

How to use {smartassembly} with MSBuild?

If you want to integrate {smartassembly}'s process directly into the Build process, you need to do the following:

1. First, you need to create your project with the GUI mode of {smartassembly}. The main assembly must be the assembly in the `\obj\Release` folder, and NOT the one in `\bin\Release`. The location of the destination assembly does not matter.
2. Then, you need to edit your C#/VB.NET project with an xml editor (or even notepad). The project file looks like this:

```
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <PropertyGroup>
    ...
  </PropertyGroup>
  ...
  <ItemGroup>
    ...
  </ItemGroup>
  <Import Project="$(MSBuildToolsPath)\Microsoft.CSharp.targets" />
  <!-- To modify your build process, add your task inside one of the targets below and uncomment it.
       Other similar extension points exist, see Microsoft.Common.targets.
  <Target Name="BeforeBuild">
  </Target>
  <Target Name="AfterBuild">
  </Target>
  -->
</Project>
```

You need to add a few lines after the `<Import Project="...">` bloc:

```
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <PropertyGroup>
    ...
  </PropertyGroup>
  ...
  <ItemGroup>
    ...
  </ItemGroup>
  <Import Project="$(MSBuildToolsPath)\Microsoft.CSharp.targets" />

  <UsingTask TaskName="SmartAssembly.MSBuild.Tasks.Build" AssemblyName="SmartAssembly.MSBuild.Tasks,
Version=4.0.0.0, Culture=neutral, PublicKeyToken=cd3409ee69028647" />
  <Target Name="BeforeBuild" Condition="'$(Configuration)' == 'Release' ">
    <CreateProperty Value="true">
      <Output TaskParameter="Value" PropertyName="RunSmartAssembly"/>
    </CreateProperty>
  </Target>
  <Target Name="AfterCompile" Condition="'$(RunSmartAssembly)' != '' ">
    <SmartAssembly.MSBuild.Tasks.Build ProjectFile="c:\temp\TestMSBuild\TestMSBuild.{sa}proj"
OverwriteAssembly="True" />
  </Target>
</Project>
```

The first Xml node adds a reference to SmartAssembly.MSBuild.Tasks.dll, which is installed into the GAC:

```
<UsingTask TaskName="SmartAssembly.MSBuild.Tasks.Build" AssemblyName="SmartAssembly.MSBuild.Tasks,
Version=4.0.0.0, Culture=neutral, PublicKeyToken=cd3409ee69028647" />
```

The second Xml node tells MSBuild to set the *RunSmartAssembly* property in the *BeforeBuild* event, when the project is *fully* built in *Release* mode. This is necessary because Visual Studio is always compiling in the background:

```
<Target Name="BeforeBuild" Condition=" '$(Configuration)' == 'Release' ">
  <CreateProperty Value="true">
    <Output TaskParameter="Value" PropertyName="RunSmartAssembly"/>
  </CreateProperty>
</Target>
```

The last Xml node tells MSBuild to protect the assembly in the *AfterCompile* event, if the *RunSmartAssembly* property is set:

```
<Target Name="AfterCompile" Condition=" '$(RunSmartAssembly)' != '' ">
  <SmartAssembly.MSBuild.Tasks.Build ProjectFile="c:\temp\TestMSBuild\TestMSBuild.{sa}proj"
  OverwriteAssembly="True" />
</Target>
```

The following properties are available:

* *ProjectFile* is required. It must be set to the filename of the {smartassembly} project.

* *OverwriteAssembly* is required and set to *True* if you want to overwrite the original assembly with the obfuscated one. If this option is not set, {smartassembly} will use the destination file name from the *ProjectFile* project.

Other optional options are:

* *Input*. Set a specific input file name instead of the one from the *ProjectFile* project.

* *Output*. Specify the output file name for the obfuscated assembly. This property is ignored if the *OverwriteAssembly* property is set to *True*.

* *AssemblyVersion*. Set a specific version number for the Assembly. This will change the version for the *AssemblyName*, not the file version.

* *MarkAsReleased*. Set this property to true to automatically mark the protected assembly as *Released*. This is necessary only if you use the exception reporting feature and want to release the processed assembly.

This way, {smartassembly} adds a flag in the database to ensure that the associated map will never been deleted. This is absolutely necessary, to avoid the multiplication of useless Maps. A map file, not marked as released, will be automatically deleted, after 15 days without new reports. {smartassembly} assumes it was a test build, no longer used.