**Integrating StyleCop into MSBuild**

**v4.1.6**

December 1st, 2007

**Motivation**

Microsoft has standardized on MSBuild as the default way for setting up build tasks. StyleCop provides an MSBuild task which can be used to integrate StyleCop directly into the build system. There are various benefits to integrating StyleCop directly into the build in this way, including:

1. MSBuild integration allows StyleCop to run automatically every time a developer compiles a file on the command line or from within Visual Studio. No additional work is required on the part of the developer to run StyleCop.
2. MSBuild allows StyleCop violations to be reported either as build errors or as build warnings.
3. MSBuild automatically passes the correct conditional compilation flags to StyleCop. StyleCop needs knowledge of the currently set conditional compilation flags in order to properly parse and analyze a code file.

**Details**

The default MSBuild task for StyleCop turns StyleCop on by default. It is possible to disable StyleCop for a particular project by setting a flag within the project file, or by setting a flag within the build environment before executing the build. In addition, by default the task treats StyleCop violations as build warnings rather than errors.

It is possible to edit the default StyleCop MSBuild task or create your own to override or change these behaviors so that StyleCop will be disabled by default, or so that StyleCop violations will be treated as build errors.

**Setup**

Smaller development groups with relatively few projects can set up the StyleCop MSBuild task by simply importing the task directly into every project file, using standard MSBuild import syntax. Larger development groups may desire a more centralized approach, with a build management team responsible for integrating StyleCop into the build environment. In this case, it may be advisable to import the StyleCop MSBuild task into another MSBuild task which is already included by every project file in the build system.

These instructions describe setting up the default StyleCop MSBuild targets file. If your group uses its own custom StyleCop targets file, the setup tasks may vary.

1. Determine the correct location within the source tree for the StyleCop MSBuild task and binaries.
2. Copy the StyleCop files to this folder and check them into your source control system. Once the installer has completed successfully, the files can be found in the folder c:\Program Files\MSBuild\ StyleCop. The following files should be copied:

StyleCop.Targets

StyleCop.dll

StyleCop.CSharp.dll

StyleCop.CSharp.Analyzers.dll

If your team has any custom StyleCop parsers or analyzers in addition to these, check these binaries into this folder as well.

If your team is using a custom MSBuild target file for StyleCop, check that file into this folder rather than the default StyleCop targets file.

1. Add the following import tag to every project in your build environment, or to a centralized MSBuild task which is included into every project:  
     
   <Import Project="{Location}\StyleCop.Targets" />  
     
   Note: The import forStyleCop.Targets must be placed after the Import for Microsoft.CSharp.targets, otherwise it will not work.
2. If you wish to disable StyleCop from running on a particular project, edit the project file for that project. Add the following line to the default property group within the file. This will cause StyleCop to become disabled for this project.   
     
   <StyleCopEnabled>false</StyleCopEnabled>
3. If you wish to disable StyleCop from running on particular files in a project that has StyleCop enabled, edit the project file by changing the Compile node for the file you wish to disable. Change the Compile node by adding the XML tag ExcludeFromStyleCop.  
     
   <Compile Include="Class1.cs">  
    <ExcludeFromStyleCop>true</ExcludeFromStyleCop>  
   </Compile>