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...

used
Code Smell

"interface" instances should not be cast to concrete types
Code Smell

Literal boolean values should not be used in assertions
Code Smell

Optional parameters should not be used
Code Smell

Public constant members should not be used
Code Smell

Array covariance should not be used
Code Smell

"nameof" should be used
Code Smell

Modulus results should not be checked for direct equality
Code Smell

"for" loop increment clauses should modify the loops' counters
Code Smell

"switch" statements should not be nested
Code Smell

Methods and properties should not be too complex
Code Smell

Control flow statements "if", "switch", "for", "foreach", "while", "do" and "try" should not be nested too deeply
Code Smell

Parameter names used into ArgumentException constructors should match an existing one

Analyze your code

Code Smell Major

Some constructors of the `ArgumentException`, `ArgumentNullException`, `ArgumentOutOfRangeException` and `DuplicateWaitObjectException` classes must be fed with a valid parameter name. This rule raises an issue in two cases:

- When this parameter name doesn't match any existing ones.
- When a call is made to the default (parameterless) constructor

Noncompliant Code Example

```
public void Foo(Bar a, int[] b)
{
  throw new ArgumentException(); // Noncompliant
  throw new ArgumentException("My error message", "c"); // N
  throw new ArgumentException("My error message", "c", inner
  throw new ArgumentNullException("c"); // Noncompliant
  throw new ArgumentNullException("My error message", "c");
  throw new ArgumentOutOfRangeException("c");
  throw new ArgumentOutOfRangeException("c", "My error messa
  throw new ArgumentOutOfRangeException("c", b, "My error me
  throw new DuplicateWaitObjectException("c", "My error mess
}
```

Compliant Solution

```
public void Foo(Bar a, Bar b)
{
  throw new ArgumentException("My error message", "a");
  throw new ArgumentException("My error message", "b", inner
  throw new ArgumentNullException("a");
  throw new ArgumentNullException(nameof(a));
  throw new ArgumentNullException("a", "My error message");
  throw new ArgumentOutOfRangeException("b");
  throw new ArgumentOutOfRangeException("b", "My error messa
  throw new ArgumentOutOfRangeException("b", b, "My error me
  throw new DuplicateWaitObjectException("b", "My error mess
}
```

Exceptions

The rule won't raise an issue if the parameter name is not a constant value (inline declaration, nameof() or const variable).

Available In:

sonarlint | sonarcloud | sonarqube

"switch/Select" statements should contain a "default/Case Else" clauses

Code Smell

"if ... else if" constructs should end with "else" clauses

Code Smell

Control structures should use curly braces

Code Smell

Expressions should not be too complex

Code Smell