

































-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Kubernetes
-  Objective C
-  PHP
-  PL/I
-  PL/SQL
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  T-SQL
-  VB.NET
-  VB6
-  XML



# Go static code analysis

Unique rules to find Bugs, Security Hotspots, and Code Smells in your GO code

All rules 38 Bug 7 Security Hotspot 2 Code Smell 29

Tags

Search by name...

Hard-coded credentials are security-sensitive
Security Hotspot
Cognitive Complexity of functions should not be too high
Code Smell
String literals should not be duplicated
Code Smell
Functions should not be empty
Code Smell
All branches in a conditional structure should not have exactly the same implementation
Bug
"=+" should not be used instead of "+="
Bug
Related "if/else if" statements should not have the same condition
Bug
Identical expressions should not be used on both sides of a binary operator
Bug
All code should be reachable
Bug
Variables should not be self-assigned
Bug
Functions should not have identical implementations
Code Smell
Two branches in a conditional structure should not have exactly the same implementation
Code Smell

## Local variable and function parameter names should comply with a naming convention

Analyze your code

Code Smell Minor ? convention

Shared naming conventions allow teams to collaborate effectively. This rule raises an issue when a local variable or function parameter name does not match the provided regular expression.

### Noncompliant Code Example

With the default regular expression `^(_|[a-zA-Z0-9]+)$`:

```
func doSomething(my_param int) { // Noncompliant
    var local_int; // Noncompliant
    ...
}
```

### Compliant Solution

```
func doSomething(myParam int) {
    var local int;
    ...
}{code}
```

Available In: sonarcloud | sonarqube

<div><div>"switch" statements should not have too many "case" clauses</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Track uses of "FIXME" tags</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Redundant pairs of parentheses should be removed</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Nested blocks of code should not be left empty</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Functions should not have too many parameters</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Using hardcoded IP addresses is security-sensitive</div><div><div><div></div></div>Security Hotspot</div></div>
<div><div>Multi-line comments should not be empty</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Boolean checks should not be inverted</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Local variable and function parameter names should comply with a naming convention</div><div><div><div></div></div>Code Smell</div></div>
<div><div>Boolean literals should not be redundant</div><div><div><div></div></div>Code Smell</div></div>