Visual Basic C#
Visual C++
F#
JScript
FxCop
Send Feedback

FxCop is an application that analyzes managed code assemblies (code that targets the .NET Framework common language runtime) and reports information about the assemblies, such as possible design, localization, performance, and security improvements. Many of the issues concern violations of the programming and design rules set forth in the <u>Design Guidelines for Class</u> <u>Library Developers</u>, which are the Microsoft guidelines for writing robust and easily maintainable code by using the .NET Framework.

FxCop is intended for class library developers. However, anyone creating applications that should comply with the .NET Framework best practices will benefit. FxCop is also useful as an educational tool for people who are new to the .NET Framework or who are unfamiliar with the .NET Framework Design Guidelines.

FxCop is designed to be fully integrated into the software development cycle and is distributed as both a fully featured application that has a graphical user interface (FxCop.exe) for interactive work, and a command-line tool (FxCopCmd.exe) suited for use as part of automated build processes or integrated with Microsoft Visual Studio® .NET as an external tool.

Visit the <u>MSDN Code Gallery Code Analysis Web site</u> for support, updated documentation, and new rule topics.

In This Section

Getting Started with FxCop

Provides a quick introduction to FxCop.

Using FxCop

Provides a complete description of the FxCop user interface and features.

Using FxCopCmd

Provides a complete description of the FxCopCmd command-line options and output results.

FxCop Warnings

Provides in-depth discussion of each FxCop rule.

FxCop FAQ

Provides answers to frequent questions about FxCop.

Visual Basic C#
Visual C++
F#
JScript
Getting Started with FxCop
Send Feedback

FxCop analyzes programming elements in managed assemblies, known as targets, by using rules that return informational messages about the targets when the rules are violated. A report that contains the informational messages appears in the user interface, or in the output window if you use the command-line tool. Messages identify any relevant programming and design issues and, when it is possible, supply information about how to fix the target. A message is associated with both a specific rule and a specific target and can be excluded from further analysis. FxCop provides a default set of rules. You can create additional custom rules by using the FxCop SDK. Projects are used to specify the set of assemblies you want to analyze, the rules that are used to analyze the assemblies, the excluded messages, and the settings to customize the saved project or report file.

In This Section

Quick Start Guide to FxCop

Provides step by step procedures for accomplishing basic FxCop tasks.

Quick Start Guide to FxCopCmd

Describes the minimum command-line options that are used to run FxCopCmd.

FxCop Workflow

Outlines the general workflow for using FxCop and FxCopCmd as part of an integrated software development cycle.

Visual Basic C#
Visual C++
F#
JScript
Quick Start Guide to FxCop Send Feedback

This document provides a brief overview of the FxCop application. For a complete description of the FxCop user interface and features, see <u>Using FxCop</u>.

The FxCop application window displays the target assemblies and rules that are included in a project, and the messages that are generated when you perform an analysis. The window is divided into three major areas: the configuration pane, the messages pane, and the properties pane, as shown in the following illustration.

FxCop application window



Creating a Project and Running an Analysis

To analyze managed assemblies, FxCop requires the following information:

- One or more target assemblies to be analyzed.
- One or more rules. FxCop enables a default set of rules in new projects.

In addition, FxCop can be used with imported message sources, which typically specify a set of messages to exclude. To begin, specify the managed code assembly, which is either an executable file or a DLL, to analyze.

To create a project and run an FxCop analysis

- On the Start menu, click FxCop.
- or —

In the installation directory, double-click FxCop.exe.

- 1. On the Project menu, click Add Targets.
- 2. Locate the assembly you want to add, and then click Open.

⊠Note

The assembly is either an .exe or a .dll file.

3. Click Analyze to run the analysis.

The analysis report is displayed in the messages pane. If there are no messages, this pane will be empty.

Viewing Message Details

The messages pane displays a report of messages generated by the analysis.

To get detailed message information

• Choose a message and press ENTER.

— or —

Double-click a message.

- 1. Right-click a message to see the available message actions, which include:
 - Excluding a message from future reports.
 - Copying the selected message to the Clipboard.
 - Customizing the set of columns displayed in the messages pane.
 - Viewing properties of the message, which include the resolution, the target that generated the message with a link to the source code if available, rule details, a link to a more in-depth rule topic, and any custom notes.

Filtering Messages

Filtering is an optional feature that allows you to see a subset of the messages that are displayed in a report. After performing an analysis, you can filter the messages by clicking items in the configuration pane:

If you want to:	Click:
View only messages generated by the rules in a specific library.	A rule library you want to view.
View only messages generated by a specific rule.	A rule you want to view.
View only messages for a specific target and its child nodes.	A target.
Remove the current filter and view all the messages.	The FxCop project node at the root of the Targets tab or Rules tab tree view.

Excluding Messages

Occasionally, you might want to exclude certain messages from future analysis reports for a project. For example, if a design choice intentionally violates a rule or a rule generates an inaccurate message, the message can be excluded from future analysis. An excluded message contains the name of the user who created it, and optional notes about the reason for the exclusion.

To exclude messages

- 1. Run an analysis that generates messages.
- 2. Select the messages to exclude.

ĭTip

To select adjacent messages, click the first message, and then hold down SHIFT and click the last message. To select nonadjacent messages, click the first message, and then hold down CTRL and click additional messages.

- 3. Right-click any of the selected messages to display the shortcut menu.
- 4. Click Exclude.
- 5. In the Edit Note dialog box, type information relevant to the excluded messages.
- 6. Click OK.

Saving a Project or Report

Saving a project allows its settings to be reused in FxCop or imported into FxCopCmd. A report file, which contains a specified subset of messages, can be imported into FxCopCmd or into a new or different project in FxCop. Project files and message reports are saved in XML format.

To save a project or a report

- 1. On the Project menu, select Options.
- 2. Choose an XML style sheet to apply to the report.
- 3. Select the types of messages to save in the project and report.
- 4. On the File menu, select either Save Project As or Save Report As.
- 5. Locate the directory where you want to save the file.
- 6. In the File Name text box, type a name for the saved file.
- 7. Click Save.

Visual Basic
C#
Visual C++
F#
JScript
Quick Start Guide to FxCopCmd
Send Feedback

This document provides a brief overview of FxCopCmd, which is the commandline companion to the FxCop application. For a complete description of FxCopCmd, see <u>Using FxCopCmd</u>.

FxCopCmd is useful for analyzing assemblies in an automated environment but it does not support creating or configuring projects, or excluding messages. Although individual types and rules can be directly specified with the commandline options, the FxCop application provides easier and more finely detailed control.

Command-line options are used to specify the assemblies to be analyzed, the rules used to analyze the assemblies, and the location of the output file.

To specify the assemblies and rules, use one of the following:

- The /project option, which uses a project saved from the FxCop application.
- The /file and /rule options, which directly specify the assemblies and rules, respectively.

To specify the location of the output file, use one of the following:

- The /out option, which saves the results of an analysis in an XML file.
- The /console option, which displays the results in a console window or, if integrated with Visual Studio, in the Output window.

The following examples illustrate a minimum command line:

• FxCopCmd /p:SomeProject.FxCop /c

• FxCopCmd /f:SomeAssembly.dll /r:"C:\Rules Directory\SomeRules.dll" /o:OutputFile.xml

⊠Note

Option values that contain spaces must be enclosed in double quotation marks and most options can be specified by using only the first letter of the option.

Visual Basic C#
Visual C++
F#
JScript
FxCop Workflow
Send Feedback

FxCop should be used in an iterative fashion; it is designed to be a fully integrated part of the software development cycle. If you are starting a new software development effort, or are adding FxCop to an existing development effort, you will benefit most by first creating an FxCop project to store settings for the analysis. As you develop code and build it into assemblies for testing, the assemblies should be analyzed by FxCop to ensure that design and code defects are caught and corrected early in the development cycle. Items reported by FxCop should be fixed in the source code or, in the case of false positives, excluded from future reports. After the criteria, specifically the targets to analyze, rules, and excluded messages, for your software development project have been saved in an FxCop project, you can easily integrate code analysis into your software build process by adding a step to your build script to execute the command–line tool.

The following sequence of steps describes the general procedure for using FxCop:

To create a new project

- 1. Compile the assemblies.
- 2. Start the FxCop application.
- 3. Add the targets (assemblies) to be analyzed to the project.
- 4. Review the rules automatically loaded by FxCop and clear the selection of any rules that should not be applied to the targets during analysis.

To analyze the assemblies and review messages

- 1. Analyze the assemblies.
- 2. Review the reported messages and determine the disposition for each:
 - Code defect fix the source code.
 - Message is reporting an intentional variance from the rule exclude the message.
 - Rule is generating a false positive exclude the message.
- 3. Exclude items as appropriate.
- 4. Correct any code defects detected by FxCop.
- 5. Rebuild your assemblies.

Repeat steps 6 through 9 until the project file is configured to detect only the kinds of items that you want to fix, and reports no items for your current build. You are now ready to add the command-line tool, FxCopCmd, to your build process to automatically generate analysis reports each time you build your assemblies.

To add FxCopCmd to the build process

- 1. Save the project.
- 2. In your build script, call FxCopCmd using the /project option.
- 3. Review the analysis report output.
- 4. If new messages need to be excluded, or other changes to the project are required, use the FxCop application to update your project and save your changes.
- 5. Correct any code defects detected by FxCopCmd.

Visual Basic C#
Visual C++
F#
JScript
Using FxCop
Send Feedback

This section contains detailed information about FxCop, the fully featured application that analyzes managed code assemblies. For a quick introduction to FxCop, see <u>Quick Start Guide to FxCop</u>. For information about the companion command-line tool, see <u>Using FxCopCmd</u>.

In This Section

Overview of FxCop

Provides a brief overview of FxCop.

Performing an Analysis

Explains how to perform an analysis, cancel an analysis, and view a summary of analysis results.

Managing FxCop Projects

Explains how to create, open, save, and import messages into a project, and how to set project options and application preferences.

