Instructions to create

Visual Studio Templates

[December 7, 2023]

Table of Contents

[**Instructions** 3](#_Toc26545703)

[Update Templates 3](#_Toc26545704)

[Export Templates 5](#_Toc26545705)

[Project Level Template Export 5](#_Toc26545706)

[Item Level Template Export 12](#_Toc26545707)

[Multi-project Template Export 16](#_Toc26545708)

[Create VSIX 17](#_Toc26545709)

[Kepler Template Merger Notes 17](#_Toc26545710)

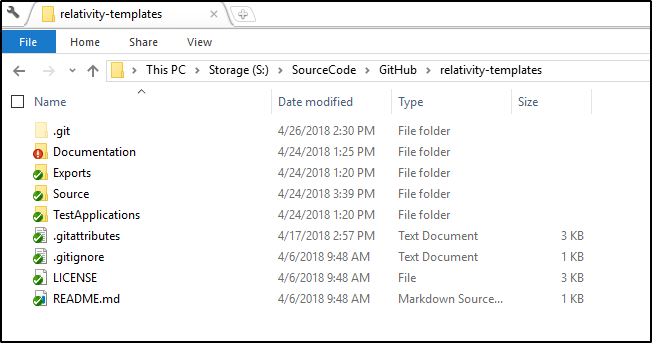
# **Instructions**

## Update Templates

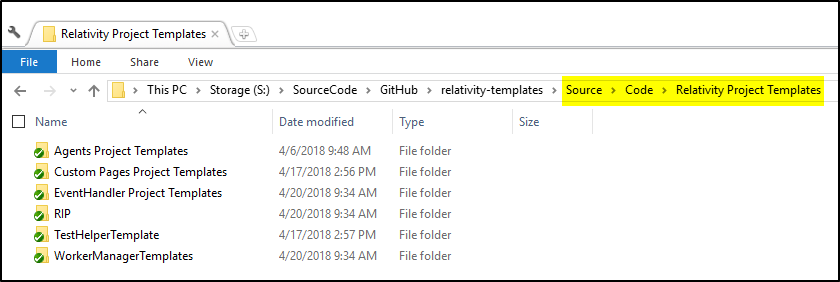
1. Clone repository from Bitbucket by running the following command in PowerShell window.

git clone https://github.com/relativitydev/server-relativity-templates.git

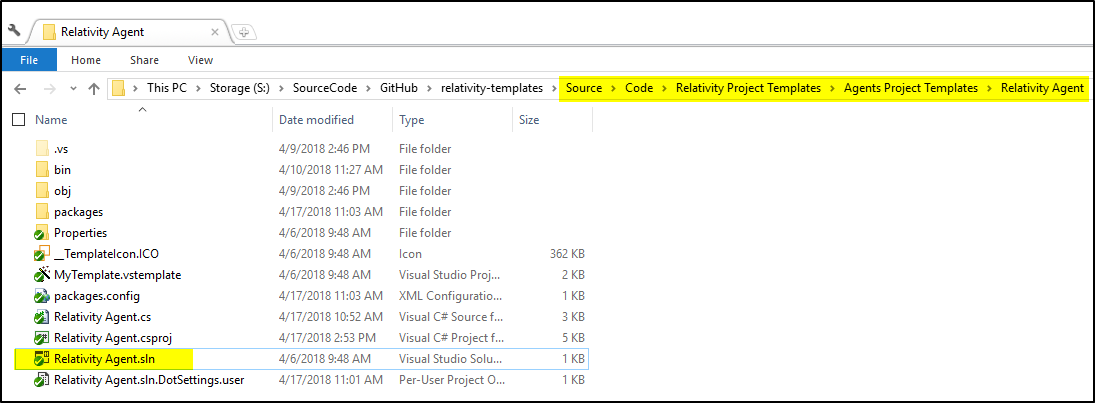
1. The repository contains the following folders.



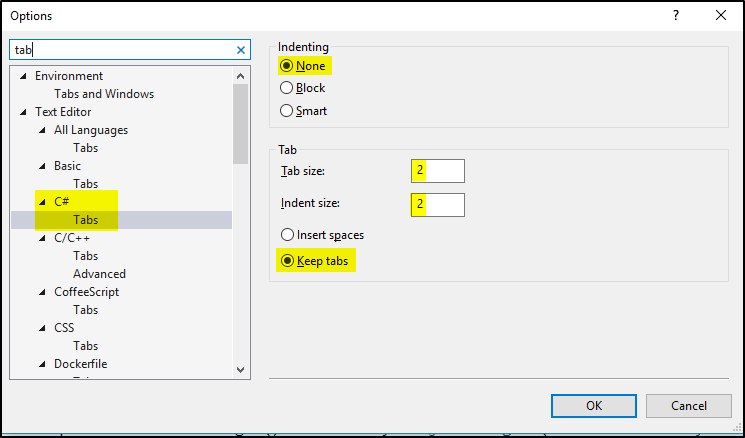
1. Open the Source\Code\Relativity Project Templates folder. It contains the following folders.



1. The instructions going forward will be for the Relativity Agent Template and the same instructions can be applied for other template types.
2. Open the **Relativity Agent.sln** file from the **Source\Code\Relativity Project Templates\Agents Project Templates\Relativity Agent** folder in **Visual Studi0 2022**.



