

- 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



## PHP static code analysis

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

All rules **268**

Vulnerability **40**

Bug **51**

Security Hotspot **33**

Code Smell **144**

Tags

Search by name...



### Security Hotspot

Regular expression quantifiers and character classes should be used concisely

Code Smell

Character classes should be preferred over reluctant quantifiers in regular expressions

Code Smell

A subclass should not be in the same "catch" clause as a parent class

Code Smell

Jump statements should not be redundant

Code Smell

"catch" clauses should do more than rethrow

Code Smell

"&&" and "||" should be used

Code Smell

Boolean checks should not be inverted

Code Smell

Local variables should not be declared and then immediately returned or thrown

Code Smell

Unused local variables should be removed

Code Smell

"switch" statements should have at least 3 "case" clauses

Code Smell

### Loops with at most one iteration should be refactored

Analyze your code

Bug Major

A loop with at most one iteration is equivalent to the use of an `if` statement to conditionally execute one piece of code. No developer expects to find such a use of a loop statement. If the initial intention of the author was really to conditionally execute one piece of code, an `if` statement should be used instead.

At worst that was not the initial intention of the author and so the body of the loop should be fixed to use the nested `return`, `break` or `throw` statements in a more appropriate way.

#### Noncompliant Code Example

```
for ($i = 0; $i < 10; $i++) { // Noncompliant
    echo "i is $i";
    break;
}
...
for ($i = 0; $i < 10; $i++) { // Noncompliant
    if ($i == $x) {
        break;
    } else {
        echo "i is $i";
        return;
    }
}
```

#### Compliant Solution

```
for ($i = 0; $i < 10; $i++) {
    echo "i is $i";
}
...
for ($i = 0; $i < 10; $i++) {
    if ($i == $x) {
        break;
    } else {
        echo "i is $i";
    }
}
```

Available In:

sonarlint | sonarcloud | sonarqube


A "while" loop should be used instead of a "for" loop

 Code Smell


Overriding methods should do more than simply call the same method in the super class

 Code Smell

"empty()" should be used to test for emptiness

 Code Smell

Interface names should comply with a naming convention

 Code Smell

Return of boolean expressions should

© 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](#)