

- 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...



be used to end lines

Code Smell

More than one property should not be declared per statement

Code Smell

The "var" keyword should not be used

Code Smell

"<?php" and "<?=" tags should be used

Code Smell

File names should comply with a naming convention

Code Smell

Comments should not be located at the end of lines of code

Code Smell

Local variable and function parameter names should comply with a naming convention

Code Smell

Field names should comply with a naming convention

Code Smell

Lines should not end with trailing whitespaces

Code Smell

Files should contain an empty newline at the end

Code Smell

Modifiers should be declared in the correct order

Code Smell

An open curly brace should be located at the beginning of a line

Character classes should be preferred over reluctant quantifiers in regular expressions

Analyze your code

Code Smell Minor ? regex

Using reluctant quantifiers (also known as lazy or non-greedy quantifiers) in patterns can often lead to needless backtracking, making the regex needlessly inefficient and potentially vulnerable to [catastrophic backtracking](#). Particularly when using `. * ?` or `. + ?` to match anything up to some terminating character, it is usually a better idea to instead use a greedily or possessively quantified negated character class containing the terminating character. For example `<. + ?>` should be replaced with `<[^>]++>`.

Noncompliant Code Example

```
<. + ?> /  
/" . * ? " /
```

Compliant Solution

```
<[ ^> ]++> /  
/" [ ^" ] * + " /
```

or

```
<[ ^> ]++> /  
/" [ ^" ] * " /
```






Exceptions

This rule only applies in cases where the reluctant quantifier can easily be replaced with a negated character class. That means the repetition has to be terminated by a single character or character class. Patterns such as the following, where the alternatives without reluctant quantifiers are more complicated, are therefore not subject to this rule:

```
<! -- . * ? --> /  
- / \ * . * ? \ * / -
```

Available In:

sonarlint | sonarcloud | sonarqube

 Code Smell
An open curly brace should be located at the end of a line  Code Smell
Tabulation characters should not be used  Code Smell
Method and function names should comply with a naming convention  Code Smell
Creating cookies with broadly defined "domain" flags is security-sensitive  Security Hotspot