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...

Member initializer values should not be redundant
Code Smell

Unassigned members should be removed
Code Smell

Empty "case" clauses that fall through to the "default" should be omitted
Code Smell

Parameters with "[DefaultParameterValue]" attributes should also be marked "[Optional]"
Code Smell

Interfaces should not simply inherit from base interfaces with colliding members
Code Smell

Variables should not be checked against the values they're about to be assigned
Code Smell

Methods should not return constants
Code Smell

Attribute, EventArgs, and Exception type names should end with the type being extended
Code Smell

Loops should be simplified with "LINQ" expressions
Code Smell

Namespaces should not be empty
Code Smell

Non-derived "private" classes and records should be "sealed"
Code Smell

"NaN" should not be used in comparisons

Analyze your code

Bug Major

`NaN` is not equal to anything, even itself. Testing for equality or inequality against `NaN` will yield predictable results, but probably not the ones you want.

Instead, the best way to see whether a variable is equal to `NaN` is to use `Number.isNaN()`, since ES2015, or (perhaps counter-intuitively) to compare it to itself. Since `NaN !== NaN`, when `a !== a`, you know it must equal `NaN`.

Noncompliant Code Example

```
var a = double.NaN;

if (a == double.NaN) // Noncompliant; always false
{
  Console.WriteLine("a is not a number");  // this is dead c
}
if (a != double.NaN)  // Noncompliant; always true
{
  Console.WriteLine("a is not NaN"); // this statement is no
}
```

Compliant Solution

```
if (double.IsNaN(a))
{
  console.log("a is not a number");
}
```

Available In:

sonarlint | sonarcloud | sonarqube

"string.IsNullOrEmpty" should be used

Code Smell

Implementations should be provided for "partial" methods

Code Smell

Duplicate casts should not be made

Code Smell

Methods should not return values that are never used

Code Smell

Caller information arguments should