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...

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

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

Code Smell

Overriding methods should do more

Useless "if(true) {...}" and "if(false){...}" blocks should be removed

Analyze your code

Bug Major cwe

`if` statements with conditions that are always false have the effect of making blocks of code non-functional. `if` statements with conditions that are always true are completely redundant, and make the code less readable.

There are three possible causes for the presence of such code:

- An if statement was changed during debugging and that debug code has been committed.
- Some value was left unset.
- Some logic is not doing what the programmer thought it did.

In any of these cases, unconditional `if` statements should be removed.

Noncompliant Code Example

```
if (true) {  // Noncompliant
  doSomething();
}
...
if (false) {  // Noncompliant
  doSomethingElse();
}
```

Compliant Solution

```
doSomething();
```

See

- MITRE, CWE-489 - Active Debug Code
- MITRE, CWE-570 - Expression is Always False
- MITRE, CWE-571 - Expression is Always True

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