sonar RULES

Products ⌄

- 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...

Bug

AssertJ assertions with "Consumer" arguments should contain assertion inside consumers

Bug

The regex escape sequence \cX should only be used with characters in the @-_ range

Bug

Regular expressions should not overflow the stack

Bug

Tests method should not be annotated with competing annotations

Bug

Assertions should not be used in production code

Bug

DateTimeFormatters should not use mismatched year and week numbers

Bug

Unicode Grapheme Clusters should be avoided inside regex character classes

Bug

Case insensitive Unicode regular expressions should enable the “UNICODE_CASE” flag

Bug

Assertions should not compare an object to itself

Bug

Regex alternatives should not be redundant

Bug

Alternatives in regular expressions

Jump statements should not occur in "finally" blocks

Analyze your code

Bug Critical cwe error-handling cert

Using `return`, `break`, `throw`, and so on from a `finally` block suppresses the propagation of any unhandled `Throwable` which was thrown in the `try` or `catch` block.

This rule raises an issue when a jump statement (`break`, `continue`, `return`, `throw`, and `goto`) would force control flow to leave a `finally` block.

Noncompliant Code Example

```
public static void main(String[] args) {
  try {
    doSomethingWhichThrowsException();
    System.out.println("OK");   // incorrect "OK" message is
  } catch (RuntimeException e) {
    System.out.println("ERROR");  // this message is not sho
  }
}

public static void doSomethingWhichThrowsException() {
  try {
    throw new RuntimeException();
  } finally {
    for (int i = 0; i < 10; i ++) {
      //...
      if (q == i) {
        break; // ignored
      }
    }

    /* ... */
    return;      // Noncompliant - prevents the RuntimeExcep
  }
}
```

Compliant Solution

```
public static void main(String[] args) {
  try {
    doSomethingWhichThrowsException();
    System.out.println("OK");
  } catch (RuntimeException e) {
    System.out.println("ERROR");  // "ERROR" is printed as e
  }
}

public static void doSomethingWhichThrowsException() {
  try {
    throw new RuntimeException();
  } finally {
```

should be grouped when used with anchors

Bug

AssertJ methods setting the assertion context should come before an assertion

Bug

AssertJ configuration should be applied

Bug

JUnit5 test classes and methods should not be silently ignored

Bug

```
    for (int i = 0; i < 10; i ++) {
      //...
      if (q == i) {
        break; // ignored
      }
    }

    /* ... */
  }
}
```

See

- MITRE, CWE-584 - Return Inside Finally Block
- CERT, ERR04-J. - Do not complete abruptly from a finally block

Available In:

sonarlint sonarcloud sonarqube