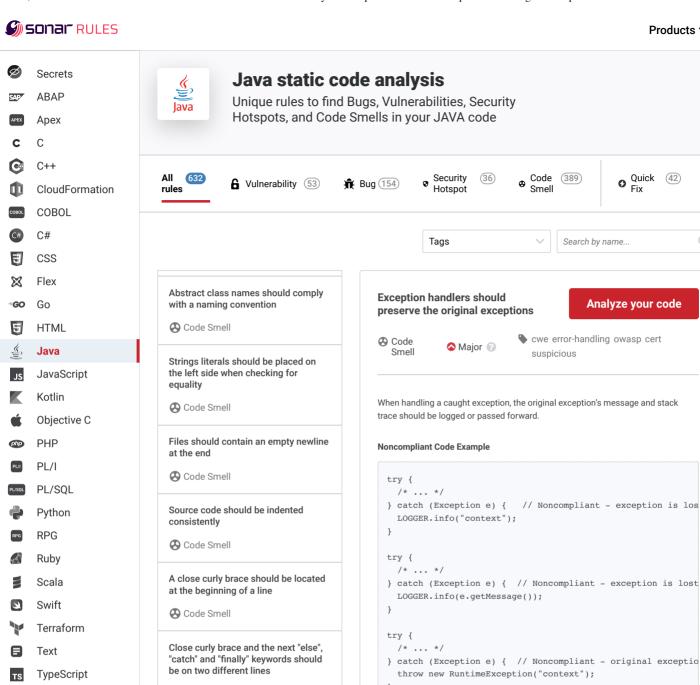
Products >

Quick 42 Fix

Q



Code Smell

Code Smell

Code Smell

at the end of a line Code Smell

Code Smell

Code Smell

used

Close curly brace and the next "else",

"catch" and "finally" keywords should

An open curly brace should be located

An open curly brace should be located

Tabulation characters should not be

Functions should not be defined with a variable number of arguments

be located on the same line

at the beginning of a line

## **Compliant Solution**

```
try {
 /* ... */
} catch (Exception e) {
 LOGGER.info(e); // exception is logged
try {
} catch (Exception e) {
 throw new RuntimeException(e); // exception stack trace
 /* ... */
} catch (RuntimeException e) {
 doSomething();
 throw e; // original exception passed forward
} catch (Exception e) {
  throw new RuntimeException(e); // Conversion into uncheck
```

## Exceptions

## https://rules.sonarsource.com/java/RSPEC-1166

T-SQL VB.NET

VB6

XML

Local-Variable Type Inference should be used

A Code Smell

Migrate your tests from JUnit4 to the new JUnit5 annotations

Code Smell

Track uses of disallowed classes

Code Smell

Track uses of "@SuppressWarnings" annotations

Code Smell

InterruptedException, NumberFormatException,
DateTimeParseException, ParseException and MalformedURLException
exceptions are arguably used to indicate nonexceptional outcomes. Similarly,

exceptions are arguably used to indicate nonexceptional outcomes. Similarly, handling NoSuchMethodException is often required when dealing with the Java reflection API.

Because they are part of Java, developers have no choice but to deal with them. This rule does not verify that those particular exceptions are correctly handled.

```
int myInteger;
try {
  myInteger = Integer.parseInt(myString);
} catch (NumberFormatException e) {
  // It is perfectly acceptable to not handle "e" here
  myInteger = 0;
}
```

Furthermore, no issue will be raised if the exception message is logged with additional information, as it shows that the developer added some context to the error message.

```
try {
   /* ... */
} catch (Exception e) {
   String message = "Exception raised while authenticating us
   LOGGER.warn(message); // Compliant - exception message log
}
```

## See

- OWASP Top 10 2021 Category A9 Security Logging and Monitoring Failures
- OWASP Top 10 2017 Category A10 Insufficient Logging & Monitoring
- CERT, ERR00-J. Do not suppress or ignore checked exceptions
- MITRE, CWE-778 Insufficient Logging

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