

































-  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... 🔍


 Bug


Min and max used in combination should not always return the same value




 Bug


Assignment of lazy-initialized members should be the last step with double-checked locking




 Bug


"String" calls should not go beyond their bounds




 Bug


Raw byte values should not be used in bitwise operations in combination with shifts




 Bug


Getters and setters should be synchronized in pairs




 Bug


Non-thread-safe fields should not be static



 Bug

"null" should not be used with "Optional"



 Bug

Unary prefix operators should not be repeated



 Bug

"+=" should not be used instead of "+="



 Bug


"read" and "readLine" return values should be used





 Bug


Inappropriate regular expressions should not be used

## Class members annotated with "@VisibleForTesting" should not be accessed from production code

 Code Smell

 Critical



 pitfall

@VisibleForTesting can be used to mark methods, fields and classes whose visibility restrictions have been relaxed more than necessary for the API to allow for easier unit testing.

Access to such methods, fields and classes only possible thanks to this relaxed visibility is fine for test code, but it should be avoided in production code. In production code these methods should be treated as if they are private.

Supported framework:

- Guava: com.google.common.annotations.VisibleForTesting
- AssertJ: org.assertj.core.util.VisibleForTesting
- Android: androidx.annotation.VisibleForTesting
- Apache Flink: org.apache.flink.annotation.VisibleForTesting

or any other annotation named VisibleForTesting

### Noncompliant Code Example

```
/** src/main/java/MyObject.java */

@VisibleForTesting String foo;

/** src/main/java/Service.java */

new MyObject().foo; // Noncompliant, foo is accessed from pr
```

### Compliant Solution




```
/** src/main/java/MyObject.java */

@VisibleForTesting String foo;

/** src/test/java/MyObjectTest.java */

new MyObject().foo; // Compliant, foo is accessed from test
```

Available In:





 |  | 

© 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-5803

1/2

 Bug
Conditionally executed code should be reachable  Bug
"notifyAll" should be used  Bug
Blocks should be synchronized on "private final" fields  Bug
Non-serializable objects should not be stored in "HttpSession" objects