Managing Target Assemblies

Explains how to add, ignore, and remove targets in a project, specify a target location using environment variables, and view target MSIL.

Managing Rules

Provides information about the default rule libraries loaded with a new project, explains how to add, group, ignore, and remove rules in a project, and discusses the information stored in a rule.

Managing Messages

Discusses the information provided by a message, and explains how to filter, view, exclude, and save messages

Visual Basic C#
Visual C++
F#
JScript
Overview of FxCop
Send Feedback

FxCop analyzes programming elements in managed assemblies, called targets, and provides an informational report that contains messages about the targets, Messages include suggestions about how to improve the source code used to generate them. FxCop represents the checks it performs during an analysis as rules. A rule is managed code that can analyze targets and return a message about its findings. Rule messages identify any relevant programming and design issues and, when possible, supply information about how to fix the target.

Using FxCop, you can perform the following tasks:

- Control which rules are applied to targets.
- Exclude a rule message from future reports.
- Apply style sheets to FxCop reports.
- Filter and save messages.
- Save and reuse application settings in FxCop projects.

■Working in the FxCop Application Window

The FxCop application window displays the targets and rules included in a project, and the messages that are generated when an analysis is performed. The window is divided into three major areas: the configuration pane on the left, the messages pane on the right, and the properties pane at the bottom, as shown in the following screen shot.

FxCop application window



The configuration pane displays targets and rules for the current project in a tree view.

The messages pane displays a report, customizable by column, about the messages generated from an analysis. You can filter the displayed messages by selecting specific targets and rules in the configuration pane.

The properties pane contains two tabs. The Output tab displays informational, warning, and error messages when the TraceGeneral <u>System.Diagnostics.TraceSwitch</u> is enabled (the default) in the FxCop.exe.config configuration file. When a message is displayed in the Output tab, you can right-click in the pane to display a shortcut menu with the following choices:

• Clear. This erases all information from the Output pane.

- Copy Selected. Copies the selected information from the Output pane to the Clipboard.
- Copy All. Copies all the information from the Output pane to the Clipboard.

The Properties tab displays detailed target, rule, or message information when an item is selected in the configuration or messages pane.

Visual Basic
C#
Visual C++
F#
JScript
Performing an Analysis
Send Feedback

To analyze assemblies, FxCop requires the following information:

- One or more target assemblies to be analyzed.
- One or more rules. FxCop enables a default set of rules in new projects.

To start, specify the managed code assembly to analyze. The assembly must be either an executable file or a DLL.

To specify an assembly and run an FxCop analysis

• On the Start menu, click FxCop.

— or —

In the installation directory, double-click FxCop.exe.

- 1. On the Project menu, click Add Targets.
- 2. Locate the assembly you want to add, and then click Open.

⊠Note

The assembly is either an .exe or a .dll file.

- 3. Start the analysis by using one of the following methods:
 - Click the Analyze button on the toolbar.
 - On the Project menu, select Analyze.

• Press the F5 key.

When analysis starts, the Analyzing with engine dialog box appears. This shows a progress meter and a Cancel button. The label of the dialog box includes the name of the analysis engine that is being used. When analysis is finished, the analysis report is displayed in the messages pane. If there are no messages, this pane is empty.

Canceling an Analysis

Analysis can be canceled at any time, but analysis results might be compromised.

To cancel an analysis

• In the Analyzing with engine dialog box, click Cancel.

The <u>Analysis Summary</u> dialog box appears. It displays analysis statistics and error messages.

A message is always in one of the following states: active, excluded, or absent. For more information, see "Message States" in the <u>Managing Messages</u> topic. When an analysis starts, all messages in the Active state are changed to the Absent state; individual messages are then changed back to Active as they are re-encountered. Any messages that are not re-encountered before analysis is canceled remain in the Absent state. This state compromises the analysis results. Excluded messages are unaffected.

■Viewing a Summary of Analysis Results

During an analysis, FxCop gathers statistics and catches exceptions thrown by rules. Rule exceptions do not prevent an analysis from continuing, unless the total number of exceptions exceeds the limit specified for the project; however, the message report is usually incomplete. For information about how to set the exception limit, see "Setting Project Options" in the

Managing FxCop Projects topic.

If an analysis generates any errors, the Analysis Summary dialog box is displayed after the analysis is completed. At any other time, select the Analysis Summary command on the Project menu to manually view a summary of the most recent analysis. The following screen shot shows a sample analysis report as it appears in the Analysis Summary dialog box.

Analysis Statistics Messages: 9 (9 new) 10/19/2004 3:28:39 PM Started: Time: 00:00:47 Checks: 295 -DX 9 Issues: 0 Exclusions (In Project): Exclusions (In Source): 0 Errors The analysis engine completed successfully without errors. Details All rules completed successfully without errors. Details

Analysis Summary dialog box

The report displays the number of messages generated, the analysis start time, and the duration of the analysis. The summary report also indicates any problems FxCop encountered while performing the analysis.

Rule Errors

To display the Rule Error(s) dialog box, click Details at the bottom of the Analysis Summary dialog box.

The following illustration shows the Rule Error(s) dialog box. The top pane displays the rules that raised exceptions while the target was analyzed. Expand a rule to display the list of errors generated by that rule. If you select an error, the bottom pane shows a tree view of the target that the rule was analyzing when it generated the error. If you double-click the target, the corresponding target is highlighted in the Targets tree of the Configuration pane.

Rule Error(s) dialog box



Analysis Errors

To display the Analysis Error(s) dialog box, click Details at the top of the Analysis Summary dialog box.

The Analysis Error(s) dialog box is shown in the following illustration.

Analysis Error(s) dialog box



The most common analysis problems are as follows:

- The current project has no targets selected.
- The current project has no rules selected.
- An assembly referenced by a target could not be located for analysis.

The vast majority of analysis errors are caused by a missing assembly. During analysis, the types referenced by an assembly must be available to FxCop. When FxCop cannot locate a type, an exception is reported for the assembly that contains the type; its message is "File or assembly name SomeAssembly, or one of its dependencies, was not found." To fix this problem, add the missing assembly to the directory that contains the assembly being analyzed, or add it to the global assembly cache. For information about the global assembly cache, see the .NET Framework SDK documentation, or visit the <u>Microsoft Developers</u> <u>Network Library</u> on the Internet.

If all managed assemblies are apparently accounted for and this error remains, a missing native dependency might be the cause. This is most common for managed code extensions for C++ assemblies.

For other errors, such as those accompanied by the message "Type SomeType

could not be loaded.", use the Assembly Binding Log Viewer (Fuslogvw.exe) that is included with the .NET Framework SDK to determine the cause of the problem. This tool provides detailed information about assembly loading failures.

Visual Basic C#
Visual C++
F#
JScript
Managing FxCop Projects
Send Feedback

A project specifies a set of targets and rules used by FxCop to perform an analysis and contains a report of the messages from the latest analysis. Each message is associated with a specific target and a specific rule. Projects are saved in files that have the extension .FxCop. Saved projects can be reused by the FxCop application (FxCop.exe) and by the command–line tool (FxCopCmd.exe). For information about how to specify a project to the command–line tool, see the /**project** option in <u>Command-Line Tool Options</u>.

In This Topic

Creating New Projects

Opening Existing Projects

Saving Projects

Importing Messages into a Project

Setting Project Options

Setting Preferences and Project Defaults

Creating New Projects

When a new project is created, FxCop does the following:

- Closes the current project if one is open.
- Resets all project options to their default values. For more information, see

Setting Project Options.

• Loads any rule libraries found in the rules directory specified by your application settings. The default rules directory is the rules subdirectory under the application startup directory, which is where FxCop.exe is located. For example, if you have not specified application settings, and FxCop.exe is installed in the directory C:\Tools\FxCop, then the C:\Tools\FxCop\Rules directory is searched for rules to add to the new project.

To create a new FxCop project

• On the File menu, select New Project.

- or -

• Press CTRL+N.

Opening Existing Projects

The simplest way to open an FxCop project is to double-click an FxCop project file in Microsoft Windows Explorer. This starts FxCop and loads the project. You can also open a project from within the FxCop application itself.

To open an existing project from within FxCop

- 1. On the File menu, choose Open Project.
- 2. Locate the project file to open.
- 3. Click Open.

FxCop loads the selected project.

⊠Note

FxCop loads only those items named explicitly in the opened project file. Unlike creating a new project, opening an existing project does not cause default rule libraries to be loaded.

Project Load Errors

If a project cannot be completely loaded, an Error dialog box is displayed. The following errors keep a project from completely loading:

- The project file is corrupted.
- A rule assembly specified in the project file cannot be found or is corrupted.
- A target assembly specified in the project file cannot be found or is corrupted and has an entry in the project report from a previous analysis.
- The version number of the project file is later than the version number of FxCop.

A target assembly specified in the project file that cannot be found or that is corrupted and does not have any entries in the project report is loaded, but causes an error on analysis.

If the version number of the project file is less than 1.312, the Project Migration Wizard is invoked.

If a project loads but all targets cannot be added or all messages cannot be imported, the Error(s) Loading Project dialog box appears, as shown in the following screen shot.

Error(s) Loading Project dialog box



This results when a target or a rule listed in the project report cannot be found. Double-click the "Could not import all messages." error message to display the Warning dialog box, which displays the targets associated with the unimported messages in a tree view, as shown in the following screen shot. Click the target to display the messages that were not loaded.

Warning dialog box

The following messag imported because a s rule either no longer o enabled for analysis. view message XML th	pes could not be specified target or exists or is not Click on items to nat failed to import.
	The following rules no longer exist: AssembliesShouldHaveValidStrongNames MarkAssembliesWithAssemblyVersion MarkAssembliesWithClsCompliant MarkAssembliesWithComVisible
C:\Test\Test.FxCop	lll e.dll rary ostractClass y

Saving Projects

To retain your current project settings, FxCop projects must be saved before you exit the application or open a different project.

