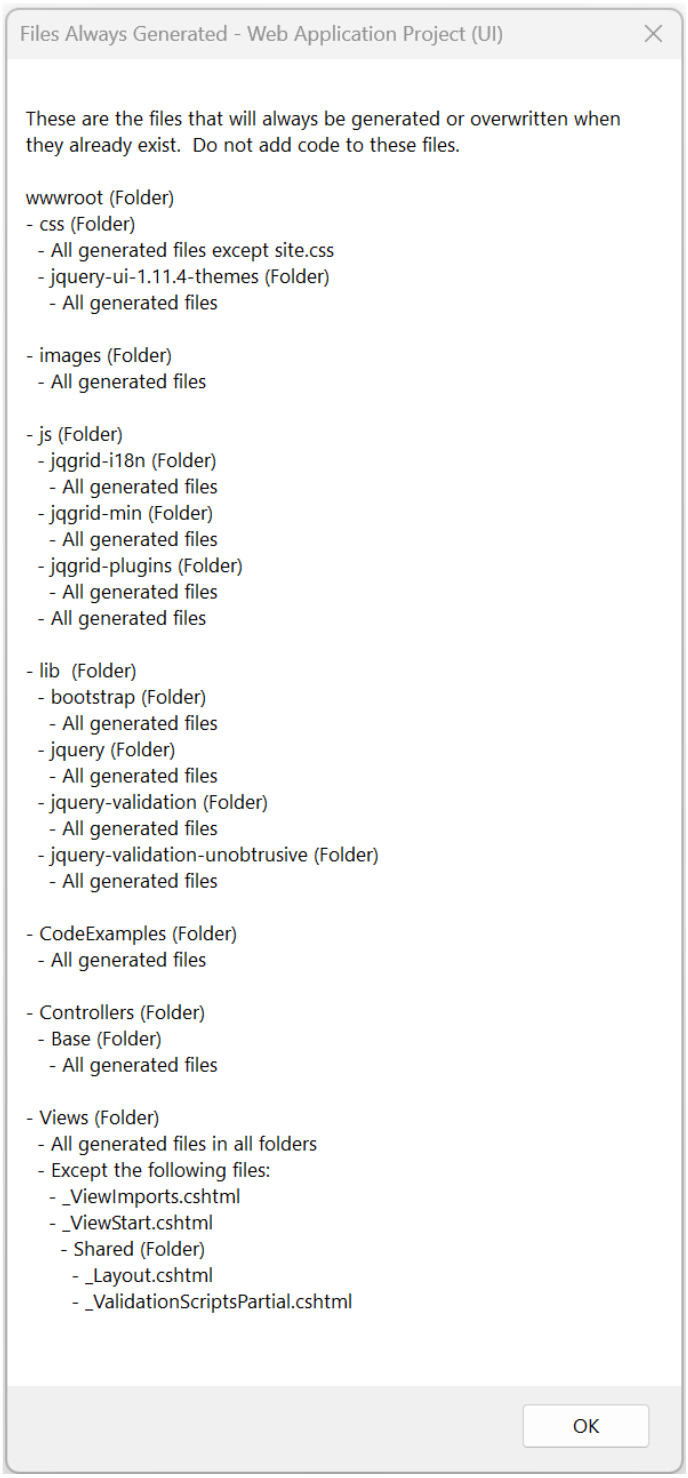


App Settings Tab – Files Written Once, Files Always Overwritten

1.1 WEB APPLICATION PROJECT

There are 2 links under the *Web Application Project*. A box pops-up when you click the respective link.





1.1.1 Files That Will Be Written Once

Only generated the first time you generate code for the same project, or when you delete it. Here are some examples.

