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...

Nested blocks of code should not be left empty
Code Smell

Methods should not have too many parameters
Code Smell

Collapsible "if" statements should be merged
Code Smell

OS commands should not be vulnerable to argument injection attacks
Vulnerability

Logging should not be vulnerable to injection attacks
Vulnerability

Empty collections should not be accessed or iterated
Bug

Mutable, non-private fields should not be "readonly"
Bug

"string.ToCharArray()" should not be called redundantly
Bug

"base.Equals" should not be used to check for reference equality in "Equals" if "base" is not "object"
Bug

Property assignments should not be made for "readonly" fields not constrained to reference types
Bug

Flags enumerations should explicitly initialize all their members
Bug

Classes should not have only "private" constructors

Analyze your code

Bug | Major | design

A class with only `private` constructors can't be instantiated, thus, it seems to be pointless code.

Noncompliant Code Example

```
public class MyClass // Noncompliant
{
  private MyClass() { ... }
}
```

Compliant Solution

```
public class MyClass
{
  public MyClass() { ... }
}
```

Exceptions

Classes that themselves access their private constructors (singletons or smart enums) are ignored. Classes with only `static` members are also ignored because they are covered by Rule {rule:csharpsquid:S1118}.

Available In:

sonarlint | sonarcloud | sonarqube

"GetHashCode" should not reference mutable fields

Bug

Results of integer division should not be assigned to floating point variables

Bug

Integral numbers should not be shifted by zero or more than their number of bits-1

Bug

"Equals(Object)" and "GetHashCode()" should be overridden in pairs