1. In the Visual Studio solution, make the following changes:
   1. Add, Remove or Update best practices (RSAPI, ObjectManager API)
   2. Add, Remove or Update NuGet packages
   3. Add, Remove or Update API helpers
   4. Add, Remove or Update Logging helper
   5. Add proper comments
   6. Spell check all the files
   7. Format tabs and spaces
      1. Make sure your Visual Studio tab settings are as per the below screenshot

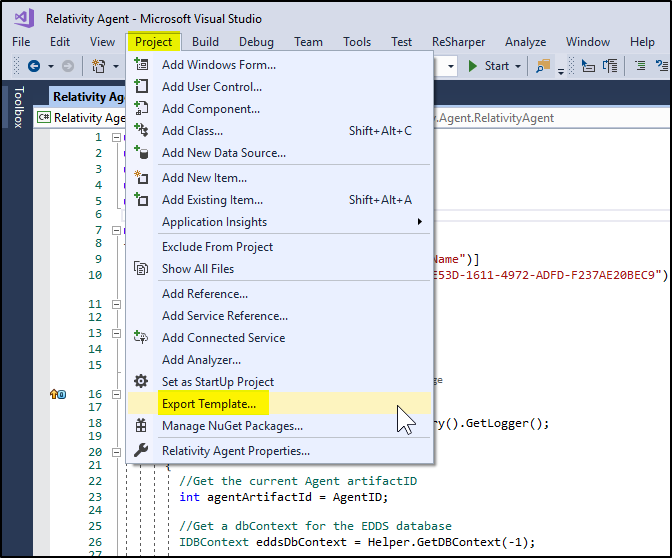


1. Make sure all the projects in the Visual Studio solution builds successfully.

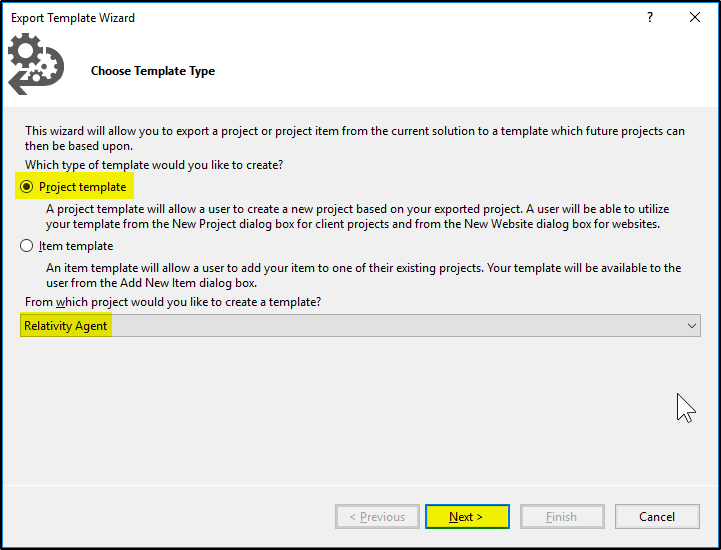
## Export Templates

### Project Level Template Export

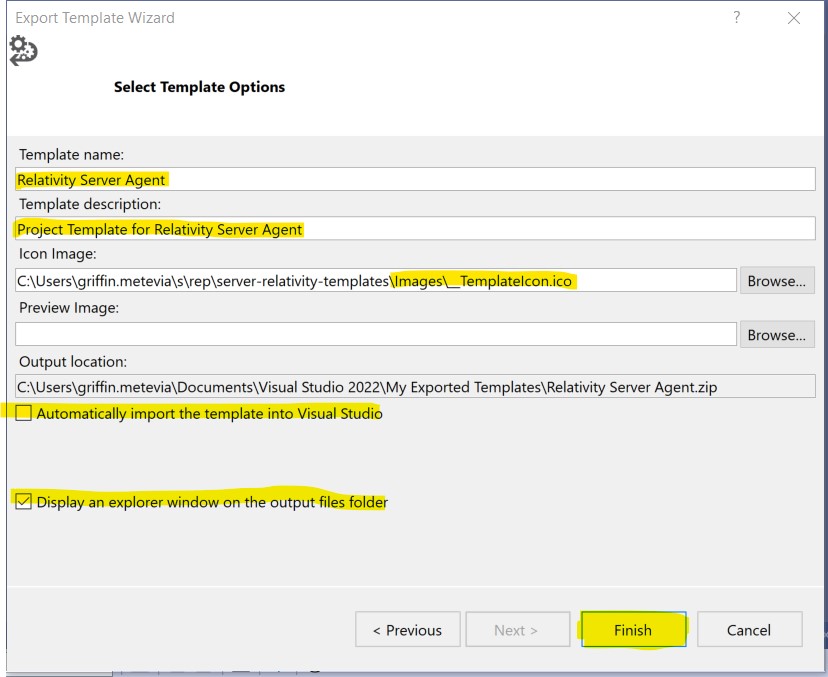
1. Select the **Export Template** option from the Visual Studio **Project** menu.



