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...

Regexes containing characters subject to normalization should use the CANON_EQ flag
Code Smell

Regular expressions should not be too complicated
Code Smell

JUnit assertTrue/assertFalse should be simplified to the corresponding dedicated assertion
Code Smell

Only one method invocation is expected when testing runtime exceptions
Code Smell

Exception testing via JUnit ExpectedException rule should not be mixed with other assertions
Code Smell

"@Deprecated" code marked for removal should never be used
Code Smell

Vararg method arguments should not be confusing
Code Smell

Whitespace for text block indent should be consistent
Code Smell

'List.remove()' should not be used in ascending 'for' loops
Code Smell

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

"else" statements should be clearly matched with an "if"
Code Smell

Collections should not be modified while they are iterated

Analyze your code

Bug Major

Generally it's not recommended to modify a collection while it's iterated and most JDK collection implementations don't support it. This may sometimes lead to a ConcurrentModificationException and it could be the source of incorrect or unspecified behaviors in the code.

This rule raises an issue when a method modifies the size of a collection, while the same collection is iterated.

Noncompliant Code Example

```
public static void foo(List<String> lst) {
  for (String element : lst) {
    if (element.startsWith("x")) {
      lst.remove(element); // Noncompliant: lst size has b
    }
  }
}
```

Available In:

sonarlint | sonarcloud | sonarqube

"Class.forName()" should not load JDBC 4.0+ drivers

Code Smell

Java features should be preferred to Guava

Code Smell

Nullness of parameters should be guaranteed

Code Smell

"Integer.toHexString" should not be used to build hexadecimal strings

Code Smell