Secrets
ABAP
Apex
C
C++
CloudFormation
COBOL
C#
CSS
Flex
Go
HTML
Java
JavaScript
Kotlin
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

C# static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your C# code

All rules 409 | Vulnerability 34 | Bug 76 | Security Hotspot 28 | Code Smell 271 | Quick Fix 52

Tags

Search by name...

"protected" members
Code Smell

Underscores should be used to make large numbers readable
Code Smell

"ToString()" calls should not be redundant
Code Smell

"==" should not be used when "Equals" is overridden
Code Smell

An abstract class should have both abstract and concrete methods
Code Smell

Multiple variables should not be declared on the same line
Code Smell

Culture should be specified for "string" operations
Code Smell

"switch" statements should have at least 3 "case" clauses
Code Smell

break statements should not be used except for switch cases
Code Smell

String literals should not be duplicated
Code Smell

Files should contain an empty newline at the end
Code Smell

Unused "using" should be removed
Code Smell

"Assembly.GetExecutingAssembly" should not be called

Analyze your code

Code Smell | Major | performance

Using `Type.Assembly` to get the current assembly is nearly free in terms of performance; it's a simple property access. On the other hand, `Assembly.GetExecutingAssembly()` can take up to 30 times as long because it walks up the call stack to find the assembly.

Note that `Assembly.GetExecutingAssembly()` is different than `Type.Assembly` because it dynamically returns the assembly that contains the startup object of the currently executed application. For example, if executed from an application it will return the application assembly, but if executed from a unit test project it could return the unit test assembly. `Type.Assembly` always returns the assembly that contains the specified type.

Noncompliant Code Example

```
public class Example
{
   public static void Main()
   {
      Assembly assem = Assembly.GetExecutingAssembly(); // N
      Console.WriteLine("Assembly name: {0}", assem.FullName
   }
}
```

Compliant Solution

```
public class Example
{
   public static void Main()
   {
      Assembly assem = typeof(Example).Assembly; // Here we
      Console.WriteLine("Assembly name: {0}", assem.FullName
   }
}
```

Available In:

sonarlint | sonarcloud | sonarqube

A close curly brace should be located at the beginning of a line

Code Smell

Tabulation characters should not be used

Code Smell

Methods and properties should be named in PascalCase

Code Smell

Track uses of in-source issue suppressions

Code Smell