- **Solution File** – You can add new projects into it.
- **Project File** – You can add new items in the project.
- **launchSettings.json** – You can add new application launch settings or update current settings.
- **site.css** – You can add new styles or update current styles.
- **favicon.ico** – You can update it with your own favicon.ico.
- **Controllers (Not in Base Folder)** – You can add new Action methods or just methods, or remove existing methods in any partial class under *Controllers* folder. E.g. If you add a new *View* in the *Category* folder, then you could add the respective *IActionResult* method in the *CategoryController* partial class.
- **_Layout.cshtml** – You can update the layout/theme of the generated web app.
- **appSettings.json** – You can add new application settings or update existing settings.
- **Program.cs** – You can inject more objects to independent classes.

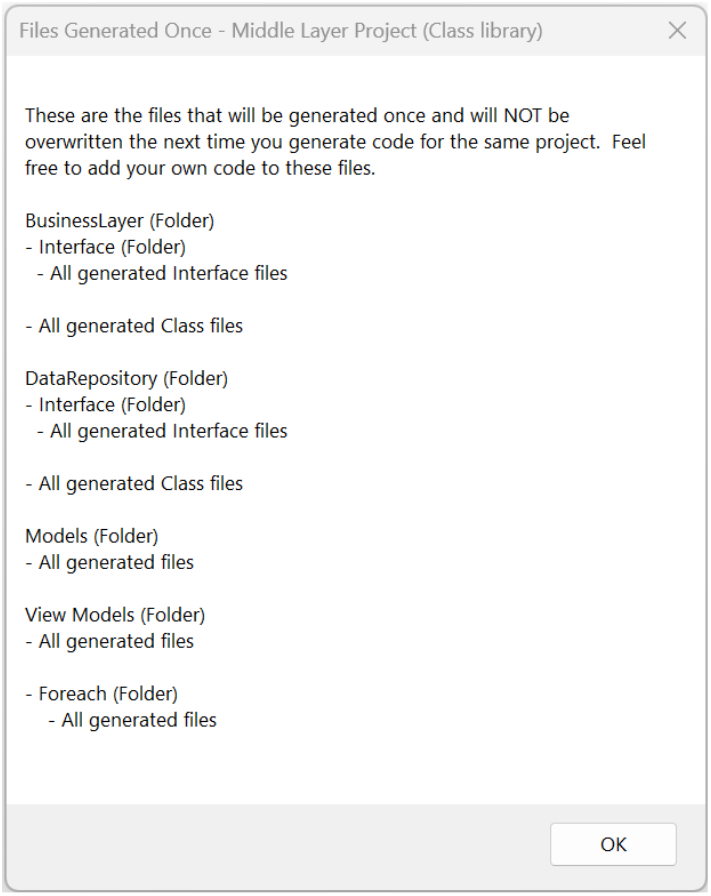
Those are just examples, of course there are a lot more you can do.

