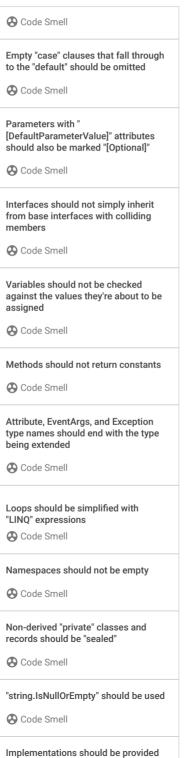


```
C# static code analysis
            Unique rules to find Bugs, Vulnerabilities, Security
            Hotspots, and Code Smells in your C# code
                                                                                      O Quick 52
Fix
                                                                  Security
                                                           (28)
                                                • Secu.
Hotspot
            6 Vulnerability (34)
                                 # Bug (76)
                                                                                                     Q
                                                 Tags
                                                                           Search by name..
                                        Conditionally executed code
                                                                                Analyze your code
                                        should be reachable
Empty "case" clauses that fall through
```



for "partial" methods

Code Smell

```
👬 Bug 🔷 Major 🕝
                        cwe unused suspicious pitfall
Conditional expressions which are always true or false can lead to dead code.
Such code is always buggy and should never be used in production.
Noncompliant Code Example
 public void Sample(bool b)
      bool a = false;
      if (a) // Noncompliant
      {
          DoSomething(); // never executed
      if (!a || b) // Noncompliant; "!a" is always "true", "b"
          DoSomething();
      else
          DoSomethingElse(); // never executed
      var d = "xxx";
      var res = d ?? "value"; // Noncompliant, d is always not
  }
Compliant Solution
```

```
public void Sample(bool b)
{
    bool a = false;
    if (Foo(a))
    {
        DoSomething();
    }
    if (b)
    {
        DoSomething();
    }
    else
    {
        DoSomethingElse();
    }
    var d = "xxx";
    var res = d;
}
```

Duplicate casts should not be made

Code Smell

Methods should not return values that are never used

Code Smell

Caller information arguments should not be provided explicitly

Code Smell

Method calls should not resolve ambiguously to overloads with "params"

Code Smell

Exceptions

This rule will not raise an issue in either of these cases:

• When the condition is a single const bool

```
const bool debug = false;
if (debug)
{
 // Print something
```

• When the condition is the literal true or false.

In these cases it is obvious the code is as intended.

- MITRE, CWE-570 Expression is Always False
- MITRE, CWE-571 Expression is Always True

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