1. In the popup window select the **Project Template** option and click **Next**.



1. Provide a template name, short spell checked description, Icon image and set other settings as shown in below screenshot. Click **Finish**.



1. Extract the zip file created in the previous step.
2. Open the **Relativity Agent.cs** file.
3. Search for the **$** replacements listed below in all the **.cs**, **.csproj** files and update them.

Visual Studio Template Replacement Documentation:

<https://msdn.microsoft.com/en-us/library/eehb4faa.aspx>

Please find below some examples.

* 1. Guid

Original:

[System.Runtime.InteropServices.Guid("E19FE53D-1611-4972-ADFD-F237AE20BEC9")]

Replacement:

[System.Runtime.InteropServices.Guid("$guid1$")]

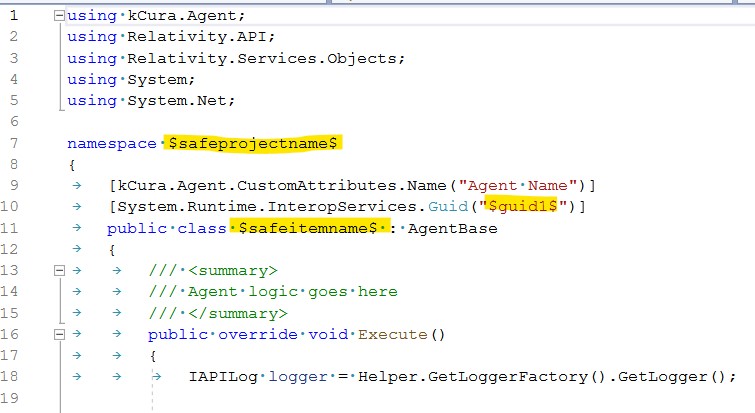
* 1. Class name

Original:

public class RelativityAgent : AgentBase

Replacement:

public class $safeitemname$ : AgentBase



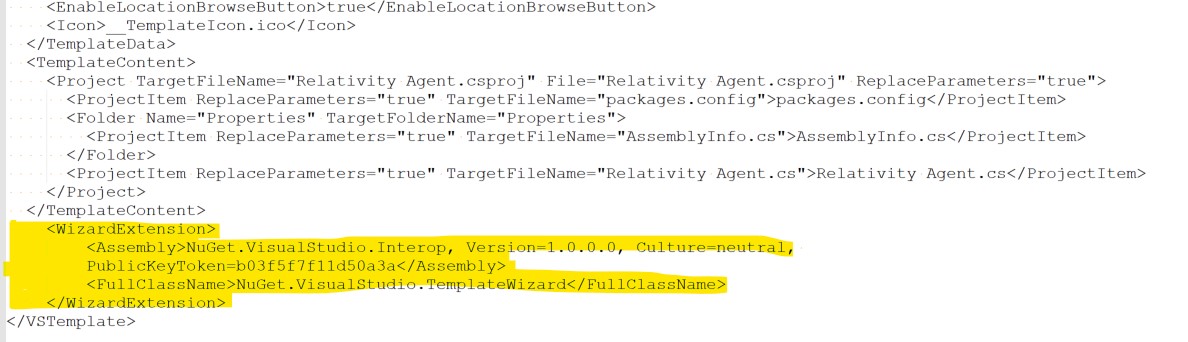
1. Open the **MyTemplate.vstemplate** file.
2. Add the following code to the vstemplate file after the <TemplateContent> tag.

<WizardExtension>

<Assembly>NuGet.VisualStudio.Interop, Version=1.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a</Assembly>

<FullClassName>NuGet.VisualStudio.TemplateWizard</FullClassName>

</WizardExtension>



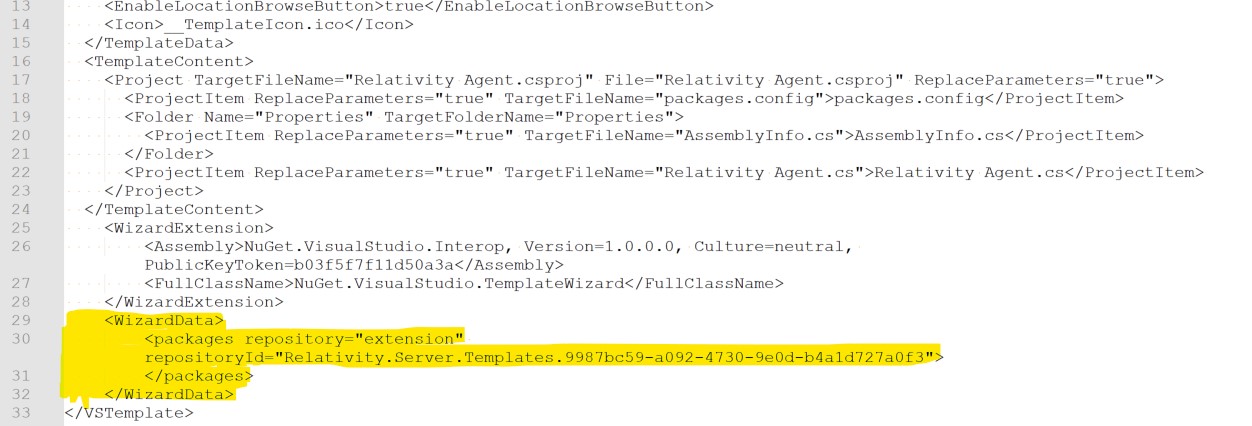
1. Add the following code to the vstemplate file after the <WizardExtension> tag.

