T4 template:

Nombre del template puede ser tt o t4

Tener en cuenta de poner nombre del archivo de la referencia:

<#@ assembly name="System.dll" #>

<#@ assembly Name="System.Data.dll"

Si agrega referencia a clase local:

<#@ assembly name="Fwk.GuidPk.dll" #> Si es la solución donde esta el guidance:

<#@ import namespace="Fwk.Guidance.Core" #> agrega el using:

**Crear solución:**

En Fwk.GuidancePackage.xml en el primer Recipe. Esto:

<Action Name="CreateSampleUnboundTemplateRef" Type="RefCreator" AssetName="**Projects\SampleLibrary\SampleLibrary.vstemplate**"

ReferenceType="Fwk.GuidPk.References.SolutionFolderAReference, Fwk.GuidPk" />

**Projects\SampleLibrary\SampleLibrary.vstemplate**

SampleSolution.vstemplate

Este archive llama al recipe: CreateSolution en WizardData

<WizardData>

<Template xmlns="http://schemas.microsoft.com/pag/gax-template"

SchemaVersion="1.0"

Recipe="CreateSolution">

**Los poyectos creados se definen dentro de ProjectCollection:**

<TemplateContent>

<ProjectCollection>

<SolutionFolder Name="SampleFolder">

<ProjectTemplateLink ProjectName="$ProjectName$">Projects\ConsoleApplication\ConsoleApplication.vstemplate</ProjectTemplateLink>

</SolutionFolder>

<SolutionFolder Name="Other Projects" />

<SolutionFolder Name="WebSites">

<ProjectTemplateLink ProjectName="$WebSiteName$">Projects\WebSite\WebSite.vstemplate</ProjectTemplateLink>

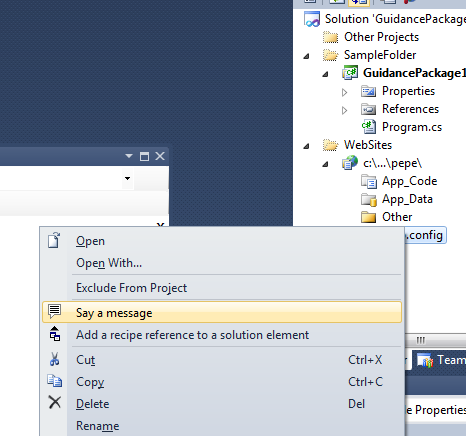
**Para agregar menues o récipes a elemento del proyecto creado:**

<RecipeReference Name="HelloWorldRecipe" Target="\SampleFolder" />

<RecipeReference Name="CustomWizardPages" Target="\SampleFolder" />

<RecipeReference Name="HelloWorldRecipe" Target="\WebSites\$WebSiteName$" />

<RecipeReference Name="HelloWorldRecipe" Target="\WebSites\$WebSiteName$\Web.config" />



Projects\ClientApplication\ClientApplication.vstemplate doesn't exist.

Dar la opción de seleccionar referencia a proyecto :

Se agrega esto al recipe:

<GatheringServiceData🡪 Wizard 🡪 Pages 🡪 Page:

<Field ValueName="Project" Label="Reference Project">

<Tooltip>Shows a complex editor for selecting a project in the current solution, provided with the Library.</Tooltip>

<Editor Type="Microsoft.Practices.RecipeFramework.Library.Editors.SolutionPickerEditor, Microsoft.Practices.RecipeFramework.Library" />

</Field>

T4 for .edmx file

**Determining the Namespace of Generated Code**

For C# projects: Specify the namespace in the **Custom Tool Namespace** property of the .edmx file, the specified namespace is used.

For ASP.NET Website projects, if the .edmx file is in the root of the App\_Code folder, the namespace of generated code will be the conceptual model’s namespace as defined in the CSDL content of the .edmx file (for more information, see [Schema Element (CSDL)](http://msdn.microsoft.com/en-us/library/bb399276.aspx)). If the .edmx file is in a sub-folder of the App\_Code folder, the namespace of the generated code will be the folder path relative to the App\_Code folder. For example, if the .edmx file is in App\_Code/Folder1/Folder2, the namespace of generated code will be Folder1.Folder2.

### Working with Multiple .edmx Files in a Project

Generated code can contain conflicting class names if multiple .edmx files based on databases with common table names are

For C# and Visual Basic projects

Specify a unique value for the **Custom Tool Namespace** property of each .edmx file.

For ASP.NET Website projects, put each .edmx file in a separate sub-folder of the App\_Code directory

Menu Icons Id

|  |  |
| --- | --- |
| <Icon ID="1047"/> |  |
| <Icon ID="525"/> |  |
| <Icon ID="1429"/> |  |
| <Icon ID="172"/> |  |
| <Icon ID="689"/> |  |
| <Icon ID="639"/> |  |
| <Icon ID="1049"/> |  |
| <Icon ID="680"/> |  |
| <Icon ID="1039"/> |  |
| <Icon ID="630"/> |  |