To save a new project

- 1. On the File menu, choose Save Project or Save Project As.
- 2. Locate the directory where the project is to be saved.
- 3. In the File name box, type a name for the project file.
- 4. Click Save.

To save an existing project under the same name

• On the File menu, choose Save Project.

FxCop overwrites the previous version.

To save an existing project under a new name

- 1. On the File menu, choose Save Project As.
- 2. Locate the directory where the project is to be saved.
- 3. In the File name box, type a name for the project file.
- 4. Click Save.

Importing Messages into a Project

You can import multiple reports from previously saved analyses. The contents of a report are treated as an extension of the project contents; when an imported message applies to a target in the project, that message is added to the project. For information about how to save a report, see

<u>Save Messages</u> and the "Saving Message Reports" section in the topic <u>Managing</u> <u>Messages</u>.

To import the messages in a saved report

- 1. On the File menu, select Import Report.
- 2. Locate the location of the report and select it.
- 3. Click OK.

A message is always in one of the following states: active, excluded, or absent. For a more information, see "Message States" in the Managing Messages topic. If the same message appears in multiple sources and has a different message state in each, for example, the message is in both the project and an imported report, or is in more than one imported report, the duplicates are resolved as follows:

- If a message is excluded in any input, the message is excluded in the project.
- If a message is active in any input, and is not excluded in any input, the message is active in the project.
- If a message is neither active nor excluded in any input, and is absent in any input, the message is absent in the project.

Import Errors

If an error occurs while importing a report because of a corrupted report file that

is not a well-formed XML document, an error dialog box is displayed with a description and the line number of the error.

A report can be imported but not all messages can be mapped because of the following reasons:

- A specified target or rule does not exist in the project.
- A specified target or rule is disabled in the project.

If any messages are not mapped, a warning message box appears that asks whether you want additional information about the unimported messages. Click Yes and the Warning dialog box appears and displays the target that the unimported messages are associated with in a tree view. Double-click the target to display the number of messages that were not mapped, and the XML from the report file for the unmapped messages.

Setting Project Options

Use project options to enable or disable FxCop features for the current project. Use

tool settings to define default options for new projects.

To edit current project options

• On the Project menu, select Options to display the Project Options dialog box.

- or -

• Right-click the top node of the Targets or the Rules tab in the configuration pane and select Properties.

Standard Tab

The following screen shot shows the Standard tab of the Project Options dialog box.

[Needs art: Standard tab of Project Options dialog box]

Standard tab of Project Options dialog box
Project Options		
Standard Saving & Co	mpression Spelling & Analysis	
Project Name: Benort Stule sheet:	My FxCop Project	
Shared Project	Apply Style sheet Attempt source file lookup	
Save Messages	es against legacy code for which the fix is a breaking change	
Report: Ac	tive All Active Called Absent	
Report options also dictate console output when analyzing this project with FxCopCmd.exe		
Suspend analysis a	fter 10 exceptions	
Disable rules after	1 exceptions	
	OK Cancel	

Project Name

This setting allows you to associate a name with your project. The name appears on the application window and is saved with your project settings.

Report Stylesheet and Apply Stylesheet

When a message report is saved, the style sheet URI specified by Report Stylesheet is included in the XML as an XML style sheet processing instruction. Browsers that recognize the instruction will display the XML using the style sheet. By default, the FxCopReport.xsl included with FxCop is specified. To omit the processing instruction, clear the Report Stylesheet text box. Select Apply Stylesheet to have the style sheet applied to the report and the transformed output saved.

Shared Project

This option allows an FxCop project file to be shared by multiple build environments with configurations that are identical except for drive mappings and FxCop installation directories. When this option is selected, project files store file locations using relative paths. Without this option applied, projects save file locations as absolute paths. In a shared project, all items, including assemblies, rules, and excluded messages, must reside in one of the following locations:

- On the same drive as the project file.
- In the directory where FxCop is installed.
- In a subdirectory of the FxCop install directory.
- On a UNC share.

For example, if a project is saved in 'C:\Team\Product X', the project's items can be located in any directory on the C drive, or in or below the FxCop installation directory. You could not, for example, add assemblies from 'D:\TestSuites\Product X', because it is not on the same drive. The project file is not required to be on the same drive as the FxCop installation directory.

The Shared Project option is disabled for new projects. If you enable this option and your project contains items that are not in valid locations, you will not be able to save your project until you remove the items.

Attempt source file lookup

This option determines whether FxCop tries to look up the source code location of the offending programming element when a message is generated. For more information, see the "Viewing Message Details" section in the topic

<u>Managing Messages</u>. Disabling source file lookup can greatly increase performance.

Save Messages

These settings allow you to specify which messages are saved in the project and in reports generated by the project. When FxCopCmd is run with the /project option, the output (console or report file) is based on the report settings in the project file. If these settings are not configured to save messages in the report, the output of FxCopCmd will not contain any messages even though messages are generated.

You can save messages with any combination of the following states:

Active - messages that were generated during the most recent analysis. You can save all active messages, or only those messages that first appeared during the most recent analysis, by choosing All Active or New Only, respectively, from the drop-down list.

Excluded - messages that have been marked as such in the current project, or in an imported report.

Absent - messages that were generated in a previous analysis, but were not generated in the most recent analysis.

Suspend analysis after and Disable rules after

These settings specify a limit on the number of exceptions encountered before a rule is disabled or analysis is canceled.

A disabled rule is excluded from the remaining analysis. The default limit is ten exceptions.

The default limit for canceling analysis is one hundred exceptions. For a discussion of the results of canceling an analysis, see

Performing an Analysis.

Clear the text box or specify -1 to allow an unlimited number of exceptions.

For information about how to view the exceptions, see the "Viewing a Summary of Analysis Results" section in the topic <u>Performing an Analysis</u>.

Saving & Compression Tab

The following screen shot shows the Saving & Compression tab of the Project Options dialog box.

Saving & Compression tab of Project Options dialog box

Project Options	
Standard Saving & Com	pression Spelling & Analysis
Project Save O Minimize save time O Minimize file differen	ces between saves
Project File	
Save Rules By:	File Name 💌
Compress project	file
The following load o which will greatly inc	otions can only be set by turning off project compression, rease the size of the project file.
 ✓ Include newly intre ✓ Include newly intre 	oduced targets in analysis by default oduced rules in analysis by default
	OK Cancel

Project Save

This setting determines whether FxCop minimizes the time it takes to save a project file (the default), or minimizes the difference between project files saved after repeated analyses. To minimize file differences, the "Minimize file differences between saves" option must be set during all the file saves. To minimize file differences, the output is saved in alphabetical order and the last analysis time is not saved.

Save Rules By

This setting determines which group of categories the rules are saved by in the project file. Consequently, this determines how they will appear the next time that the project is loaded, which might be different from their appearance in the current Rules pane. For more information, see the "Grouping Rules" section in the topic Managing Rules.

Compress Project File

This setting allows you to specify whether the project file is compressed when it is saved. To compress a project, FxCop saves information about the target and

rule hierarchies only to the depth that it is necessary to reproduce the check states for existing targets and rules. For example, when all nodes under a parent node are selected or none of the child nodes are selected, FxCop saves only the parent node and the check state. When opening the compressed project, FxCop assumes the check state applies to all child nodes.

FxCop compresses project files to minimize their size; however, compressing a file makes it difficult for FxCop to detect additions to the rule libraries and targets that are loaded for analysis. By default, FxCop sets the state of new rules and targets according to the state of the closest ancestor node that exists in the project. For example, if a target assembly contains new types, and the project is set to analyze only certain namespaces in the assembly, the new types will be analyzed only if their namespaces are among those selected. Depending on your project and workflow, this behavior might not be correct. You can use the Compress Project File option to disable project file compression, and to specify whether FxCop should enable new rules and targets.

Spelling & Analysis Tab

The following screen shot shows the Spelling & Analysis tab of the Project Options dialog box.

Spelling & Analysis tab of Project Options dialog box

📱 Proje	ct Options		×
Standard	Saving & Compre	ssion Spelling & Analysis	
Diction	ary Locale:	English (U.S.) : "en-us"	
Control	Flow Analysis:	Use default engine setting	
<u>Bun all overridable rules against all targets</u>			
🗹 Ena	ble multithreading o	during load phase.	
Enable multithreading during analysis phase.			
Search <u>G</u> lobal Assembly Cache for missing references.			
		OK Cance	•

Dictionary Locale

This setting determines the language that is used for the spelling rules. The Microsoft Office Proofing Tools spelling checker for the specified language must be installed. This setting overrides the setting in <u>Spelling Options</u>.

Control Flow Analysis

To enable control flow analysis, select True from the drop-down dialog box; select False to disable it. When Use default engine setting is selected, control flow analysis depends on the Enable Control Flow Analysis setting, which is true by default.

Run all overridable rules against all targets

Most FxCop rules check only targets (programming elements) that are externally visible, that is, a target that can be accessed outside its containing assembly. In C#, this refers to types and members that have public, protected, or protected internal accessibility. However, many rules that typically check only externally visible targets allow this behavior to be overridden. When this setting is enabled, these rules check targets of all visibilities.

■Setting Preferences and Project Defaults

Application settings control FxCop's appearance and behavior, and the default options for new projects.

To edit application settings

• On the Tools menu, select Settings to display the Settings dialog box.

Settings are divided into four categories, which appear on the tabs in the Settings dialog box:

- Preferences
- Fonts & Colors
- Analysis Engines
- Project Defaults

Preferences Tab

The following screen shot shows the Preferences tab of the Settings dialog box.

Preferences tab of the Settings dialog box

Settings	×
Preferences	
General Preferences	
Check for updates to FxCop on startup	
Filter messages when selecting targets	
	1
Source Code Editor	
Launch in alternate application	
Command: Instanted ava	
Commanda. Inotepad.exe	
Arguments: [\$(File)	Л
OK Cancel]

Check for updates to FxCop on startup

When this option is enabled, on startup, FxCop queries its Web site for an updated version. The result of the query is output to the Properties pane. No information other than the version number is transmitted and no files are downloaded.