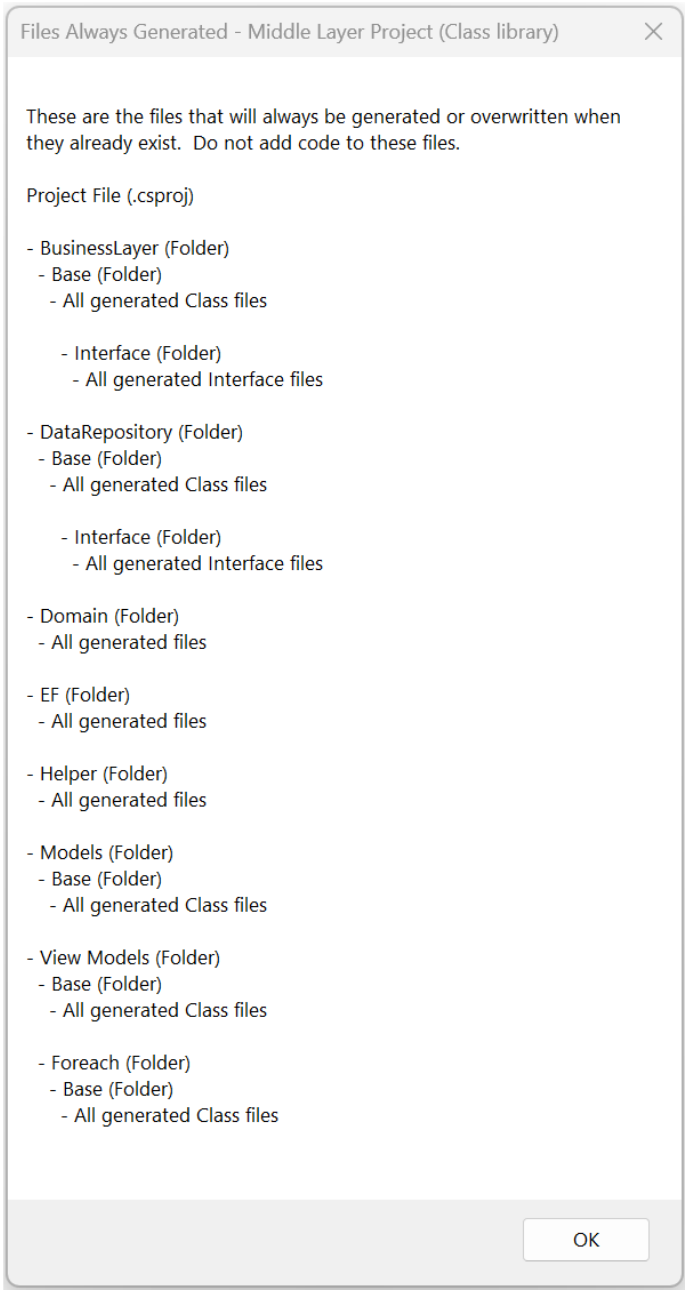
1.1.2 Files That Will Always Be Overwritten

These files will always be generated, and if they already exist, these files will be overwritten without warning. Don’t write code or update any code on these files.

1.2 MIDDLE LAYER PROJECT (CLASS LIBRARY)

There are 2 links under the *Class Library Project*. A box pops-up when you click the respective link.





1.2.1 Files That Will Be Written Once

Only generated the first time you generate code for the same project, or when you delete it. Here are some examples.

- **BusinessLayer (Folder)** – Not including All Files under the *Base Folder*.
    - **Interface (Folder)** – You can add new interfaces or update existing ones in any partial interface.
    - **BusinessLayer (Folder)** – You can implement the new interfaces you added in the *Interface Folder* above in the respective partial class found in this table. You can also add new methods or update existing methods.

- **DataRepository (Folder)** – Not including All Files under the *Base Folder*.
    - **Interface (Folder)** – You can add new interfaces or update existing ones in any partial interface.
    - **DataRepository (Folder)** – You can implement the new interfaces you added in the *Interface Folder* above in the respective partial class found in this table. You can also add new methods or update existing methods.
- **Models (Folder)** – Not including All Files under the *Base Folder*. You can add new properties or update existing properties to any partial class.
- **ViewModels (Folder)** – Not including All Files under the *Base Folder*.
    - **ViewModels (Folder)** – You can add new properties or update existing ones in any partial class.
    - **Foreach (Folder)** – Not including All Files under the *Base Folder*.
        - **Foreach (Folder) -** You can add new properties or update existing ones in any partial class.

