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...

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

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

Overriding methods should do more

Short-circuit logic should be used to prevent null pointer dereferences in conditionals

Analyze your code

Bug Major

When either the equality operator in a null test or the logical operator that follows it is reversed, the code has the appearance of safely null-testing the object before dereferencing it. Unfortunately the effect is just the opposite - the object is null-tested and then dereferenced *only* if it is null, leading to a guaranteed null pointer dereference.

Noncompliant Code Example

```
if ($obj == null && $obj->isOpen()) {
  echo "Object is open";
}

if ($obj != null || $obj->isOpen()) {
  echo "Object is not open";
}
```

Compliant Solution

```
if ($obj == null || $obj->isOpen()) {
  echo "Object is open";
}

if ($obj != null && !$obj->isOpen()) {
  echo "Object is not open";
}
```

Available In:

sonarlint | sonarcloud | sonarqube

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 not be wrapped into an "if-then-else" statement

☢ Code Smell

Boolean literals should not be redundant