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

Methods and properties that don't access instance data should be static

[**Analyze your code**]

⊙ Code Smell   ⊘ Minor ⊙   🏷 pitfall

Methods and properties that don't access instance data can be `static` to prevent any misunderstanding about the contract of the method.

Noncompliant Code Example

```
public class Utilities
{
    public int MagicNum // Noncompliant
    {
        get
        {
            return 42;
        }
    }

    private static string magicWord = "please";
    public string MagicWord  // Noncompliant
    {
        get
        {
            return magicWord;
        }
        set
        {
            magicWord = value;
        }
    }

    public int Sum(int a, int b)  // Noncompliant
    {
        return a + b;
    }
}
```

Compliant Solution

```
public class Utilities
{
    public static int MagicNum
    {
        get
        {
            return 42;
        }
    }

    private static string magicWord = "please";
    public static string MagicWord
```

Left sidebar

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

Main content

```
    {
        get
        {
            return magicWord;
        }
        set
        {
            magicWord = value;
        }
    }

    public static int Sum(int a, int b)
    {
        return a + b;
    }
}
```

Exceptions

Methods with the following names are excluded because they can't be made `static`:

- Application_AuthenticateRequest
- Application_BeginRequest
- Application_End
- Application_EndRequest
- Application_Error
- Application_Init
- Application_Start
- Session_End
- Session_Start

Available In:

sonarlint ⊝ | sonarcloud ⬡ | sonarqube ⟩⟩