

- 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

### Exceptions should provide standard constructors

Analyze your code

- Code Smell
- Minor ?
- convention

Exceptions types should provide the following constructors:

- public MyException()
- public MyException(string)
- public MyException(string, Exception)
- protected or private MyException(SerializationInfo, StreamingContext)

That fourth constructor should be protected in unsealed classes, and private in sealed classes.

Not having this full set of constructors can make it difficult to handle exceptions.

#### Noncompliant Code Example

```
using System;

namespace MyLibrary
{
    public class MyException // Noncompliant: several constructors missing
    {
        public MyException()
        {
        }
    }
}
```

#### Compliant Solution

```
using System;
using System.Runtime.Serialization;

namespace MyLibrary
{
    public class MyException : Exception
    {
        public MyException()
        {
        }

        public MyException(string message)
            : base(message)
        {
        }

        public MyException(string message, Exception innerException)
            : base(message, innerException)
        {
        }
    }
}
```

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

```
protected MyException(SerializationInfo info, Streamin
    : base(info, context)
    {
    }
}
```

Available In:

**sonarlint**  | **sonarcloud**  | **sonarqube** 

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.  
[Privacy Policy](#)