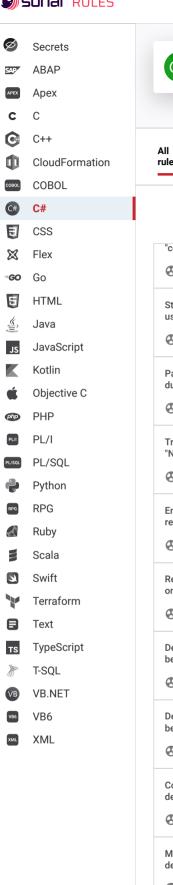
Quick 52 Fix





```
C# static code analysis
              Unique rules to find Bugs, Vulnerabilities, Security
              Hotspots, and Code Smells in your C# code
ΔII
     409
              6 Vulnerability (34)
                                      # Bug (76)
rules
  "const" instead
  Code Smell
 Strings or integral types should be
 used for indexers
 Code Smell
 Parameter names should not
 duplicate the names of their methods
 Code Smell
 Track use of
 "NotImplementedException"
 Code Smell
 Empty "default" clauses should be
 removed
  Code Smell
 Redundant property names should be
 omitted in anonymous classes
 A Code Smell
 Declarations and initializations should
 be as concise as possible
 Code Smell
 Default parameter values should not
 be passed as arguments
 Code Smell
 Constructor and destructor
 declarations should not be redundant
 A Code Smell
 Method parameters should be
 declared with base types
 Code Smell
 The simplest possible condition
 syntax should be used
 Code Smell
 Redundant parentheses should not be
 used
```

Code Smell

```
Q
           Tags
                                           Search by name..
Mutable, non-private fields should
                                                 Analyze your code
not be "readonly"
Rug OMinor @
Using the readonly keyword on a field means that it can't be changed after
initialization. However, when applied to collections or arrays, that's only partly true.
readon Ly enforces that another instance can't be assigned to the field, but it
cannot keep the contents from being updated. That means that in practice, the field
value really can be changed, and the use of readonly on such a field is misleading,
and you're likely to not be getting the behavior you expect.
This rule raises an issue when a non-private, readonly field is an array or
collection
Noncompliant Code Example
  public class MyClass
    public readonly string[] strings; // Noncompliant
    // ...
Compliant Solution
  public class MyClass
    public string[] strings;
    // ...
or
  public class MvClass
    public readonly ImmutableArray<string> strings;
    // ...
or
  public class MyClass
    private readonly string[] strings;
    // ...
 Available In:
```

sonarlint ⊕ | sonarcloud ↔ | sonarqube

Security

• Secu. Hotspot

(28)

"GC.SuppressFinalize" should not be invoked for types without destructors

Code Smell

Members should not be initialized to default values

Code Smell

Sequential tests should not check the same condition

Code Smell

Redundant modifiers should not be used

🕰 Code Smell

© 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