Filtering messages when you select targets or rules

This setting enables or disables message filtering. For more information, see the "Filtering Messages" section in the topic

Managing Messages.

Source Code Editor

If Program Database (PDB) information is available for an analyzed assembly, FxCop can locate and open the source code used to build the assembly. This feature is accessed from the Source hyperlink on the Message Details dialog box that is described in the "Viewing Message Details" section of the topic Managing FxCop Projects or from the Properties tab of the properties pane when the message is selected.

By default, FxCop tries to open the source code file by using Microsoft Visual Studio .NET. To use a different code editor, select the Launch in an alternative application option. The Command and Arguments text boxes are pre-filled to use Microsoft Notepad. Specify a different editor program using the Command box. You can use environment variables when you specify this setting. In the Arguments box, specify any arguments to be passed to the program specified in Command. The argument setting can refer to the file to be opened as \$(File), and the line number to locate as \$(Line).

Fonts & Colors Tab

Use this tab to customize the fonts and colors displayed in the configuration, messages, and properties panes.

To change the font that is used in the configuration pane, click Tree in the Font section. To change the font that is used in the messages pane, click List in the Font section. To change the font that is used in the properties pane, click Output in the Font section. All options display a standard Font dialog box you can use to format the font in the corresponding pane.

In addition to setting the font that is used for report messages, you can also assign a color to each message level. To change the default color that is used for a message level, select the message level from the drop-down list and then click the MessageLevel. Use the Color dialog box that appears to choose the color for the selected message level.

Analysis Engines Tab

The following screen shot shows the Analysis Engines tab of the Settings dialog box and the Introspection Engine Settings dialog box, which is displayed when you click Settings.

Analysis Engines tab of the Settings dialog box

🚦 Settir	ngs 🔀
	Analysis Engines
Name	
✓ Intro	ospection
Engir	ne Information
File	ypes:
Disabl	e Settings
	🐺 Introspection Engine Settings 🛛 🗖 🗙
	Number of Analysis Threads: 2
	Enable Control Flow Analysis
	OK Cancel

The current version of FxCop uses only one engine, the Introspection engine. If this check box is clear, analysis will not occur.

Select the name Introspection to enable Engine Information and the Settings button. Engine Information provides a brief description of the engine, and lists the types of files the engine can load.

The Settings button displays the Introspection Engine Settings dialog box, which contains the following two settings:

Number of Analysis Threads

By default, the Number of Analysis Threads is set to the number of processors on the analyzing computer. To remove any possibility of a deadlock, which would prevent analysis from completing, set the number of threads to 1; however, this will generally increase analysis time.

Enable Control Flow Analysis

Control flow analysis is an analysis technique that can construct all possible code paths and all possible object manipulations. An example of this technique would be following a string through a method flow graph. This type of analysis is capable of looking at a different set of problems than static analysis techniques but slows down analysis time considerably.

Disabling this option prevents several default FxCop rules from running. When an analysis is performed with this option disabled, a message is displayed in the Output tab of the properties pane. The message is not erased if this option is subsequently enabled and another analysis is performed.

Project Defaults Tab

Use this tab to choose default settings for new projects. The following screen shot shows the Project Defaults tab of the Settings dialog box.

Settings	
Project Defaults	
Rules Directory:	
Report Stylesheet:	
Save Messages	
Project File	
(OK Cancel

Project Defaults tab of the Settings dialog box

Rules Directory

All rule libraries in this directory are included in a new project. You can use environment variables when specifying this setting.

The following settings are identical to those in the

<u>Project Options</u> dialog box except that they are applied to new projects, not the current project:

- Report Stylesheet
- Shared Project
- Apply Stylesheet
- Attempt source file lookup
- Do not fire messages against legacy code for which the fix is a breaking change
- Save Messages
- Compress project file
- Suspend analysis after
- Disable rules after

Project Defaults (cont.) Tab

Spelling Options

This setting determines the language that is used for the spelling rules. The Microsoft Office Proofing Tools spelling checker for the specified language must be installed.

Send <u>feedback</u> on this topic to Microsoft.

Visual Basic C#
Visual C++
F#
JScript
Managing Target Assemblies
Send Feedback

FxCop analyzes managed code executables and DLLs, called target assemblies. At least one target assembly must be added to the current project before an analysis can be run; multiple target assemblies can be included in a single analysis. A target for analysis can be an assembly, module, namespace, type, type member, parameter, or resource.

You can exclude targets from analysis temporarily, or you can remove a target assembly from a project entirely. If a target is excluded from analysis, it remains on the Targets tab in the configuration pane, but it is not included in analysis reports. If you remove a target assembly from a project, it no longer appears on the Targets tab and any messages associated with the target are deleted. Once a target assembly is removed, it must be added to the project again before it can be included in an analysis.

In This Section

Adding Target Assemblies

Ignoring Targets During Analysis

Removing Target Assemblies from a Project

Specifying a Location for a Target Assembly using Environment Variables

Viewing Metadata and MSIL

Adding Target Assemblies

One or more target assemblies must be specified for analysis. First, open a new or existing project, and then add target assemblies (.exe or .dll files) using either of the following procedures.

To drag a target assembly into the current project

- 1. Click the Targets tab in the configuration pane.
- 2. From a file viewer, such as Microsoft® Windows Explorer, drag the file or files to be added into the configuration pane.

To add a target assembly to the current project

- 1. From the Project menu, select Add Targets.
- 2. Browse to the target to add.
- 3. Click Open.

FxCop adds the target assembly to the current project with all its types and type members enabled. If you do not want to include all the target assembly's types and members, you must disable them. For more information, see

Ignoring Targets During Analysis.

Target Load Errors

If any errors are encountered, the Error(s) Loading Assemblies dialog box appears, listing the assemblies that could not be loaded and the reasons why. Double-click one of the list items to display the AssemblyLoadException Details dialog box, which provides the information shown in the following table, if available.

Detail item	Description
FileName	The name of the assembly that could not be loaded.
HelpLink	A link to the Help file associated with the exception.
InnerException	The inner exception that is contained in the exception.
InnerExceptions	All inner exceptions. When this cell is selected, ellipses appear in the cell. Clicking the ellipses displays the Exception Collection Editor dialog box. This dialog box is not part of the functionality of FxCop.
Message	The reason the assembly did not load.
Source	The FxCop assembly that generated the exception.

StackTrace	The stack trace associated with the exception.
TargetSite	The FxCop member that generated the exception.

Delay-Signed Assemblies

When you try to add a delay-signed assembly, FxCop reports the following error:

Strong name validation failed for assembly 'SomeAssemblyName'

The .NET Framework cannot verify the signature of a delay-signed assembly, and therefore considers it unsafe to load. A delay-signed assembly will not load unless you disable signature verification for the assembly by using the Strong Name command-line tool (Sn.exe) that is included with the .NET Framework. To disable verification of the delay-signed assembly, type the following at the command line:

SN -Vr SomeAssemblyName

This allows the assembly named SomeAssemblyName to skip verification. After an assembly is registered in this manner, the common language runtime will not try to verify the signature. Currently running instances of FxCop will not detect the new registration; you must exit the application and start it again to load the assembly.

⊠Note

Disabling verification of an assembly creates a security vulnerability. To reenable verification of the assembly, type the following at the command line:

SN -Vu SomeAssemblyName

For more information, see the following topics on MSDN:

- <u>Delay Signing an Assembly</u>
- <u>Strong Name Tool</u>

Ignoring Targets During Analysis

You ignore a target assembly, type, or type member during analysis by clearing the check box next to the target name, which also clears the selection from all child nodes. An ignored target remains on the Targets tab in the configuration pane and messages relevant to it remain in the messages pane and are written to any saved reports. On subsequent analyses, the targets that are not selected are ignored and therefore do not produce any messages. For information about how to permanently remove a target assembly from the Assemblies list, see

Removing Target Assemblies from a Project.

To ignore a target and all its child nodes

- 1. Click the Targets tab in the configuration pane.
- 2. Expand the tree view, if necessary.
- 3. Clear the check box next to the target name.

To include an ignored target and all its child nodes, select the check box again.

⊠Note

Any changes that you made to the Targets tab are not retained in the project file until you save the project.

For information about how to remove a target assembly from the Assemblies list, see <u>Removing Target Assemblies from a Project</u>.

Removing Target Assemblies from a Project

You can ignore a target assembly, type, or type member during analysis, or you can remove a target assembly from the current project entirely. When you remove an assembly from the current project, it no longer appears on the Targets tab in the configuration pane. It must be added to the project again to be included in an analysis. If an assembly is removed, any report messages generated for targets within the assembly are also removed from the project.

To remove an assembly from the current project

- 1. Click the Targets tab in the configuration pane.
- 2. Select the target assembly to be removed by clicking its name.
- 3. From the Edit menu, choose Remove File.

You can also right-click the assembly to be removed, and choose Remove.

For information about how to temporarily ignore an assembly, see

Ignoring Targets During Analysis.

■Specifying a Location for a Target Assembly using Environment Variables

You can specify the location a target assembly is subsequently loaded from using a path that includes environment variables. FxCop reads environment variables when the application is started. Any changes to environment variables are ignored until FxCop is shut down and restarted.

To specify the location using environment variables

- 1. Click the Targets tab in the configuration pane.
- 2. Right-click the target assembly and select Properties.
- 3. Type the location in the Save Name box.

⊠Note

Using environment variables for the location of an assembly can result in the assembly failing to load if the environment variables are not initialized. This might require manually editing the project file to correct. As an alternative to environment variables, consider using a Shared Project, which can be found in the "Shared Project" section of the topic

Managing FxCop Projects.

Wiewing Metadata and MSIL

Metadata and Microsoft intermediate language (MSIL) can be displayed for targets that are loaded in FxCop.

