

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C# 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... 🔍

Security Hotspot
Disabling ASP.NET "Request Validation" feature is security-sensitive
Security Hotspot
Allowing requests with excessive content length is security-sensitive
Security Hotspot
Setting loose file permissions is security-sensitive
Security Hotspot
Formatting SQL queries is security-sensitive
Security Hotspot
Using hardcoded IP addresses is security-sensitive
Security Hotspot
"goto" statement should not be used
Code Smell
"new Guid()" should not be used
Code Smell
Parameter validation in "async"/"await" methods should be wrapped
Code Smell
Parameter validation in yielding methods should be wrapped
Code Smell
Events should have proper arguments
Code Smell
"P/Invoke" methods should not be visible
Code Smell
Native methods should be wrapped

### Inner class members should not shadow outer class "static" or type members

Analyze your code

Code Smell Critical design pitfall

It's possible to name the members of an inner class the same as the static members of its enclosing class - possible, but a bad idea. That's because maintainers may be confused about which members are being used where. Instead the inner class' members should be renamed and all the references updated.

#### Noncompliant Code Example

```
class Outer
{
    public static int A;

    public class Inner
    {
        public int A; //Noncompliant
        public int MyProp
        {
            get { return A; } // Returns inner A. Was that intend
        }
    }
}
```

#### After a rename





```
class Outer
{
    public static int A;

    public class Inner
    {
        public int B;
        public int MyProp
        {
            get { return A; } // Still compiles and runs but func
        }
    }
}
```

#### Compliant Solution




```
class Outer
{
    public static int A;

    public class Inner
    {
        public int InnerA;
        public int MyProp
    }
}
```

 Code Smell
Methods should not have identical implementations
 Code Smell
Non-flags enums should not be marked with "FlagsAttribute"
 Code Smell
Classes implementing "IEquatable<T>" should be sealed
 Code Smell
"GC.SuppressFinalize" should not be called

```
{  
    get { return InnerA; }  
}  
}
```

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](#)