
































-  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... 

statically
 Code Smell
Silly math should not be performed
 Code Smell
Classes named like "Exception" should extend "Exception" or a subclass
 Code Smell
Exceptions should be either logged or rethrown but not both
 Code Smell
Objects should not be created only to "getClass"
 Code Smell
Primitives should not be boxed just for "String" conversion
 Code Smell
Constructors should not be used to instantiate "String", "BigInteger", "BigDecimal" and primitive-wrapper classes
 Code Smell
"URL.hashCode" and "URL.equals" should be avoided
 Code Smell
Two branches in a conditional structure should not have exactly the same implementation
 Code Smell
Unused assignments should be removed
 Code Smell
"Object.wait(...)" should never be called on objects that implement "java.util.concurrent.locks.Condition"
 Code Smell
A field should not duplicate the name

Exceptions should not be created without being thrown

Analyze your code

-  Bug
-  Major
-  Quick Fix
-  error-handling

Creating a new `Throwable` without actually throwing it is useless and is probably due to a mistake.

Noncompliant Code Example

```
if (x < 0)
    new IllegalArgumentException("x must be nonnegative");
```





Compliant Solution

```
if (x < 0)
    throw new IllegalArgumentException("x must be nonnegative")
```

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](#)

<div>of its containing class</div> <div> Code Smell</div>
<div>JUnit4 @Ignored and JUnit5 @Disabled annotations should be used to disable tests and should provide a rationale</div> <div> Code Smell</div>
<div>Anonymous inner classes containing only one method should become lambdas</div> <div> Code Smell</div>
<div>"switch" statements should not have too many "case" clauses</div> <div> Code Smell</div>