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... 🔍

simplified

Code Smell

Boolean checks should not be inverted

Code Smell

Inheritance list should not be redundant

Code Smell

Redundant casts should not be used

Code Smell

Strings should not be concatenated using '+' in a loop

Code Smell

Unused local variables should be removed

Code Smell

Private fields only used as local variables in methods should become local variables

Code Smell

A "while" loop should be used instead of a "for" loop

Code Smell

"Equals" and the comparison operators should be overridden when implementing "IComparable"

Code Smell

Nested code blocks should not be used

Code Smell

Overriding members should do more than simply call the same member in the base class

Code Smell

"Any()" should be used to test for emptiness

Variables should not be self-assigned

Analyze your code

🐛 Bug   🔻 Major ❓

There is no reason to re-assign a variable to itself. Either this statement is redundant and should be removed, or the re-assignment is a mistake and some other value or variable was intended for the assignment instead.

Noncompliant Code Example

```
public void SetName(string name)
{
    name = name;
}
```

Compliant Solution

```
public void SetName(string name)
{
    this.name = name;
}
```

Available In:

sonarlint   sonarcloud   sonarqube

emptiness

⊗ Code Smell

Boolean literals should not be
redundant

⊗ Code Smell

Empty statements should be removed

⊗ Code Smell

Fields should not have public
accessibility

⊗ Code Smell

URIs should not be hardcoded

⊗ Code Smell