Doc as Code tools

Design target and principles

This three tools are made for the VS reference document prototype. The project spec is as follow link:

[VS reference documentation spec](https://microsoft-my.sharepoint.com/personal/peterbi_microsoft_com/_layouts/15/WopiFrame.aspx?sourcedoc=%7bF00DB8A4-B0BF-4C71-B193-87A3ACFEB385%7d&file=OA%20reference%20doc%20spec.docx&action=default)

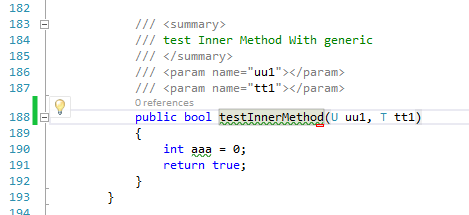
User Manual

DocAsCode plug-in

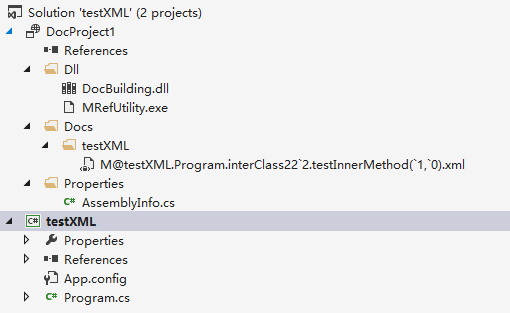
General

This plug-in project is under {IXP}\Personal\niqun\DocAsCode. Its mian function is :

* Analyze all your C# code, and determine if it is an API that should generate doc files to record its information.
* If it is an API that should be record (public or protected method, constructor，filed, struct, enum, class or namespace), there will be a warning with a wavy line under it.
* Create a doc project (need doc project plug-in installed) in the same solution.
* Click the wary line, there will be a tip bulb showed up, click the bulb, there will be a tip “Create a documentation.” under it.



* Click the tip, the corresponding doc file will be generated under the doc project. Here we used ‘@’ character to replace ‘:’, and used ‘!’ to replace ‘\*’ for the filename construction rules of Windows OS.
* **All the doc file format is DDUE by now.**

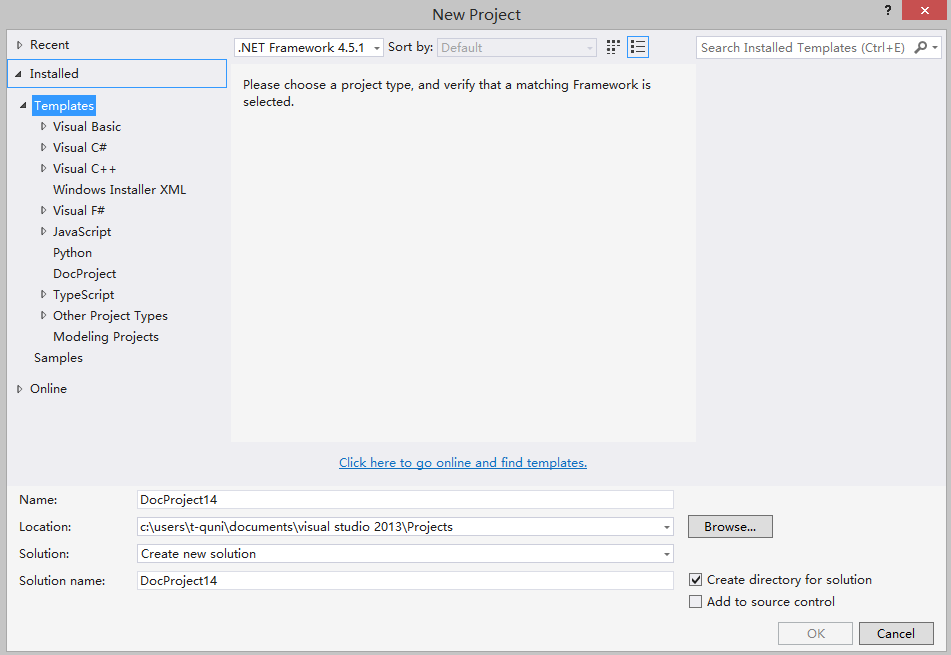


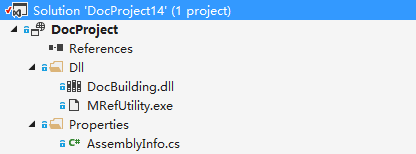
Doc project plug-in

General

This plug-in project is under {IXP}\Personal\niqun\DocProject. Its mian function is :

* Provide a new project type named DocProject.
* Open File->new->project, there will be a DocProject type in Installed/Templates folder (**This need .NET Framwork 4.5.1**).
* Type in the project name and there will be a Doc Project in the new solution.





Doc Buidling project

General

This is a class project which implemented the Microsoft.Build.Framework.ITask interface or extended Microsoft.Build.Utilities.Task class. It is the building process of the doc project. The main function of the Building project is to generate the HTML files for the C# code project’s APIs, which translated from DDUE format xml files:

* Get the directory of the corresponding C# code project’s assembly(.exe or .dll)
* Get the directory of the corresponding C# code project’s doc files(.xml)
* Get the directory of the MSBuild.exe in the local device.
* Start the MrefUtility.exe with all the above information as parameters.
* Generate the HTML files.

By now it is registered in the DocProject.ProjectTemplate\Dll\DocProject.docproj file of DocProject plug-in project.

But the MrefUtility.exe, which under {IXP}\IXP\CLiX\Work\Build\Application\MRefUtility\ bin\Debug, needs some config files, so unfortunately it still won’t work by now.

<ItemGroup>

<None Include="Properties\AssemblyInfo.cs"></None>

<None Include="Dll\DocBuilding.dll"></None>

<None Include="Dll\MRefUtility.exe"></None>

</ItemGroup>

Modified part of Build service for DocAsCode

General

To generate the HTML files for the C# code project APIs, I modified some part of Build service project under {IXP}\IXP\CLiX\Work\Build directory. The changed project are as follows:

* {IXP}\IXP\CLiX\Work\Build\Library\MRef.Stubbing
* {IXP}\IXP\CLiX\Work\Build\source\TopicEngine\InternalComponentLibrary\InternalComponentLibrary\MRefReflection

For the reason I only changed some C# code in the second project, to not including too many files, I only put the directory I changed.