<WizardData>

<packages repository="extension" repositoryId="Relativity.Server.Templates.9987bc59-a092-4730-9e0d-b4a1d727a0f3">

</packages>

</WizardData>



1. Add all the NuGet packages used in the project to the <packages> child tag under <WizardData> tag as shown in below screenshot.

<package id="Relativity.Server.Agent.SDK" version="17.4.2" />

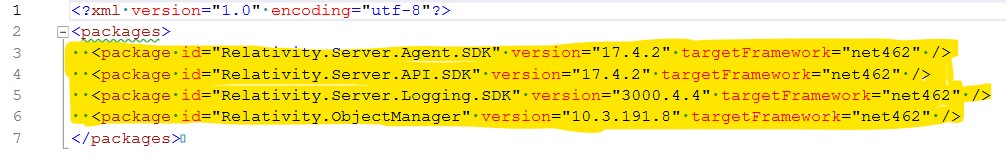
<package id="Relativity.Server.API.SDK" version="17.4.2" />

<package id="Relativity.Server.Logging.SDK" version="3000.4.4" />

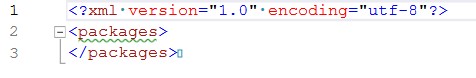
<package id="Relativity.ObjectManager" version="10.3.191.8" />



packages.config file:

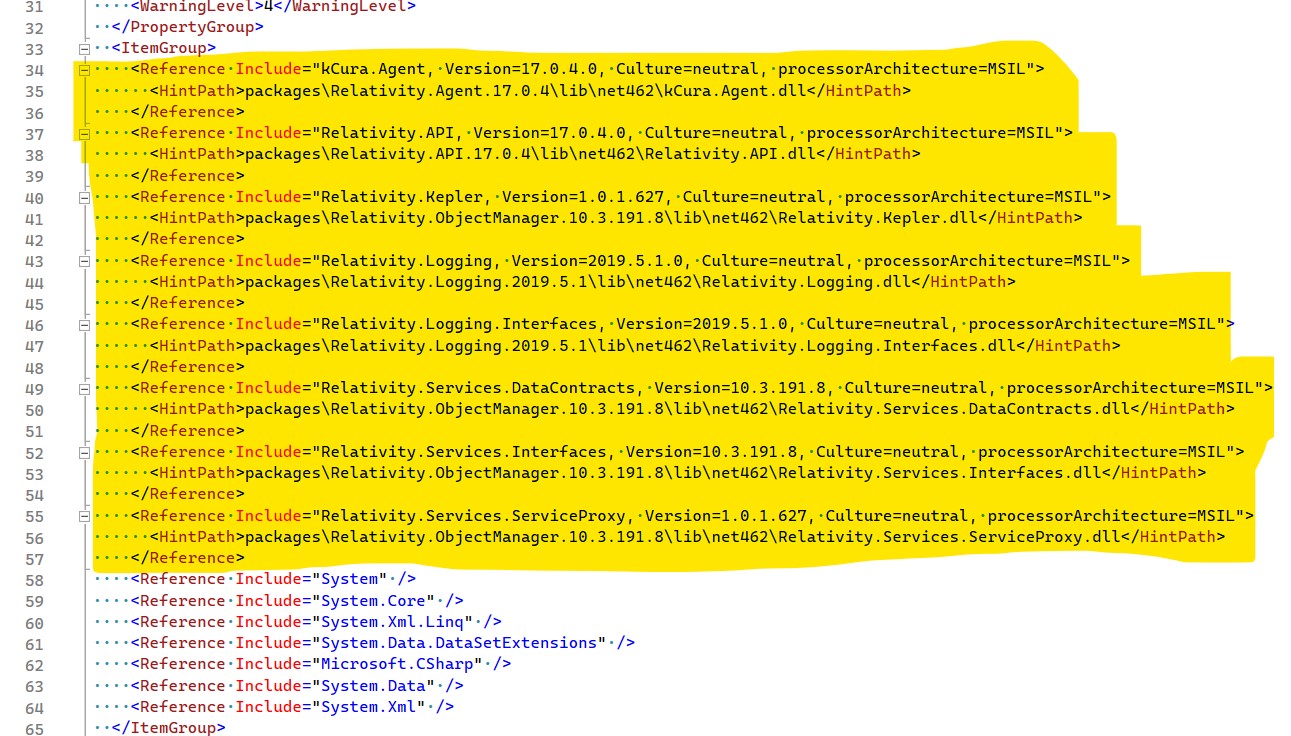


1. Remove all the NuGet package references from the **packages.config** file.

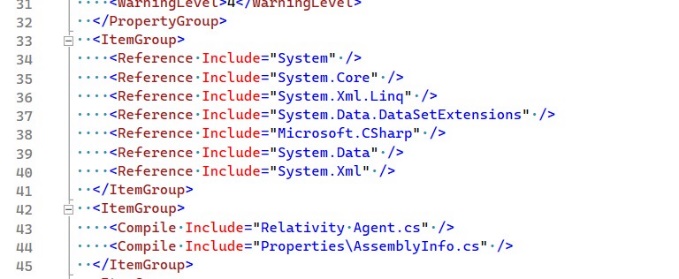


1. Remove all the NuGet package references from the **csproj** file.

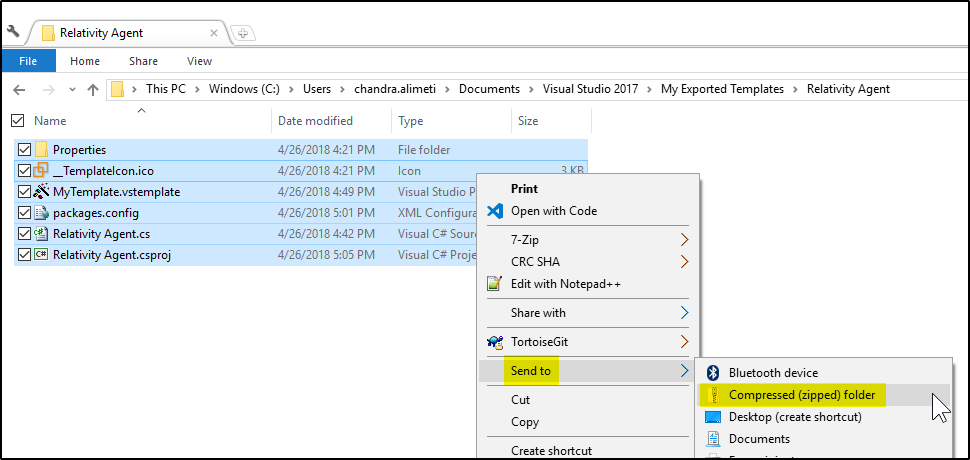
Before:

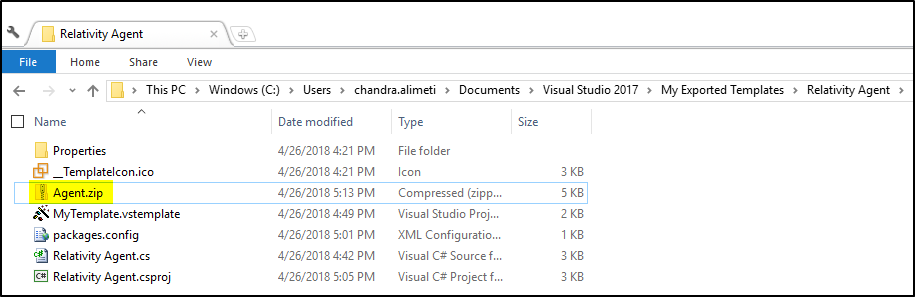


After:

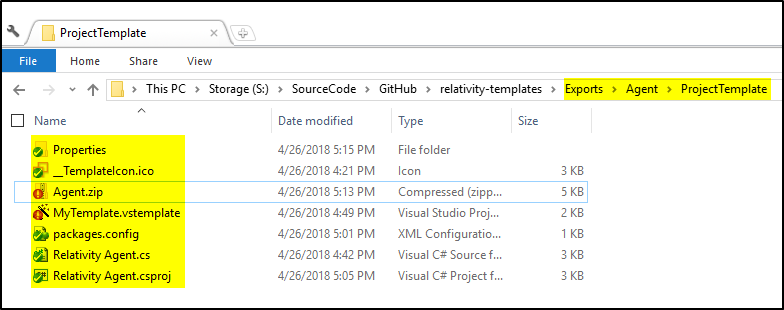


1. Once you are done making all the changes, select all the files and zip it up.
   1. If you are updating the templates
      1. Delete the previous zip file.
      2. Name the zip file with the same name as previous zip file name.





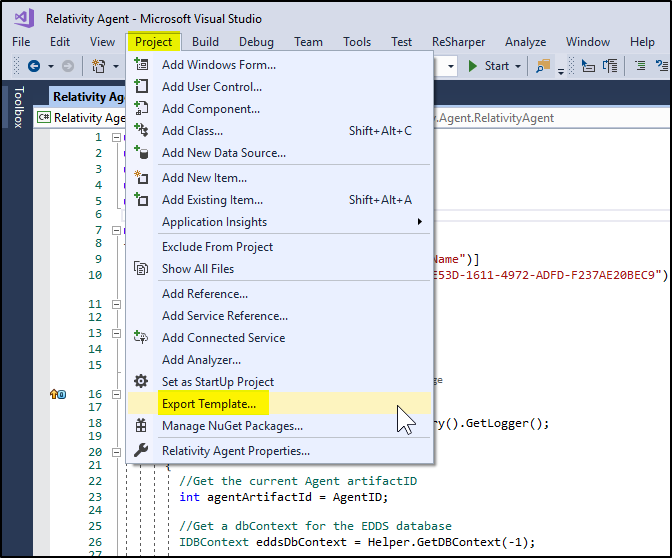
1. Copy all the files from the exported folder along with the created zip file to the **Exports/Agent/ProjectTemplate** folder.