To view metadata

- 1. Click the Targets tab in the configuration pane.
- 2. Expand the assembly and subnodes.
- 3. Select the target element.

The metadata appears in the properties pane.

1. Right-click the target element, choose View, and then choose IL.

A window appears that displays the MSIL.

Send

<u>feedback</u> on this topic to Microsoft.

Visual Basic C#
Visual C++
F#
JScript
Managing Rules
Send Feedback

Rules, which are contained in rule libraries, determine the issues FxCop looks for during an analysis. FxCop includes a default set of rule libraries. Each rule is assigned a level that indicates the importance of the issue detected by the rule, and a certainty that estimates the probability of an issue being detected correctly.

In This Section

Default Rule Libraries

Adding Rules to a Project

Grouping Rules

Ignoring Rules During Analysis

Removing Rule Libraries from a Project

Viewing Rule Details

Specifying a Location for a Rule Assembly using Environment Variables

Default Rule Libraries

FxCop includes the following rule libraries, based on the .NET Framework Design Guidelines that are loaded by default when a new project is created:

- COM rules that detect COM Interop issues.
- Design rules that detect potential design flaws. These coding errors typically do not affect the execution of your code.
- Globalization rules that detect missing or incorrect usage of information related to globalization and localization.
- Naming rules that detect incorrect casing, cross language keyword collisions, and other issues related to the names of types, members, parameters, namespaces, and assemblies.
- Performance rules that detect elements in your assemblies that will degrade performance.
- Security rules that detect programming elements that leave your assemblies vulnerable to malicious users or code.
- Usage rules that detect potential flaws in your assemblies that can affect code execution.

You can view the currently loaded rule libraries from the Rules tab in the configuration pane. Double-click the library name to see the rules contained in a library. Double-click the rule name to display the Rule Details dialog box, which displays the rule's detailed information.

Adding Rules to a Project

To set the scope of an analysis, one or more rule libraries must be included in a project. For new projects, all rule libraries located in the Rules Directory specified in the New Project Defaults tab (described in the

Managing FxCop Projects topic) are automatically loaded. For existing projects, the settings for rule libraries are saved as part of the project and are loaded together with the other project items. When you open an existing project, only the saved rules are loaded; libraries in the rules directory specified in your application settings are not automatically loaded.

To drag a rule library into the current project

- 1. Click the Rules tab in the configuration pane.
- 2. In a file viewer, such as Windows Explorer, drag the file or files to be added into the configuration pane.

To add a rule library to the current project

- 1. On the Project menu, select Add Rules.
- 2. Locate the rule library to add.
- 3. Click Open.

FxCop adds the library to the current project.

Grouping Rules

Rules can be grouped by file name, message level, or breaking change. This allows messages to be filtered by these groupings, and the rules in a specific group to be enabled or disabled as a unit. By default, rules are grouped by the file name of the assembly that contains the rule. "Breaking change" means that an assembly that has a dependency on one version of a library will no longer work with a new version of the library that has been changed to fix a violation of the rule.

To change how rules are grouped

- 1. Right-click a rule or group in the Rules pane.
- 2. Select Group By.
- 3. Choose one of the groups from the menu.

Ignoring Rules During Analysis

You can ignore a rule or an entire rule library during analysis. Ignored items remain in the configuration pane but are not used for further analysis of your code.

To exclude a rule or rule library from analysis

- 1. Click the Rules tab.
- 2. If you want to ignore a rule, expand the library that contains that rule.
- 3. Clear the check box next to the rule or rule library to be ignored.

To include an ignored rule or rule library, select the check box next to it.

⊠Note

Any changes that you made to the rules are not retained in the project file until you save the project.

For information about how to remove rule libraries from a project, see

Removing Rule Libraries from a Project.

Removing Rule Libraries from a Project

When you remove a rule library from the current project, it no longer appears in the configuration pane and must be added to the project again before its rules can be included in an analysis. If a rule library is removed, all messages generated by its contained rules are also removed from the project.

To remove a rule library from the current project

- 1. Click the Rules tab.
- 2. Select the rule library to be removed.
- 3. On the Edit menu, choose Remove File.

You can also select the rule library to be removed and press ENTER, or rightclick the rule library and choose Remove.

⊠Note

Any changes that you made to the rule libraries are not retained in the project file until you save the project.

For information about how to temporarily ignore rule libraries during analysis, see

Ignoring Rules During Analysis.

Viewing Rule Details

You can access the currently loaded rule libraries from the Rules tab of the configuration pane. To see the rules contained in a rule library, double-click the rule library name. To see detailed information about a rule, double-click the rule. Alternatively, you can right-click the rule and select Properties on the shortcut menu. The following screen shots show an example of the Rule Details dialog box.

Rule Details tab of the Rule Details dialog box



Support tab of the Rule Details dialog box

Message	Details 🔲 🗖 🔀
Issues Rul	e Details Message Rule Support Notes
Type Name	: ReviewVisibleEventHandlers
Owner:	
E-mail:	[none]
URL:	http://www.gotdotnet.com/team/fxcop/docs/rules.aspx? version=1.35url=/Security/ReviewVisibleEventHandlers.html
- File Inform	nation
Name:	Security Rules
Version:	1.35.0.0
File:	c:\program files\microsoft fxcop 1.35\rules\securityrules.dll
	OK Exclude Previous Next

The following table describes the information stored in a rule and displayed in the Rule Details dialog box.

Rule element	Description
Rule Details tab	
Name	The rule's friendly name; typically, a single sentence that describes the rule.
Level	The level of importance associated with the rule. (The severity of the problem reported by the rule. For a discussion of the levels, see

Level and Certainty.)

Certainty

The confidence associated with the rule.

File

The name of the library that contains the rule.

Category

The category of the rule.

CheckId

The unique identifier for the rule.

Description

A short description that details the issues and possible causes of errors identified by the rule.

Support tab

Type Name

A unique identifier for the rule.

Group Owner

The organization that authored the rule.

Owner

The individual responsible for the rule.

E-mail

The e-mail address for support and bug reports related to the rule.

URL

A link to more in-depth information pertaining to the rule.

File Name

The name of the library that contains the rule.

File Version

The version of the library that contains the rule.

File Path

The full path and name of the library that contains the rule.

Not visible

Resolution

Instructions and information about how to fix the issues that are identified by the rule. This information is stored in a rule but exposed through the Message Details dialog box. For more information, see the Viewing Message Details section of the Managing FxCop Projects topic.

Level and Certainty

Each rule is assigned a level that indicates the importance of the issue detected by the rule and a certainty that estimates the probability of an issue being detected correctly.

Three major factors determine the level assigned to a rule:

- The visibility of the detected issue.
- The probability that the detected issue will adversely affect application

behavior.

• The risk associated with not fixing the issue.

Rules are assigned one of five importance levels:

- Critical ErrorMessages at this level reflect issues that are highly visible, that prevent code from operating correctly in common scenarios, or both. Critical error messages should be resolved first, and should be excluded only after carefully assessing the affect of ignoring the error.
- ErrorIssues reported at this level have less affect on usability and behavior than critical errors, but should not be excluded without careful assessment.
- Critical WarningMessages at this level reflect issues that generally have little or no adverse effect on code behavior; they are primarily concerned with code maintainability, and correcting less-than-optimal choices for visible elements. However, for a minority of cases, these messages are considered errors. Therefore, they should be reviewed closely before they are excluded.
- WarningIssues reported at this level are generally concerned with doing things correctly to keep your code base stable, extensible, and maintainable.
- InformationalMessages at this level are returned by rules that report information about a target, as opposed to detecting errors in a target.

The Level Name column in the messages pane indicates the level of the rule that generated the message. The Level column provides an iconic view of the same information.

Certainty estimates the probability of an issue being detected correctly. Certainty is affected by the issue-detection algorithm of a rule and by issue-specific considerations, which cannot be checked using static code analysis. Certainty is expressed as a percentage. A high value indicates confidence that the rule is detecting a real issue. If the certainty factor of a rule is low, because it will be for rules where there is a high expectation of false positives, the message certainty value reflects this.

It is unwise to automatically ignore messages based on level or certainty. You

should review all messages to determine whether a message is reporting an actual issue, and then assess the risks involved in ignoring the error.

■Specifying a Location for a Rule Assembly using Environment Variables

You can specify the location a rule assembly is subsequently loaded from using a path that includes environment variables. FxCop reads environment variables when the application is started. Any changes to environment variables are ignored until FxCop is restarted. Rules must be grouped by file name for this option to be available.

To specify the location using environment variables

- 1. In the configuration pane, click the Rules tab.
- 2. Right-click the rule assembly and select Properties.
- 3. Type the location in the Save Name box.

⊠Note

Using environment variables for the location of an assembly can cause the assembly failing to load if the environment variables are not initialized. This might require manually editing the project file to correct. As an alternative to environment variables, consider using a Shared Project, which can be found in the "Shared Project" section of the topic

Managing FxCop Projects.

Send <u>feedback</u> on this topic to Microsoft.
Visual Basic C#
Visual C++
F#
JScript
Managing Messages
Send Feedback

Messages are generated by rules as they analyze targets. Each message is bound to a specific rule and a specific target. A message can report multiple occurrences, also know as issues, of a rule violation.

To generate messages, specify the set of target assemblies and rules and then run an FxCop analysis using one of the following methods:

- Click Analyze on the toolbar.
- Press the F5 function key.
- On the Project menu, select Analyze.

The messages pane displays the messages generated by the analysis as shown in the following screen shot. The text font color of a message indicates its message level. To change the font or color, see the "Setting Preferences and Project Defaults" section of the topic <u>Managing FxCop Projects</u>.

FxCop application window



Use the Active, Excluded In Project, and Absent buttons at the top of the pane to view a list of messages that are in the corresponding message state. For more

information, see <u>Message States</u>. Sort the messages by clicking the appropriate column heading.

