

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... 🔍

Security Hotspot

Allowing user enumeration is security-sensitive

Allowing requests with excessive content length is security-sensitive

Disabling auto-escaping in template engines is security-sensitive

Allowing deserialization of LDAP objects is security-sensitive

Setting loose POSIX file permissions is security-sensitive

Formatting SQL queries is security-sensitive

Deprecated annotations should include explanations

Restricted Identifiers should not be used as Identifiers

Redundant constructors/methods should be avoided in records

Records should be used instead of ordinary classes when representing immutable data structure

"Stream.toList()" method should be used instead of "collectors" when unmodifiable list needed

Constant names should comply with a naming convention

Analyze your code

Code SmellCriticalconvention

Shared coding conventions allow teams to collaborate efficiently. This rule checks that all constant names match a provided regular expression.

Noncompliant Code Example

With the default regular expression `^[A-Z][A-Z0-9]*(_[A-Z0-9]+)*$`:

```
public class MyClass {
    public static final int first = 1;
}

public enum MyEnum {
    first;
}
```

Compliant Solution

```
public class MyClass {
    public static final int FIRST = 1;
}

public enum MyEnum {
    FIRST;
}
```

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-115

1/2

 Code Smell
Operator "instanceof" should be used instead of "A.class.isInstance()"  Code Smell
String multiline concatenation should be replaced with Text Blocks  Code Smell
Single-character alternations in regular expressions should be replaced with character classes  Code Smell
Reluctant quantifiers in regular