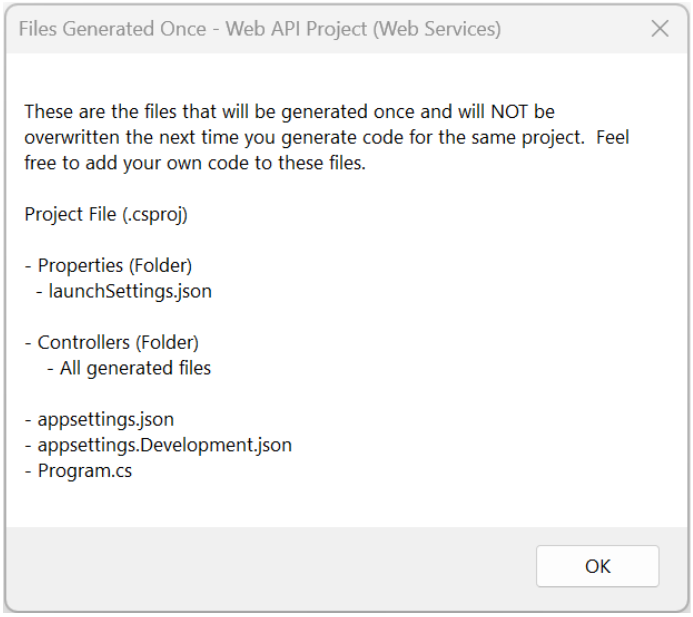
Those are just examples, of course there are a lot more you can do.

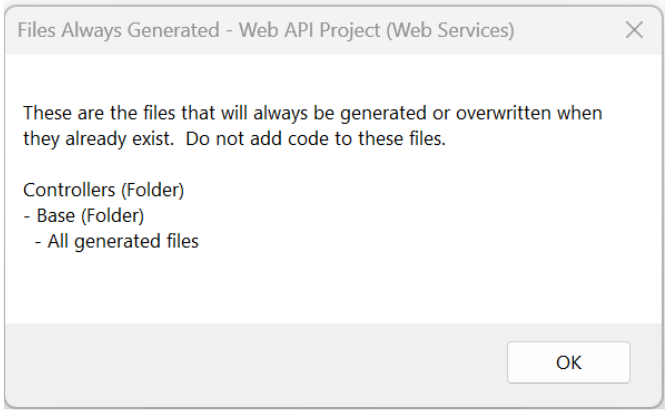
1.2.2 Files That Will Always Be Overwritten

These files will always be generated, and if they already exist, these files will be overwritten without warning. Don't write code or update any code on these files.

1.3 Web API Project

There are 2 links under the *Web API Project*. A box pops-up when you click the respective link.





1.3.1 Files That Will Be Written Once

Only generated the first time you generate code for the same project, or when you delete it. Here are some examples.

- **Project File** – You can add new items in the project.
- **launchSettings.json** – You can add new application launch settings or update current settings.
- **Controllers (Not in Base Folder)** – You can add new Action methods or just methods, or remove existing methods in any partial class under *Controllers* folder. E.g. If you add a new *View* in the *Category* folder, then you could add the respective *IActionResult* method in the *CategoryController* partial class.
- **appSettings.json** – You can add new application settings or update existing settings.
- **Program.cs** – You can inject more objects to independent classes.

Those are just examples, of course there are a lot more you can do.

1.3.2 Files That Will Always Be Overwritten

- Controllers (Folder)
  - Base (Folder)
    - *All generated files*

1.4 Why is it Important to Know Which Files Will be Written Once or Always Overwritten

Note: This part is not in the *App Settings Tab*.

So why do we have some files written just once while the others are always overwritten when we generate code for the same project?

1.4.1 Files That Will Be Written Once

For the files that are written once like the *launchSettings.json* for the *Web Application Project* and *Web API Project*, you might want to add your own code, for this instance you might want to add your own settings to the *launchSettings.json.* AspCoreGen 6.0 MVC will NOT overwrite this file if it already exist.

Another example is the *Solution File (.sln)*. You may want to add another project that is not generated by AspCoreGen 6.0 MVC to your solution, this will make sure that the project will load in the same solution even when you keep generating code for the same project over, and over again.

The same goes with the *Project File (.csproj)*, you will be able to add new MVC Views, images, class files, even references, etc. and know that those you added will always be loaded even after regeneration.

You probably want to add your own *favicon.ico* and overwrite the one we added to the project by default. Or even add new styles to the *site.css* without fear of losing your code.

