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

"this" should not be exposed from constructors

Analyze your code

Code Smell | Major | multi-threading suspicious

In single-threaded environments, the use of `this` in constructors is normal, and expected. But in multi-threaded environments, it could expose partially-constructed objects to other threads, and should be used with caution.

The classic example is a class with a `static` list of its instances. If the constructor stores `this` in the list, another thread could access the object before it's fully-formed. Even when the storage of `this` is the last instruction in the constructor, there's still a danger if the class is not `final`. In that case, the initialization of subclasses won't be complete before `this` is exposed.

This rule raises an issue when `this` is assigned to any globally-visible object in a constructor, and when it is passed to the method of another object in a constructor

Noncompliant Code Example

```
public class Monument
{
  public static readonly List<Monument> ALL_MONUMENTS = new
  // ...

  public Monument(string location, ...)
  {
    ALL_MONUMENTS.Add(this);  // Noncompliant; passed to a m

    this.location = location;
    // ...
  }
}
```

Exceptions

This rule ignores instances of assigning `this` directly to a `static` field of the same class because that case is covered by {rule:csharpsquid:S3010} .

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