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...

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
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

"[Optional]" should not be used on "ref" or "out" parameters

Analyze your code

Code Smell | Critical | Quick Fix | pitfall

The use of `ref` or `out` in combination with `[Optional]` is both confusing and contradictory. `[Optional]` indicates that the parameter doesn't have to be provided, while `out` and `ref` mean that the parameter will be used to return data to the caller (`ref` additionally indicates that the parameter may also be used to pass data into the method).

Thus, making it `[Optional]` to provide the parameter in which you will be passing back the method results doesn't make sense. In fact, the compiler will raise an error on such code. Unfortunately, it raises the error on method calls where the `[Optional]` parameter has been omitted, not the source of the problem, the method declaration.

Noncompliant Code Example

```
class MyClass
{
    public void DoStuff([Optional] ref int i) // Noncomplian
    {
        Console.WriteLine(i);
    }

    public static void Main()
    {
        new MyClass().DoStuff(); // This doesn't compile, CS
    }
}
```

Compliant Solution

```
class MyClass
{
  public void DoStuff(ref int i)
  {
    Console.WriteLine(i);
  }

  public static void Main()
  {
    var i = 42;
    new MyClass().DoStuff(ref i);
  }
}
```

Available In:

sonarlint | sonarcloud | sonarqube

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

Code Smell