sonar

RULES

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 rules632

Vulnerability53

Bug154

Security Hotspot36

Code Smell389

Quick Fix42

Tags ▾

Search by name... 🔍

Code Smell

Exit methods should not be called

Code Smell

HTTP response headers should not be vulnerable to injection attacks

Vulnerability

Members of Spring components should be injected

Vulnerability

Classes should not be loaded dynamically

Vulnerability

Equality operators should not be used in "for" loop termination conditions

Code Smell

"Bean Validation" (JSR 380) should be properly configured

Code Smell

Spring beans should be considered by "@ComponentScan"

Code Smell

Number patterns should be regular

Code Smell

Lazy initialization of "static" fields should be "synchronized"

Code Smell

Wildcard imports should not be used

Code Smell

Modulus results should not be checked for direct equality

Code Smell

Comparators should be "Serializable"

Code Smell

Collection constructors should not be used as java.util.function.Function

Analyze your code

Code Smell

Major ?

It is very common to pass a collection constructor reference as an argument, for example `Collectors.toCollection(ArrayList::new)` takes the `ArrayList::new` constructor. When the method expects a `java.util.function.Supplier` it is perfectly fine. However when the method argument type is `java.util.function.Function` it means that an argument will be passed to the constructor.

The first argument of Collections constructors is usually an integer representing its "initial capacity". This is generally not what the developer expects, but the memory allocation is not visible at first glance.

This rule raises an issue when a collection constructor is passed by reference as a `java.util.function.Function` argument.

Noncompliant Code Example

```
Arrays.asList(1, 2, 54000).stream().collect(Collectors.toMap
```

Compliant Solution

```
Arrays.asList(1, 2, 54000).stream().collect(Collectors.toMap
```

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

https://rules.sonarsource.com/java/RSPEC-5329

1/2

 Code Smell
"Serializable" classes should have a "serialVersionUID"  Code Smell
"switch" statements and expressions should not be nested  Code Smell
Constructors should only call non-overridable methods  Code Smell
Methods should not be too complex  Code Smell