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...

Return values from functions without side effects should not be ignored
Bug

Values should not be uselessly incremented
Bug

Collections should not be passed as arguments to their own methods
Bug

Related "if/else if" statements should not have the same condition
Bug

Objects should not be created to be dropped immediately without being used
Bug

Identical expressions should not be used on both sides of a binary operator
Bug

Loops with at most one iteration should be refactored
Bug

Variables should not be self-assigned
Bug

Constructing arguments of system commands from user input is security-sensitive
Security Hotspot

Deserializing objects without performing data validation is security-sensitive
Security Hotspot

Disabling ASP.NET "Request Validation" feature is security-sensitive

Exception types should be "public"

Analyze your code

Code Smell | Critical | error-handling api-design owasp

The point of having custom exception types is to convey more information than is available in standard types. But custom exception types must be `public` for that to work.

If a method throws a non-public exception, the best you can do on the caller's side is to `catch` the closest `public` base of the class. That is, you lose all that custom information you created the exception type to pass.

Noncompliant Code Example

```
internal class MyException : Exception   // Noncompliant
{
  // ...
}
```

Compliant Solution

```
public class MyException : Exception
{
  // ...
}
```

Exceptions

This rule ignores Exception types that are not derived directly from `System.Exception`, `System.SystemException`, or `System.ApplicationException`.

See

- OWASP Top 10 2017 Category A10 - Insufficient Logging & Monitoring

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

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