- 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#

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...

sealed

Code Smell

Overloads with a "StringComparison" parameter should be used

Code Smell

Overloads with a "CultureInfo" or an "IFormatProvider" parameter should be used

Code Smell

Types should not extend outdated base types

Code Smell

Properties should be preferred

Code Smell

Generics should be used when appropriate

Code Smell

Type names should not match namespaces

Code Smell

Strings should be normalized to uppercase

Code Smell

Exceptions should provide standard constructors

Code Smell

Assemblies should be marked with "NeutralResourcesLanguageAttribute"

Code Smell

Interfaces should not be empty

Code Smell

Enumerations should have "Int32" storage

Code Smell

Redundant pairs of parentheses should be removed

Analyze your code

Code Smell Major Quick Fix confusing

The use of parentheses, even those not required to enforce a desired order of operations, can clarify the intent behind a piece of code. But redundant pairs of parentheses could be misleading, and should be removed.

Noncompliant Code Example

```
if (a && ((x + y > 0))) // Noncompliant
{
  //...
}

return ((x + 1));  // Noncompliant
```

Compliant Solution

```
if (a && (x + y > 0))
{
  //...
}

return x + 1;

return (x + 1);
```

Available In:

sonarlint sonarcloud sonarqube

Generic methods should provide type parameters

Code Smell

Multidimensional arrays should not be used

Code Smell

"static readonly" constants should be "const" instead

Code Smell

Strings or integral types should be used for indexers

Code Smell