




 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... 🔍

should be omitted if they are declared in the same file

Code Smell

Switch arrow labels should not use redundant keywords

Code Smell

Text blocks should not be used in complex expressions

Code Smell

Pattern Matching for "instanceof" operator should be used instead of simple "instanceof" + cast

Code Smell

Call to Mockito method "verify", "when" or "given" should be simplified

Code Smell

Character classes should be preferred over reluctant quantifiers in regular expressions

Code Smell

Consecutive AssertJ "assertThat" statements should be chained

Code Smell

Chained AssertJ assertions should be simplified to the corresponding dedicated assertion

Code Smell

Exception testing via JUnit @Test annotation should be avoided

Code Smell

Escape sequences should not be used in text blocks

Code Smell

Simple string literal should be used for single line strings

Code Smell

"hashCode" and "toString" should not be called on array instances

Analyze your code

BugMajor?

While hashCode and toString are available on arrays, they are largely useless. hashCode returns the array's "identity hash code", and toString returns nearly the same value. Neither method's output actually reflects the array's contents. Instead, you should pass the array to the relevant static Arrays method.

Noncompliant Code Example

```
public static void main( String[] args )
{
    String argStr = args.toString(); // Noncompliant
    int argHash = args.hashCode(); // Noncompliant
}
```

Compliant Solution

```
public static void main( String[] args )
{
    String argStr = Arrays.toString(args);
    int argHash = Arrays.hashCode(args);
}
```

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-2116

1/2

<div>Boxed "Boolean" should be avoided in boolean expressions</div> <div> Code Smell</div>
<div>Type parameters should not shadow other type parameters</div> <div> Code Smell</div>
<div>"read(byte[],int,int)" should be overridden</div> <div> Code Smell</div>
<div>An iteration on a Collection should be performed on the type handled by the Collection</div> <div> Code Smell</div>