- 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...

Configuration should not be changed dynamically
Code Smell

Class constructors should not create other objects
Code Smell

PHP parser failure
Code Smell

The names of methods with boolean return values should start with "is" or "has"
Code Smell

"php_sapi_name()" should not be used
Code Smell

Classes should not have too many lines of code
Code Smell

Deprecated features should not be used
Code Smell

Files should not contain inline HTML
Code Smell

Files should contain only one top-level class or interface each
Code Smell

Classes should not have too many fields
Code Smell

Track uses of "NOSONAR" comments
Code Smell

Statements should be on separate lines
Code Smell

Classes should not be coupled to too

Assertion arguments should be passed in the correct order

Analyze your code

Code Smell | Major | tests suspicious phpunit

The standard PHPUnit assertion methods such as `assertEquals`, expect the first argument to be the expected value and the second argument to be the actual value. Swap them, and your test will still have the same outcome (succeed/fail when it should) but the error messages will be confusing.

This rule raises an issue when the second argument to an assertions library method is a hard-coded value and the first argument is not.

Noncompliant Code Example

```
self::assertEquals($runner.getExitCode(), 0, "Unexpected exi
```

Compliant Solution

```
self::assertEquals(0, $runner.getExitCode(), "Unexpected exi
```

Available In:

sonarlint | sonarcloud | sonarqube

many other classes (Single Responsibility Principle)

Code Smell

"switch case" clauses should not have too many lines of code

Code Smell

Assignments should not be made from within sub-expressions

Code Smell

Files should not have too many lines of code

Code Smell