




 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



Java static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVA code

All rules 632

Vulnerability 53

Bug 154

Security Hotspot 36

Code Smell 389

Quick Fix 42

Tags ▾

Search by name... 🔍

Abstract class names should comply with a naming convention

Code Smell

Strings literals should be placed on the left side when checking for equality

Code Smell

Files should contain an empty newline at the end

Code Smell

Source code should be indented consistently

Code Smell

A close curly brace should be located at the beginning of a line

Code Smell

Close curly brace and the next "else", "catch" and "finally" keywords should be on two different lines

Code Smell

Close curly brace and the next "else", "catch" and "finally" keywords should be located on the same line

Code Smell

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

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

Functions should not be defined with a variable number of arguments

Code Smell

Repeated patterns in regular expressions should not match the empty string

Analyze your code

Bug

Minor

regex

A regex should never include a repetitive pattern whose body would match the empty string. This is usually a sign that a part of the regex is redundant or does not do what the author intended.

Noncompliant Code Example

```
"(?:)" // same as the empty regex, the '*' accomplishes nothing
"(:|x)" // same as the empty regex, the alternative has no effect
"(?:x)" // same as 'x*', the empty alternative has no effect
"(?:x|y)" // same as 'x*', the first alternative would always match
"(?:x?)" // same as 'x*'
"(?:x?)+ " // same as 'x*'
```

Compliant Solution

```
"x"
```

Available In:

sonarlint

sonarcloud

sonarqube

https://rules.sonarsource.com/java/RSPEC-5842

1/2

<div>Local-Variable Type Inference should be used</div> <div> Code Smell</div>
<div>Migrate your tests from JUnit4 to the new JUnit5 annotations</div> <div> Code Smell</div>
<div>Track uses of disallowed classes</div> <div> Code Smell</div>
<div>Track uses of "@SuppressWarnings" annotations</div> <div> Code Smell</div>