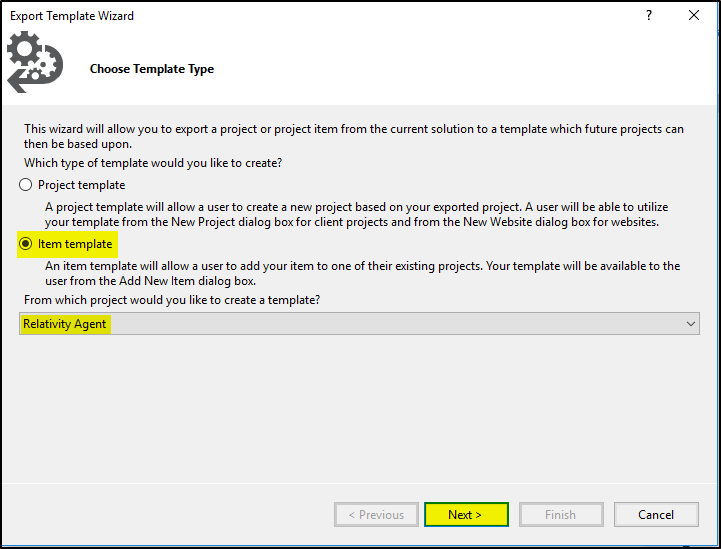
1. Follow the above steps to create Project Level templates for all the other template types.

### Item Level Template Export

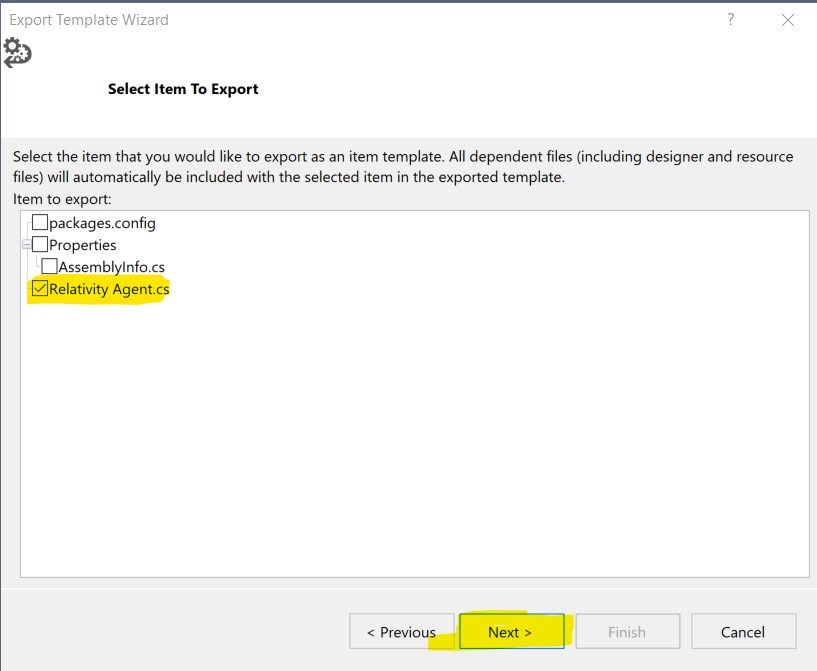
1. Select the **Export Template** option from the Visual Studio **Project** menu.



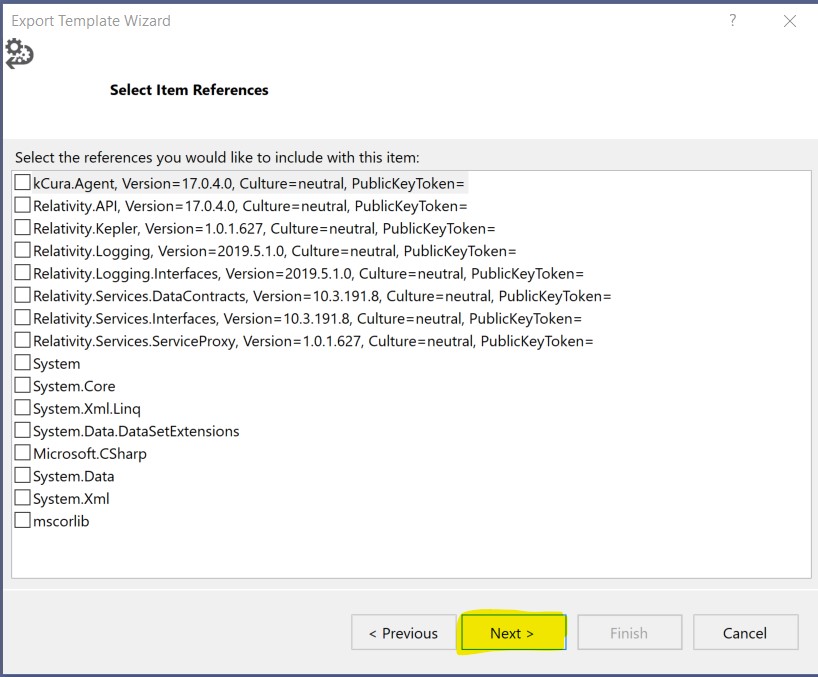
1. In the popup window select the **Item Template** option and click **Next**.



