

 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

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

Comma-separated labels should be used in Switch with colon case

Analyze your code

Code Smell

Info ?

java14

In Java 14 there is a new way to write cases in Switch Statement and Expression when the same action should be performed for different cases. Instead of declaring multiples branches with the same action, you can combine all of them in a single case group, separated with commas. It will result in a more concise code and improved readability.

This rule reports an issue when multiple cases in a Switch can be grouped into a single comma-separated case.

Noncompliant Code Example





```
// Switch Expression
int i = switch (mode) {
    case "a":
    case "b":
        yield 1;
    default:
        yield 3;
};

// Switch Statement
switch (mode) {
    case "a":
    case "b":
        doSomething();
        break;
    default:
        doSomethingElse();
}
```

Compliant Solution

```
// Switch Expression
int i = switch (mode) {
    case "a", "b":
        yield 1;
    default:
        yield 3;
};

// Switch Statement
switch (mode) {
    case "a", "b":
        doSomething();
        break;
    default:
        doSomethingElse();
}
```

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

```
// Or even better:
switch (mode) {
  case "a", "b" -> doSomething();
  default -> doSomethingElse();
}
```

See

- [JEP 361: Switch Expressions](#)

Available In:

sonarlint

| sonarcloud

| sonarqube

© 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](#)