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

Culture should be specified for "string" operations

Analyze your code

Code Smell | Minor | unpredictable

`string.ToLower()`, `ToUpper`, `IndexOf`, `LastIndexOf`, and `Compare` are all culture-dependent, as are some (floating point number and `DateTime`-related) calls to `ToString`. Fortunately, all have variants which accept an argument specifying the culture or formatter to use. Leave that argument off and the call will use the system default culture, possibly creating problems with international characters.

`string.CompareTo()` is also culture specific, but has no overload that takes a culture information, so instead it's better to use `CompareOrdinal`, or `Compare` with culture.

Calls without a culture may work fine in the system's "home" environment, but break in ways that are extremely difficult to diagnose for customers who use different encodings. Such bugs can be nearly, if not completely, impossible to reproduce when it's time to fix them.

Noncompliant Code Example

```
var lowered = someString.ToLower(); //Noncompliant
```

Compliant Solution

```
var lowered = someString.ToLower(CultureInfo.InvariantCultur
```

or

```
var lowered = someString.ToLowerInvariant();
```

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