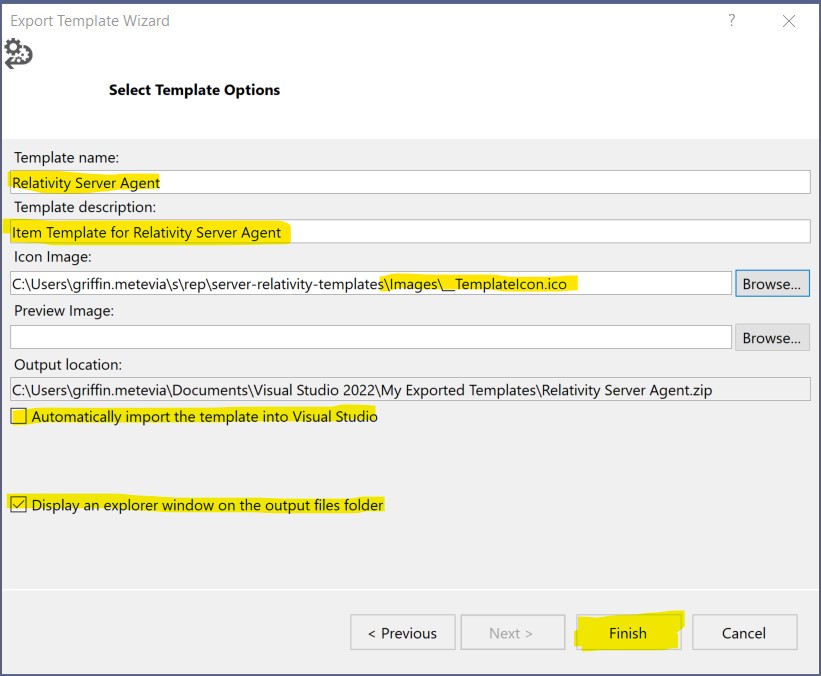
1. Select the **.cs** file to export and click **Next**.



1. Do **not** select any references and click **Next**.



1. Provide a template name, short spell checked description, Icon image and set other settings as shown in below screenshot. Click **Finish**.



1. Extract the zip file created in the previous step.
2. Open the **Relativity Agent.cs** file.
3. Search for all **$** variables and make sure they are correct. Please find below some examples.
   1. Guid

Original:

[System.Runtime.InteropServices.Guid("E19FE53D-1611-4972-ADFD-F237AE20BEC9")]

Replacement:

[System.Runtime.InteropServices.Guid("$guid1$")]

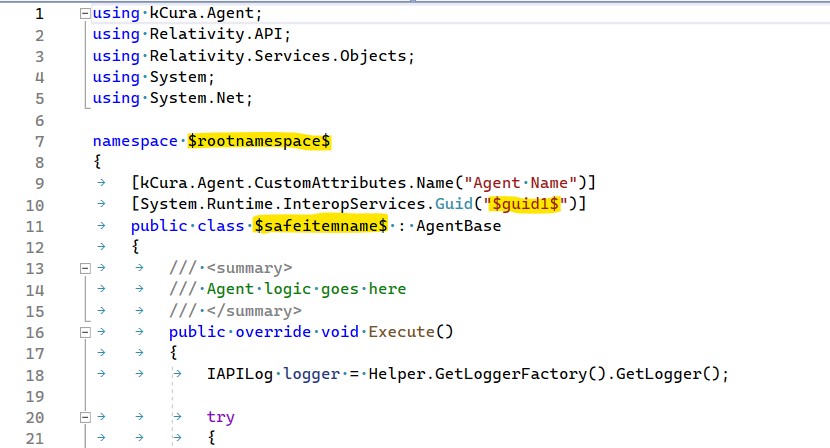
* 1. Class name

Original:

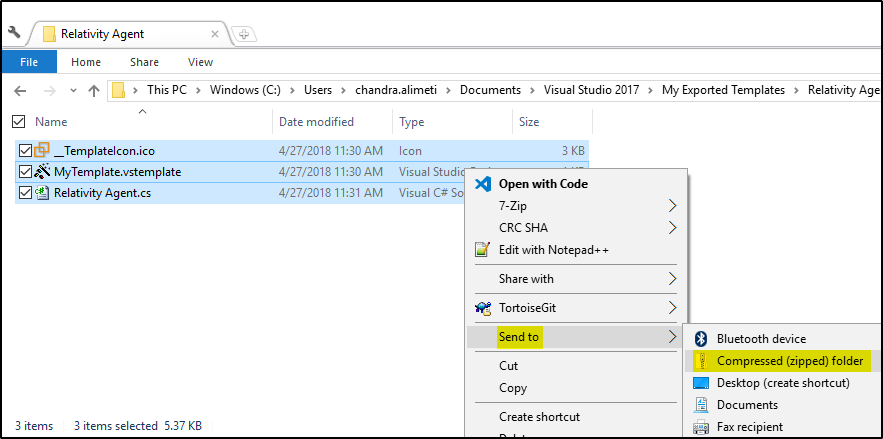
public class RelativityAgent : AgentBase