Right-click a message to see available message actions. Depending on the message state, the message actions include the following:

- Changing the state of the message. For more information, see <u>Message</u> <u>States</u>.
- Adding a note to a message. For more information, see <u>Viewing Message</u> <u>Details</u>.
- Displaying details about the message. For more information, see <u>Viewing</u> <u>Message Details</u>.
- Customizing the set of columns displayed in the messages pane. For more information, see <u>Configuring Columns</u>.
- Sorting the messages.
- Copying the selected message to the Clipboard.

One or more messages can be copied to the Clipboard in either comma-separated value (CSV) format or XML format. The information copied consists of the same columns as those displayed in the messages pane, and is a subset of the information saved in the project or report file.

To copy messages

- 1. Select the messages to be copied.
- 2. Right-click one of the messages.
- 3. Select Copy As and then select the format, either Csv or Xml.

In This Section

Message States

Configuring Columns

Filtering Messages

Viewing Message Details

Excluding Messages

Saving Message Reports

Message States

During its lifetime, the state of a message can change. The following table identifies the possible states of a message. The Active, Excluded In Project, and Absent states are mutually exclusive and can be viewed by using the corresponding buttons at the top of the messages pane. The New state can be combined with any of the other states. Messages in the New state appear in a bold font.

The target associated with an excluded message is not checked by the rule associated with the excluded message during an analysis run.

Message state	Description
New	The message was reported in the most recent analysis and was not previously present in the project in any other state.
Active	The message was reported in the most recent analysis. It might be in the New state.
Excluded In Project	The message was manually removed from the active list.
Absent	The message was previously in the active list, is not excluded, and was not reported in the most recent

analysis.

Configuring Columns

You can configure which columns are displayed in what order in each of the message lists, and set each column's width.

To configure columns

- 1. Right-click in the messages pane.
- 2. Select Configure Columns.
- 3. Select the check box next to each column name to display.
- 4. Select the column name to allow changing the column width.
- 5. Use the Move Up and Move Down buttons to arrange the column order.
- 6. Click OK.

The following table describes each column.

Column	Description
New	Whether the message was first seen in the most recent analysis. Appears for active messages only.
Level	An iconic view of the message level. Appears in the default view.

Level Name	The message level name.
Certainty	The estimate of the probability that the issue is detected correctly. Appears in the default view.
Fix Category	Whether the fix for a violation of the rule constitutes a breaking change. Appears in the default view.
Resolution	Describes how to resolve a violation of the rule.
Rule	The name of the rule. Appears in the default view.
Item	The programming element that caused the message to be generated. Appears in the default view.
Rule Assembly	The assembly that contains the rule that generated the message.

I	
File	The assembly that contains the programming element that caused the message to be generated.
Created	The date and time the message was first seen.
Last Seen	The date and time the message was last seen.
Issues	The number of issues associated with the message. An issue is one occurrence of the rule violation.
User	The user who added the last note.
Last Note	Contents of the last note.
Last Note Modified	The date and time the last note was modified.
Seen Last Run	Whether the message was seen in the most recent analysis. Appears for excluded messages only.

Filtering Messages

Filtering lets you see a subset of the messages present in the messages pane. After you perform an analysis, you filter the displayed messages by clicking items in the configuration pane:

• Select a rule category to display only messages generated by the rules in that category. For more information about rule categories, see the "Grouping Rules" section in the topic

Managing Rules.

- Select a specific rule to display only messages generated by that rule.
- Select a target to display only messages for that target and its child nodes.

To remove the current filter and view all the messages, click the FxCop project node at the root of the tree view in the Targets or Rules tab.

When filtering is disabled from the Tools Settings dialog box, all messages are displayed regardless of any selection in the configuration pane. The Tools Settings are described in the "Setting Preferences and Project Defaults" section of the <u>Managing FxCop Projects</u> topic.

Viewing Message Details

When you select a message and press ENTER, right-click a message and select Properties, or double-click a message, the Message Details dialog box displays information about the message and the rule that generated the message. Previous and Next buttons provide a means to navigation through the list of messages. When an active or an excluded message is displayed, an Exclude or Unexclude button is available, respectively, that lets you change the exclude state of the message. For more information, see

Excluding Messages. This dialog box also lets you store text notes with the message. The following screen shots show an example dialog box for an active message.

Message Details Issues Rule Details Message Rule Support Notes Microsoft.VisualStudio.Tools.Applications.ProjectGenerator.serializer_ Item: Level Certainty Resolution 🛚 😢 Critical Error Consider making ProjectGenerator.serializer_Unkno 40 > < Source: [Location not stored in Pdb] Resolution Consider making ProjectGenerator.serializer_UnknownNode(Object, XmlNodeEventArgs):Void not externally visible. 0K Exclude Previous Next

Issues tab of the Message Details dialog box

Message tab of the Message Details dialog box



The following table describes the information available on the Issues tab and the Message tab in this dialog box. The Rule Support tab and the Rule Details tab display information about the rule that generated the message. For a description of these elements, see the "Viewing Rule Details" section of the <u>Managing Rules</u> topic.

Message elements	Description
Issues tab	
Item	The item that caused the message to be reported. This can be a namespace, assembly, resource, type, member, or parameter.
	The importance of the issue

Level	that is identified by the rule. For a discussion of the levels, see the "Level and Certainty" section of the <u>Managing Rules</u> topic.
Certainty	An estimate of the probability that the issue is detected correctly.
Source	The location of this item in source code if Program Database (PDB) information is available. More information is provided immediately following this table.
Resolution	Instructions on the recommended way to fix the issue reported by the message.
Message tab	
File	The assembly that contains the item.
Created	The date and time when the message was first reported. This value is stored in coordinated universal time

	(UTC) but displayed in local time.
Last Seen	The most recent date and time that the message was reported. This value is stored in UTC but displayed in local time.
Status	The state of the message.
Tree Location	The location of the item in the Targets tree of the configuration pane. Double- click the item to change the display focus to the configuration pane that displays the item.

The Source detail item displays one of the following values:

- Source file and line number informationClick the link to have FxCop open the source code in an editor. For information about how to specify a source code editor, see the "Setting Preferences and Project Defaults" section in the <u>Managing FxCop Projects</u> topic.
- <Source Lookup Disabled> Source file lookup is disabled. For information about how to enable source file lookup, see the "Attempt source file lookup" section of the <u>Managing FxCop Projects</u> topic.
- <Location not stored in Pdb>Either the correct PDB file for the assembly is unavailable or source information is unavailable for the programming element. The PDB file must be located either in the same directory as its corresponding assembly or in the location referred to by the _NT_SYMBOL_PATH environment variable. PDB information is available

only for methods and property accessors.

The Notes tab allows optional user-supplied notes to be associated with the message. Notes can also be added to a message when it is excluded from analysis.

To add a note to a message

- 1. Double-click a message.
- 2. Select the Notes tab.
- 3. Click Add to display the Edit Note dialog box.
- 4. Type the note in the Text box.
- 5. Click OK.

The note, the user who entered the note, and the time the note was entered are displayed in the Note pane.

Excluding Messages

Occasionally, you might want to allow specific exceptions to an FxCop rule without ignoring or removing the rule or target. For example, you might implement a design choice that intentionally violates the .NET Framework best practices for a specific type, or FxCop might report a "false positive," a message that is not accurate.

To eliminate analysis by a specific rule on a specific target, exclude the associated message. Excluded messages are displayed using the Excluded In Project button on the messages pane.

Excluding a message causes the state of the message to change but does not eliminate the message. The excluded message contains the name of the user who created it and an optional note that explains the reason for excluding the message. The excluded message can be un-excluded, in which case it will appear in the active or absent list depending on the value of the Seen Last Run field.

To exclude messages

• Right-click the messages and select Exclude.

You can also double-click a single message, and then click Exclude in the Message Details dialog box.

- 1. Type relevant information in the Text box.
- 2. Click OK.

To re-include excluded messages

- 1. Click the Excluded button located on the top of the message pane.
- 2. Select one or more excluded messages.
- 3. Right-click the selection and then click Mark as Active or Mark as Absent.

You can also double-click a single message, and then click Unexclude in the Message Details dialog box.

Saving Message Reports

FxCop saves message reports in XML format. A saved report contains all messages that match the report settings in the Project Options dialog box. For more information about report settings, see the "Save Messages" section of the

Managing FxCop Projects topic.

Saved messages can be imported into a project and are treated as an extension of the project contents. For more information about how to import message reports, see the "Importing Messages into a Project" section of the <u>Managing FxCop</u> <u>Projects</u> topic.

To save a report

- 1. On the File menu, select Save Report As. If no messages match the report settings, this option is unavailable.
- 2. Locate the directory where you want to save the report.
- 3. In the File Name box, type a name for the saved report file.
- 4. Click Save.

The XML schema for reports is located in the \Xml subdirectory installed by FxCop. The schema file name is FxCopReport.xsd. If the Report Stylesheet box in the Project Options dialog box contains a value, the saved XML will contain a processing instruction for the specified style sheet. If Apply Stylesheet is selected, the report style sheet XSL is applied to the report data, and the result of the transformation is saved. For more information about Report Stylesheets, see the "Report Stylesheet and Apply Stylesheet" section in the <u>Managing FxCop</u> <u>Projects</u> topic.

Visual Basic C#
Visual C++
F#
JScript
Using FxCopCmd
Send Feedback

FxCopCmd is the command-line companion to the FxCop application. Like the application, FxCopCmd analyzes managed code assemblies and reports information about the assemblies, such as possible design, localization, performance, and security improvements. FxCopCmd can be used as a standalone tool, added to automated build processes, or integrated with Microsoft Visual Studio .NET as an external tool.

If you are starting a new software development effort, or are adding FxCop to an existing development effort, start by creating a project to store settings for the analysis using the FxCop application. FxCopCmd does not support creating or configuring projects. An FxCop project specifies the set of assemblies to be analyzed, the rules used to analyze the assemblies, and the most recent analysis results reported by FxCop. Although individual types and rules can be directly specified with the FxCopCmd's command-line options, an FxCop project is easier to use. After a project has been created and saved, you can run the analysis it describes by using the application or the command–line tool.

