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

Types should not have members with visibility set higher than the type's visibility

Analyze your code

Code Smell | Major | confusing

There's no point in having a `public` member in a non-`public` type because objects that can't access the type will never have the chance to access the member.

This rule raises an issue when a type has methods, fields, or inner types with higher visibility than the type itself has.

Noncompliant Code Example

```
internal class MyClass
{
    public static decimal PI = 3.14m;  // Noncompliant

    public int GetOne() // Noncompliant
    {
        return 1;
    }

    protected record NestedType // Noncompliant: outer class
    {
        public bool FlipCoin() // Noncompliant: outer class
        {
            return false;
        }
        // ...
    }
}
```

Compliant Solution

```
public class MyClass // Class visibility upgrade makes membe
{
    public static decimal PI = 3.14m;

    public int GetOne()
    {
        return 1;
    }

    protected record NestedType
    {
        public bool FlipCoin() // Outer type is public
        {
            return false;
        }
        // ...
    }
}
```

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

Exceptions

User defined operators need to be public:

```
public static implicit operator byte(MyClass a) => 1; // Com
public static explicit operator MyClass(byte a) => new MyCla
```

Nested types, even if private, can be used and inherited in the parent type. In this case, the visibility of the outer type is considered.

```
internal class MyClass
{
    private class NestedClass
    {
        public int PublicProperty { get; } // Noncompliant:
        protected internal int ProtectedInternalProperty { g
        internal int InternalProperty { get; } // Compliant:
        protected int ProtectedProperty { get; } // Complian
        private protected int PrivateProtectedProperty { get
        private int PrivateProperty { get; }
    }
}
```

Available In:

sonarlint | sonarcloud | sonarqube