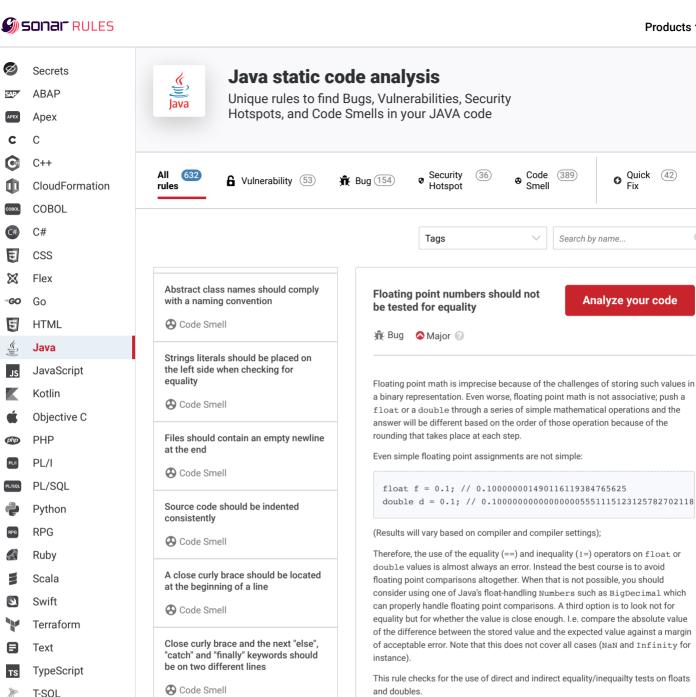


Products ✓

O Quick 42



can properly handle floating point comparisons. A third option is to look not for equality but for whether the value is close enough. I.e. compare the absolute value of the difference between the stored value and the expected value against a margin of acceptable error. Note that this does not cover all cases (NaN and Infinity for

This rule checks for the use of direct and indirect equality/inequality tests on floats

Noncompliant Code Example

```
float myNumber = 3.146;
if ( myNumber == 3.146f ) { //Noncompliant. Because of float
  // ...
if ( myNumber != 3.146f ) { //Noncompliant. Because of float
if (myNumber < 4 | | myNumber > 4) { // Noncompliant; indirec
float zeroFloat = 0.0f;
if (zeroFloat == 0) { // Noncompliant. Computations may end
```

Exceptions

Since NaN is not equal to itself, the specific case of testing a floating point value against itself is a valid test for NaN and is therefore ignored. Though using Double.isNan method should be preferred instead, as intent is more explicit.

Code Smell

Local-Variable Type Inference should be used

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

```
float f;
double d;
if(f!= f) { // Compliant; test for NaN value
   System.out.println("f is NaN");
} else if (f!= d) { // Noncompliant
   // ...
}

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