

- 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... 🔍

Vulnerability
Files that define symbols should not cause side-effects
Bug
Increment (++) and decrement (--) operators should not be used in a method call or mixed with other operators in an expression
Code Smell
Test methods should be discoverable
Code Smell
Use of namespaces should be preferred to "include" or "require" functions
Code Smell
Duplicate values should not be passed as arguments
Code Smell
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

Methods should not have identical implementations

Analyze your code

Code Smell

Major ?

confusing duplicate suspicious

When two methods have the same implementation, either it was a mistake - something else was intended - or the duplication was intentional, but may be confusing to maintainers. In the latter case, one implementation should invoke the other.

Noncompliant Code Example

```
class A {
    private const CODE = "bounteous";

    public function getCode() {
        doTheThing();
        return A::CODE;
    }

    public function getName() { // Noncompliant
        doTheThing();
        return A::CODE;
    }
}
```

Compliant Solution

```
class A {
    private const CODE = "bounteous";

    public function getCode() {
        doTheThing();
        return A::CODE;
    }

    public function getName() {
        return $this->getCode();
    }
}
```

Exceptions

Methods that are not accessors (getters and setters), with fewer than 2 statements are ignored.


Available In:

sonarlint


sonarcloud

sonarqube


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

---

© 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](#)