For a quick introduction to FxCopCmd, see <u>Quick Start Guide to FxCopCmd</u>. For a complete description of the FxCop application's user interface and features, see <u>Using FxCop</u>.

In This Section

Command-Line Tool Options

Describes the command-line options supported by FxCopCmd.

Analysis Reports

Discusses the analysis reports sent to the console window or an XML file.

FxCopCmd Errors

Provides information about the errors that can occur while loading or analyzing targets.

Adding FxCopCmd to Your Build Process

Explains how to add FxCopCmd to a build script.

Visual Studio Integration of FxCop

Discusses integrating FxCopCmd with Visual Studio.

Visual Basic
 C#
Visual C++
F#
JScript
Command-Line Tool Options
Send Feedback

The following table shows the options that are supported by FxCopCmd. Option values are not case sensitive.

Option	Details
/ applyoutXsl or	Applies the XSL transformation that is specified in /outXsl to the analysis report before saving the file.
/aXsl	Cannot appear multiple times.
	Optional.
/ console or / c	Directs analysis output to the console or to the Output window in Visual Studio .NET. By default, the XSL file FxCopConsoleOutput.xsl is applied to the output before it is displayed. For more information, see <u>Visual Studio</u> Integration of ExCop

	Required unless / consoleXsl , / out , or / update is present.
/consoleXsl: <xsl> or /cXsl: <xsl> where <xsl> is the name of the XSL file to use.</xsl></xsl></xsl>	Specifies the XSL or XSLT file that contains a transformation to be applied to the analysis output before it is displayed in the console or the Output window of Visual Studio. This option overrides the default XSL file applied to the analysis output. Required unless /console, /out, or /update is present.
/directory: <directory> or /d: <directory> where <directory> is the name of the directory to search for target assembly dependencies.</directory></directory></directory>	Specifies an additional directory to search for assembly dependencies. FxCopCmd always searches the target assembly directory and the current working directory. Can appear multiple times. Optional.

/file:	
<file directory=""></file>	
or	Specifies the target assembly to analyze. If
/ f: <file directory=""></file>	FxCopCmd tries to analyze all files that have
where	the leve of lun extension.
<file directory=""> is the name of an executable</file>	Can appear multiple times.
(.exe) file or DLL, or a directory to	Required unless / project is present.
browse ioi	
assemblies	
/ help or / ?	Displays a summary of FxCopCmd options. Other options that appear with /help are ignored.
/ help or / ?	Displays a summary of FxCopCmd options. Other options that appear with / help are ignored.
/ help or / ? / import: <file directory=""> or</file>	Displays a summary of FxCopCmd options. Other options that appear with / help are ignored. Specifies the name of an analysis report or project file to import. Any messages in the imported file that are marked as excluded are not included in the analysis results.

<file directory=""> is the name of the FxCop report (.xml) or project (.FxCop) file to use, or a directory to browse for report or project files.</file>	 that have the .xml extension. To import all FxCop project files instead, include *.FxCop. If analysis results are saved to a project file by using the /update option, the imported messages are not saved.
	Can appear multiple times.
	Optional.
	Specifies the file name for the analysis report. For more information, see <u>Analysis Reports</u> .
/out: <file></file>	If the file exists, it is overwritten without warning. If no items are reported by the analysis and the file does not exist, it is not created. If
or	deleted.
/ o: <file> where <file> is the name of the file for the analysis report.</file></file>	By default, the file includes an xml-style sheet processing instruction that references FxCopReport.xsl. The file is saved in XML

	format unless / applyoutXsl is specified.
	Cannot appear multiple times.
	Required unless / console, /consoleXsl , or / update is present.
/outXsl: <xsl></xsl>	
or /oXsl: <xsl> where <xsl> is the name of the XSL file that is referenced in the processing instruction.</xsl></xsl>	Reference the specified XSL in the XML report file, /outxsl:none generates an XML report with no XSL style sheet.
/ project: <file> or /p: <file> where <file> is the name of the FxCop project file (.FxCop) to use.</file></file></file>	Specifies the file name of an FxCop project file. Cannot appear multiple times. Required unless / file and / rule are present.

/ rule: <file directory=""></file>	
or / r: <file directory=""></file>	Specifies the location of rule libraries to load. If you specify a directory, FxCopCmd tries to load all files that have the .dll extension.
where <file directory=""> is the name of a rule library (.dll) or a directory to browse for rule libraries.</file>	Can appear multiple times. Required unless / project is present.
/ summary or / s	Includes a summary report that has the informational messages returned by FxCopCmd. The summary shows the number of items found, how many items were new, and the running time for the analysis. Optional.
/ types: <type list> or</type 	Specifies the types to analyze. This option disables analysis of assemblies, namespaces, and resources; only the specified types and their

/ t: <type list=""></type>	members are included in the analysis.			
where <type list> is a comma- separated list of type names to analyze.</type 	The elements of <type list> can use the wildcard character '*' at the end of the name to select multiple types. Cannot appear multiple times.</type 			
	Optional.			
/ update or / u	Saves the results of the analysis in the project file. This option is ignored if the / project option is not specified. Optional.			
/ verbose or	Outputs verbose information during analysis.			
/v	Optional.			

At minimum, you must specify either of the following:

- The location of the analysis output, using the /**console**, /**consoleXsl**, or /**out** option, and one of the following:
 - A project that uses the **/project** option.
 - An assembly to analyze and a rules directory pair using the /**file** and

/rule options.

-or-

• A project that uses the **/project** option and the **/update** option.

Option arguments have the following characteristics:

• They can contain wildcard characters. For example, you could specify the target assemblies by using a path in this form:

..\MyBuildDirectory\Bin*.*

- They can contain environment variables. For example, if you have the environment variable FXCOPRULEDIR defined, you could specify the rule libraries using a path that contains %FXCOPRULEDIR%.
- Option arguments that contain spaces must be enclosed in quotation marks. This includes environment variables that refer to values that contain spaces. For example, if the environment variable FXCOPRULEDIR refers to C:\Program Files\Microsoft FxCop\Rules, the rule option is specified as /r:"%FXCOPRULEDIR%".

If the **/file** and **/project** options are both present, the assemblies specified by the **/file** option are loaded after the assemblies specified in the project file. If an attempt is made to load a duplicate assembly, FxCopCmd displays an error message that indicates that an assembly with an identical signature has already been loaded.

If the **/rule** and **/project** options are both present, the rule libraries specified by the **/rule** option are loaded after those specified by the **/project** option. If an attempt is made to load a duplicate rule library, FxCopCmd displays an error message that indicates that a rule library with an identical signature has already been loaded.

Visual Basic C#
Visual C++
F#
JScript
Analysis Reports
Send Feedback

Analysis reports can be displayed or saved to an XML file. In either case, informational messages are displayed that show the progress of the analysis, such as rule assembly loading and engine initialization, in addition to any errors that are encountered. These messages can be suppressed by using two System.Diagnostics.BooleanSwitch switches in the FxCopCmd.exe.config file located in the FxCop installation directory. You can turn off the display of loading and initialization messages by changing the value of the WriteStandardOutput switch to 0. Similarly, you can suppress the display of any error messages by changing the value of the WriteStandardErrorOutput switch to 0.

When the **/project** option is specified, the output is based on the Save Messages settings for reports in the project file. The output is either a console or a report file. These settings determine which message states are included in the report. If no messages are included in the report, the output does not contain any messages, even if messages are generated in the FxCopCmd analysis. For information about how to change the Save Messages settings, see the "Setting Project Options" section of the Managing FxCop Projects topic.

When the **/output** option is used, the analysis report is saved to an XML file. The XML schema, FxCopReport.xsd, is located in the \Xml subdirectory installed by FxCop. This directory also contains XSL style sheets that transform the analysis report XML into formatted reports. By default, a style sheet processing instruction for the summary/detail report style sheet, FxCopReport.xsl, is automatically added to analysis reports. You can specify a different style sheet or disable this feature by changing the project settings by using the FxCop application. For more information, see the "Report Stylesheet" and Apply Stylesheet" section in the <u>Managing FxCop Projects</u> topic.

If source code information is available for the programming element that caused

a message to be generated, the analysis report includes the source code file and line number information for the element. FxCop looks for source code information in the program database (PDB) file for the assembly. This file must be located in the same directory as the corresponding assembly or in the location specified by the _NT_SYMBOL_PATH environmental variable. The C# and Visual Basic .NET compilers emit PDB information only for methods and properties; therefore, other programming elements that cause a message to be generated do not have this information.

Visual Basic C#
Visual C++
F#
JScript
FxCopCmd Error Codes
Send Feedback

Errors can occur while you are loading items into FxCopCmd for analysis, or during the analysis itself. FxCopCmd does not consider all errors to be fatal. If, despite errors, FxCopCmd has sufficient information to perform a partial analysis, it performs the analysis and reports the errors that occurred. Error messages are sent to the console's error stream, and informational messages about the progress of the analysis are sent to the console's output stream.

In addition to outputting error messages, FxCopCmd returns an error code. The error code is a 32-bit integer that contains a bitwise combination of numeric values that correspond to errors. The following table describes the error code values that are returned by FxCopCmd.

Error	Numeric value
No errors	0x0
Analysis error	0x1 - fatal error
Rule exceptions	0x2
Project load error	0x4

Assembly load error	0x8
Rule library load error	0x10
Import report load error	0x20
Output error	0x40
Command line switch error	0x80
Initialization error	0x100
Assembly references error	0x200
Unknown error	0x1000000

The "Analysis error" (bit 1) is set in the returned value if any errors were fatal. This means that analysis could not finish. When applicable, the error code also contains the underlying cause of the fatal error. The following are some of the conditions that are considered fatal errors:

- The analysis could not be performed because of insufficient input.
- The analysis threw an exception that is not handled by FxCopCmd.
- The specified project file could not be found or is corrupted.
- The output option was not specified or the file could not be written.