Another examples is the *_Layout.cshtml*. You are probably going to want to add your own design to the overall web application. Here's your chance.

1.4.2 Files That Will Always Be Overwritten

Do not add your own code in any of these files.

An example would be one of the generated MVC view, the *ListCrudRedirect.cshtml*. Maybe you want to add a new column or remove a column in the generated JQGrid. Or maybe you want no sorting, or *Add New Record* functionality only no *Editing* or *Deleting*. For whatever reason you may want to update the generated code for this MVC view, **Don't**.

You should instead create a new MVC View and add it to the *Web Application Project*. Copy all the code from the *ListCrudRedirect.cshtml*, and then do all your changes there.

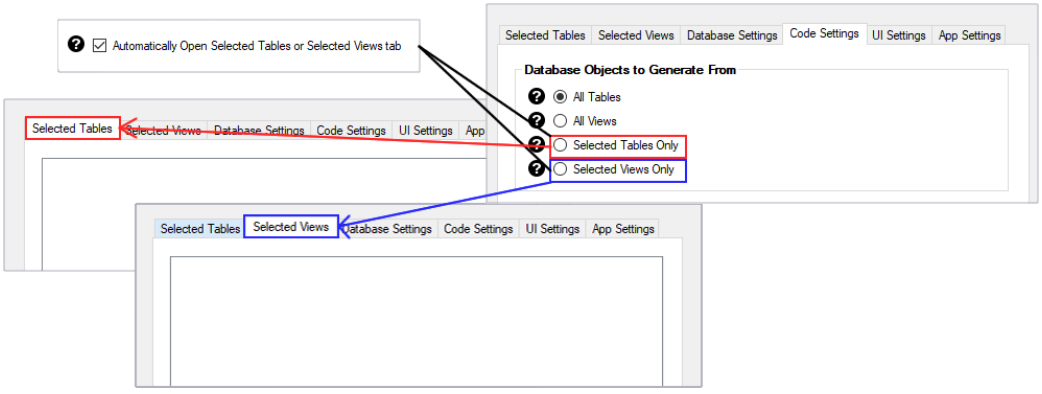
So you have *YourNewView.cshtml*, and it references a javacript file *jqgrid-listcrudredirect.js*, you want to add new javascript functions to it, **Don't**.

Instead, create a new javascript file *your-new-javascript.js*, copy all the code from *jqgrid-listcrudredirect.js* into it, and then reference *your-new-javascript.js* from *YourNewView.cshtml*.

2 Other Settings

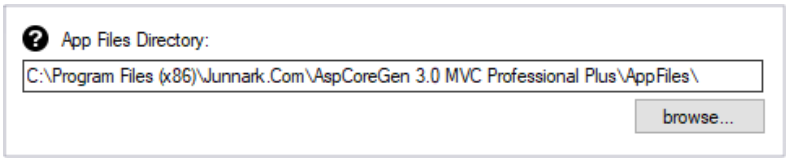
2.1 Automatically Open Selected Tables or Selected Views Tab

In the *Code Settings Tab*, when you click the *Selected Tables Only* or *Selected Views Only* option under the *Database Object to Generate From* and this (*Automatically Open Selected Tables or Selected Views Tab*) is checked, AspCoreGen 6.0 MVC will automatically open the *Selected Tables Tab* or *Selected Views Tab* respectively.