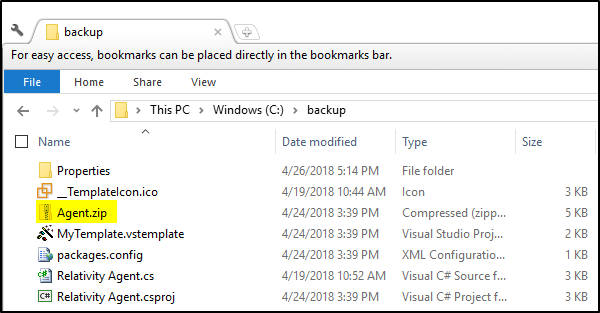
Replacement:

public class $safeitemname$ : AgentBase

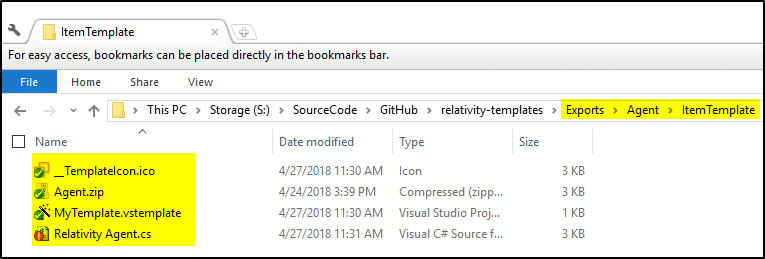


1. Once you are done making all the changes, select all the files and zip it up.
   1. If you are updating the templates
      1. Delete the previous zip file.
      2. Name the zip file with the same name as previous zip file name.





1. Copy all the files from the exported folder along with the created zip file to the **Exports/Agent/ItemTemplate** folder.



1. Follow the above steps to create Item Level templates for all the other template types.

### Multi-project Template Export

1. Create Project Templates normally for each project in the multi project solution. Do not Zip each project individually. (Also if your project references another project in the solution, update the reference as follows Ex: using RIP.Provider 🡪 using $saferootprojectname$.Provider. Update .csproj files this way also where other projects in the solution are referenced).
2. Add the following to the .vstemplatefiles below the other WizardExtension tag (this will replace the $saferootprojectname$ variables with the correct values when the user names their project when creating the template):

<WizardExtension>

<Assembly>RelativityWizard, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null</Assembly>

<FullClassName>RelativityWizard.ChildWizard</FullClassName>

</WizardExtension>

1. Go to the corresponding **Exports** folder (Ex: Exports/ManagerWorker/Project/) and replace all of the individuation project folders with the updated Project Files and .vstemplate you just created.
2. Zip up all of the files in the “Project” folder and name the zip file the same as it’s corresponding file in the **Exports** folder

## Create VSIX

1. Open the **AllRelativityTemplates.sln** file from **Source\VSIX\AllRelativityTemplates** folder in Visual Studio 2017.
2. Copy all of the ZIP files for the Item templates in the **Exports** folder to the corresponding folder in **Source\VSIX\AllRelativityTemplates\ItemTemplates\Relativity\**
3. Copy all of the ZIP files for the Project templates in the **Exports** folder to the corresponding folder in **Source\VSIX\AllRelativityTemplates\ProjectTemplates\Relativity\**
4. Update the Templates Version number in the wizard.
5. Open the **Source\VSIX\AllRelativityTemplates.sln** and build the solution. The updated .vsix file will be located at **Source\VSIX\AllRelativityTemplates\bin\Debug\AllRelativityTemplates.vsix** after the build succeeds.
6. Upload this file to the Visual Studio Marketplace. After uploading, place this file in the **VSIX Uploaded** folder.

## Kepler Template Merger Notes

Some notes and story behind the way we implemented the Kepler Templates.

* The original Kepler Template Solution was developed by another team, and worked great, but it was not connected to the original Relativity templates project. It needed to be merged and thus we had to do some less than ideal things to achieve the merger.
  + This resulted in exporting each Kepler project manually in the way we did it above for our older templates, with minimal changes to the code in the projects.
  + Most of the changes were done at the Meta data level, such as making the vstemplates mimic some of the settings and style we have in other older projects.
  + Ensure we aren’t using packages.config and other files and all NuGet references are done in the .vstemplates files.
  + What this means is you should edit the code, files, and vstemplate xml manually (for now) under *D:\SourceCode\GitHub\relativity-templates\Exports\Kepler* before pushing it to the AllRelativityTemplates solution.
  + In terms of the Wizards, this was a bit annoying to transfer over, but it mostly involved transferring over the Windows Forms over to the AllRelativityTemplates project manually, some namespace renames, and making sure we took out the digital signature of the dll since none of the other older ones did.
    - Transferring over a Windows Form can be tricky, if you copy and paste the 3 files over, you’ll need to manually edit the csproj file outside of the Visual Studio (look at how other Windows Forms are structured in there and mimic the data placement), or, create a similarly named form in Visual Studio and override it with the other code outside of Visual Studio.
    - One catch that got me was I had to change the Namespace in the copied over Wizards, and utilize that new namespace properly in the vstemplate WizardExtension references to the wizard dll.
    - This is something to consider adding back in the future, and the key is already in the Kepler project that was committed, and we just need to update the project settings to utilize a key on build (very easy stuff in Visual Studio, just grab the snk that already exists in the Wizard project in AllRelativityTemplates)
      * 