




 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... 🔍

sensitive

Security Hotspot

Allowing requests with excessive content length is security-sensitive

Security Hotspot

Disabling auto-escaping in template engines is security-sensitive

Security Hotspot

Allowing deserialization of LDAP objects is security-sensitive

Security Hotspot

Setting loose POSIX file permissions is security-sensitive

Security Hotspot

Formatting SQL queries is security-sensitive

Security Hotspot

Deprecated annotations should include explanations

Code Smell

Restricted Identifiers should not be used as Identifiers

Code Smell

Redundant constructors/methods should be avoided in records

Code Smell

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

Code Smell

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

Code Smell

Operator "instanceof" should be used instead of "A class instanceof"

### Exceptions should not be thrown in finally blocks

Analyze your code

Code Smell

Critical

error-handling cert suspicious

Throwing an exception from within a finally block will mask any exception which was previously thrown in the try or catch block, and the masked's exception message and stack trace will be lost.

#### Noncompliant Code Example

```
try {
    /* some work which end up throwing an exception */
    throw new IllegalArgumentException();
} finally {
    /* clean up */
    throw new RuntimeException(); // Noncompliant; masks
}
```

#### Compliant Solution

```
try {
    /* some work which end up throwing an exception */
    throw new IllegalArgumentException();
} finally {
    /* clean up */
}
}
```

See

CERT, ERR05-J

- Do not let checked exceptions escape from a finally block

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-1163

1/2

<div>instead of <code>A.class.isInstance()</code></div> <div> Code Smell</div>
<div>String multiline concatenation should be replaced with Text Blocks</div> <div> Code Smell</div>
<div>Single-character alternations in regular expressions should be replaced with character classes</div> <div> Code Smell</div>
<div>Reluctant quantifiers in regular expressions should be followed by an expression that can't match the empty string</div> <div> Code Smell</div>