Visual Basic C#
Visual C++
F#
JScript
Adding FxCopCmd to Your Build Process
Send Feedback

For ongoing development projects, you can automate the task of analyzing your assemblies. By using the following procedure, you can customize your build system so that a saved project runs every time that your assemblies are built.

Procedures

To analyze assemblies as part of the build process

- 1. Use a build script to build your managed assemblies.
- 2. In your build script, call FxCopCmd to analyze your assemblies.
- 3. Review the analysis report output from step 2.
- 4. If new messages must be excluded, or other changes to the project are required, use the FxCop application to update your project and save your changes.
- 5. Correct any code defects that are detected by FxCopCmd.

If you are not using a project file to specify information to FxCopCmd with the /**project** option, you must supply an assembly using the /**file** option, and a rules directory using the /**rules** option. In either case, you must also specify the location of the output file that uses the /**out** option. For more information, see <u>Command-Line Tool Options</u>.

Visual Basic C#
Visual C++
F#
JScript
Visual Studio Integration of FxCop Send Feedback

To integrate FxCopCmd into Visual Studio, you must configure Visual Studio to run FxCopCmd as an external tool. After it is configured, FxCopCmd uses the /**console** and /**consoleXsl** options to send an abbreviated analysis report, including source code information if available, to the Output window in Visual Studio.

To set up FxCop as an external tool in Visual Studio

• On the Tools menu, click External Tools, and then click Add.

The External Tools configuration dialog box is displayed. The following screen shot shows the dialog box after the steps in this procedure have been completed.

External Tools			×		
Me <u>n</u> u contents:					
FxCop			Add		
ActiveX Control Test Co&ntainer		Delete			
Create &GUID Error Loo&kup					
ATL/MFC &Trace Tool					
			Move Up		
			Move Do <u>w</u> n		
<u>T</u> itle:	FxCop				
<u>C</u> ommand:	\Program	\Program Files\Microsoft FxCop\FxCopCmd.e			
Arguments:	/c /f:\$(Ta	/c /f:\$(TargetPath) /r:"\Program Files\Micros			
Initial directory:	\Program Files\Microsoft FxCop				
✓ Use <u>O</u> utput window					
Treat output as Unicode 🔽 Close on exit					
ОК	Cancel	App <u>ly</u>	Help		

FxCop appears on the Tools menu.

After you successfully build your project, you can perform an FxCop analysis without leaving the integrated development environment (IDE). To do this, click FxCop on the Tools menu. Make sure the Output window is visible; this is where the analysis output is displayed. The following screen shot shows the IDE with the output of an FxCopCmd analysis displayed in the Output window.


If source code information is available for the programming element that caused a message to be generated, the list item for that element in the analysis report starts with the source code file and line number information for the element. Clicking the item will move the cursor to the corresponding line in the Code Editor. For more information, see <u>Analysis Reports</u>.

Send *feedback* on this topic to Microsoft.

Visual Basic C#
Visual C++
F#
JScript
FxCop Warnings
Send Feedback

FxCop provides warnings that indicate rule violations in managed code libraries. The warnings are organized into rule areas such as design, localization, performance, security, and so forth. Each warning signifies a violation of an FxCop rule. This section provides in-depth discussions and examples for each FxCop warnings.

The following table shows the type of information provided for each warning:

Item	Description
Туре	The TypeName for the rule.
CheckId	The unique identifier for the rule.
Category	The category of the warning.
Message Level	The importance of the issue that is identified by the rule.
	The estimate of the

Certainty	probability that the issue is detected correctly. This value is an integer between 1 and 99.	
Breaking Change	Whether the fix for a violation of the rule constitutes a breaking change. Breaking change means that an assembly that has a dependency on the target that caused the violation will not re- compile with the new fixed version or might fail at runtime because of the change. When multiple fixes are available and at least one fix is a breaking change and one fix is not, both 'Breaking' and 'Non Breaking' are specified.	
Cause	The specific managed code that causes the rule to generate a warning.	
Description	Discusses the issues behind the rule.	
How to Fix Violations	Explains how to change the source code to satisfy the rule and prevent it from generating a warning.	

When to Exclude Warnings	Describes when it is safe to exclude a warning from the rule.
Example Code	Examples that violate the rule and "fixed" examples that satisfy the rule.
Related Warnings	Related warnings.

In This Section

Design Warnings

Warnings that support proper library design as specified by the .NET Framework Design Guidelines.

Globalization Warnings

Warnings that support world-ready libraries and applications.

Interoperability Warnings

Warnings that support interacting with COM clients.

Naming Warnings

Warnings that support adherence to the naming conventions of the .NET Framework Design Guidelines.

Performance Warnings

Warnings that support high performance libraries and applications.

Security Warnings

Warnings that support safer libraries and applications.

Usage Warnings

Send feedback on this topic to Microsoft.

Visual Basic C#
Visual C++
F#
JScript
FxCop FAQ
Send Feedback

This document discusses frequently asked questions about FxCop installation, loading and analysis, rules, and other FAQs.

Installation

Question

FxCopCmd does not work as an external tool in Visual Studio. What can be the problem?

Answer

Inspect the settings in the External Tools dialog box.

- The value of all command-line options that contain a space must be enclosed in quotation marks (specified in the Arguments text box). This includes environment variables that when you expand them contain spaces.
- The length of the expanded command line must be less than 255 characters.
- Verify the command line by selecting the "Prompt for arguments" check box and running FxCop from the VS IDE. A dialog box will appear with the command line expanded.
- Make sure that the "Use Output window" check box is selected.

Question

Can FxCop be used in a side-by-side installation of the .NET Framework?

Answer

Yes.

Loading/Analysis

Question

Why will not FxCop load some of my assemblies?

Answer

FxCop only loads and analyzes managed code assemblies. The following are the major reasons an assembly fails to load:

- The assembly is a native binary. This generates an invalid file format exception.
- The assembly is a COM interop assembly generated by the Type Library Importer tool (Tlbimp.exe). FxCop does not load these assemblies because they wrap non-managed code.
- All dependencies of the target assembly could not be found. Although interop assemblies are not loaded, if they are dependencies of the target assembly, they must be located by FxCop. Look at the dependency list in the manifest using the MSIL Disassembler (Ildasm.exe).

Question

- Why doesn't FxCop resolve the source locations of all messages?
- I built my assemblies with debug information. Why does <PDB not available or source lookup disabled> appear in the Source field of the Message dialog box?
- What does <Unknown File> in the Source field of the Message dialog box mean?

Answer

FxCop requires program debug database (PDB) files in order to provide source locations.

- <Source Lookup Disabled > indicates that source file lookup is disabled in the UI.
- <Location not stored in Pdb> indicates that either the correct PDB file cannot be located or no PDB information exists for the target element. This can occur if the PDB does not exist, does not exist in the same directory as the target assembly, or does not match the version of the target assembly. If the PDB is in a different directory, set the _NT_SYMBOL_PATH environment variable to the location of the PDB file. Another possibility is that the analyzed item is picked up from the GAC. The C# and Visual Basic .NET compilers only generate PDB information for methods and properties. Therefore FxCop cannot find the source location for programming elements such as attributes, types, or namespaces.

Question

Messages are displayed in the UI when I analyze a project. Why aren't messages written when I analyze the same project on the command-line? When I include the /summary switch, it shows that messages were generated.

Answer

The console message output depends on the report options set in the project file. By default, all active messages are written to the console. To verify this setting, from the Project menu, select Options to display the Project Options dialog box. In the Save Messages group, the Active check box for a report should be selected.

Rules

Question

I need help understanding the FxCop rules.

Answer

More information about a rule is available by double-clicking the rule and then selecting the URL link under the Support tab.

Question

I have developed some custom rules that I want to share with other people. Is there somewhere I can post these rules?

Answer

Locate <u>GotDotNet User Samples</u> and filter on FxCop in the "All related products" drop-down list.

Question

The message from the Virtual methods and their overrides require the same LinkDemand status rule states that I should add a LinkDemand to my code. I do not have the source code for the virtual method. How do I find which permission to link demand?

Answer

Use the Permissions View tool (Permview.exe), with the /decl option, to examine the assembly that contains the virtual method. The /decl option displays all declarative security at the assembly, class, and method level for the assembly.

Question

I have lots of CultureInfoShouldBePassed messages on code generated by the VS IDE, specifically from the InitializeComponent method of a form. The note just before the method states "do not modify the contents of this method." How can I fix these?

Is there any way of telling the Visual Studio .NET Designer to use CultureInfo.InvariantCulture?

Answer

The CultureInfoShouldBePassed rule alerts a developer to all culture-aware overloads and requires an explicit decision of whether a global audience is expected. There currently is not a way to tell the Visual Studio .NET designer to specify a CultureInfo.InvariantCulture. However, you can safely exclude these FxCop messages on code generated by Visual Studio.

Miscellaneous

Question

How can I add custom words to the FxCop dictionary? How can I add project specific dictionaries?

Answer

Create a file that is named CustomDictionary.xml. Add the following XML structure, with the new words (case insensitive) under the <Recognized> node.

Copy Code

```
<Dictionary>
<Words>
<Recognized>
<Word>aNewWord</Word>
<Word>AnotherNewWord</Word>
</Recognized>
</Words>
</Dictionary>
```

To use the dictionary with all projects, put the file in the FxCop install directory (usually C:\Program Files\Microsoft FxCop). For project-specific dictionaries, put the file in a separate directory together with the project file. For the words to be recognized, you must close and restart FxCop after you create or modifying the custom dictionary.

Question

How can we share a project/report file among our group when we have different directory/drive structures?

Answer

Instead of using the Shared Project option, which requires all files to be on the

same drive, do the following:

- Use environment variables to specify all target and rule paths.
- Change the Report Stylesheet link (choose Options from the FxCop Project menu) to your own shared XSL document.

Send <u>feedback</u> on this topic to Microsoft.