

- 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

### ASP.NET HTTP request validation feature should not be disabled

Analyze your code

Vulnerability Major

ASP.NET has a feature to validate HTTP requests to prevent potentially dangerous content to perform a cross-site scripting (XSS) attack. There is no reason to disable this mechanism even if other checks to prevent XSS attacks are in place.

This rule raises an issue if a method with parameters is marked with `System.Web.Mvc.HttpPostAttribute` and not `System.Web.Mvc.ValidateInputAttribute(true)`.

#### Noncompliant Code Example

```
public class FooBarController : Controller
{
    [HttpPost] // Noncompliant
    [ValidateInput(false)]
    public ActionResult Purchase(string input)
    {
        return Foo(input);
    }

    [HttpPost] // Noncompliant
    public ActionResult PurchaseSomethingElse(string input)
    {
        return Foo(input);
    }
}
```

#### Compliant Solution

```
public class FooBarController : Controller
{
    [HttpPost]
    [ValidateInput(true)] // Compliant
    public ActionResult Purchase(string input)
    {
        return Foo(input);
    }
}
```

#### Exceptions

Parameterless methods marked with `System.Web.Mvc.HttpPostAttribute` will not trigger this issue.

#### See

- [OWASP Top 10 2017 Category A7](#) - Cross-Site Scripting (XSS)
- [MITRE, CWE-79](#) - Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- [SANS Top 25](#) - Insecure Interaction Between Components

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

- [OWASP ASP.NET Request Validation](#)

#### Deprecated

This rule is deprecated; use {rule:csharpsquid:S5753} instead.

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](#)