
































-  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	
Strings literals should be placed on the left side when checking for equality	
Files should contain an empty newline at the end	
Source code should be indented consistently	
A close curly brace should be located at the beginning of a line	
Close curly brace and the next "else", "catch" and "finally" keywords should be on two different lines	
Close curly brace and the next "else", "catch" and "finally" keywords should be located on the same line	
An open curly brace should be located at the beginning of a line	
An open curly brace should be located at the end of a line	
Tabulation characters should not be used	
Functions should not be defined with a variable number of arguments	

Parentheses should be removed from a single lambda input parameter when its type is inferred

Analyze your code

 Code Smell  Minor  java8

There are two possible syntaxes for a lambda having only one input parameter with an inferred type: with and without parentheses around that single parameter. The simpler syntax, without parentheses, is more compact and readable than the one with parentheses, and is therefore preferred.

**Note** that this rule is automatically disabled when the project's `sonar.java.source` is lower than 8.

Noncompliant Code Example

```
(x) -> x * 2
```

Compliant Solution

```
x -> x * 2
```

Available In:  
  

© 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](#)

<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>