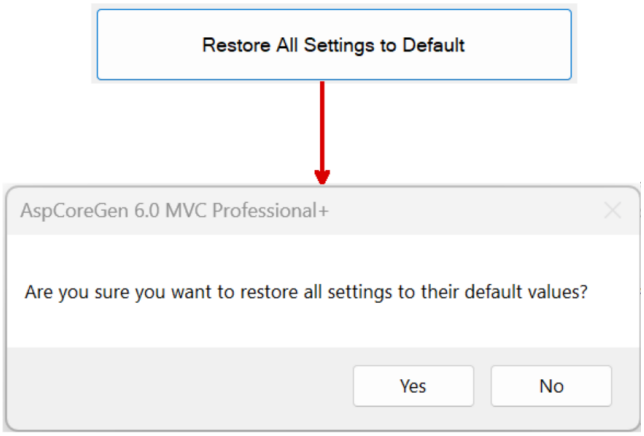
2.2 App Files Directory

The *App Files Directory* is the location of the *App Files* folder is in your computer. The default directory is: *C:\Program Files (x86)\Junnark.Com\AspCoreGen 6.0 MVC Professional Plus\AppFiles*. AspCoreGen 6.0 MVC uses the *App Files Directory* for various settings required by the application.



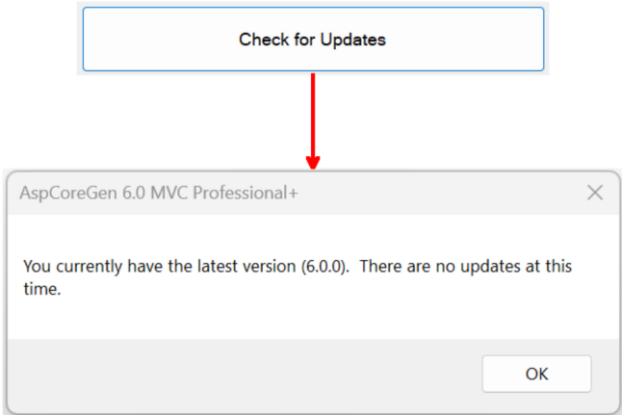
2.3 Restore All Settings to Default (Button)

This button will restore AspCoreGen 6.0 MVC's settings to their default values just like when you first installed it. This will not make you reactivate AspCoreGen 6.0 MVC. It will also pop-up a confirmation dialog to before completely restoring of all settings. Clicking *Yes* will *Restore All Settings to Default*.

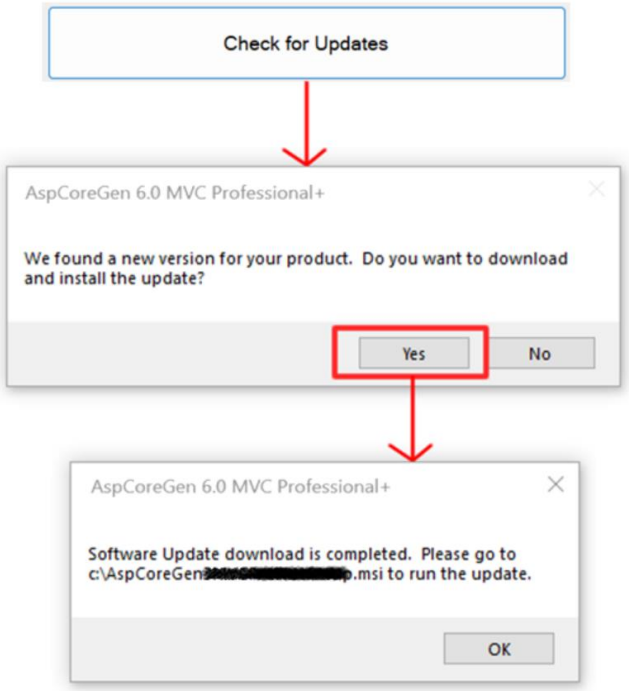


2.4 Check for Updates (Button)

This button will check if there is an available Update for AspCoreGen 6.0 MVC. If there is none, a message will pop-up letting you know that you have the latest version.



The update will be downloaded to a local folder, or it will start the installation when an update is available.



You can read end-to-end tutorials on more subjects on using AspCoreGen 6.0 MVC Professional Plus that came with your purchase. These tutorials are available to customers and are included in a link on your invoice when you purchase AspCoreGen 6.0 MVC Professional.

Note: Some features shown here are not available in the Express Edition.

End of tutorial.