




 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

Code Smell
Unused "private" classes should be removed
Code Smell
"Stream.peek" should be used with caution
Code Smell
"Map.get" and value test should be replaced with single method call
Code Smell
"@RequestMapping" methods should not be "private"
Code Smell
Raw types should not be used
Code Smell
"Arrays.stream" should be used for primitive arrays
Code Smell
Printf-style format strings should be used correctly
Code Smell
Assertion arguments should be passed in the correct order
Code Smell
Ternary operators should not be nested
Code Smell
"writeObject" should not be the only "synchronized" code in a class
Code Smell
Reflection should not be used to increase accessibility of classes, methods, or fields
Code Smell

Tags ▾

Search by name... 🔍

Assertions should not compare an object to itself

Analyze your code

Bug

Major

tests

Assertions comparing an object to itself are more likely to be bugs due to developer's carelessness.

This rule raises an issue when the actual expression matches the expected expression.

Noncompliant Code Example

```
assertThat(actual).isEqualTo(actual); // Noncompliant
```

Compliant Solution

```
assertThat(actual).isEqualTo(expected);
```





Exceptions

In a unit test validating the `equals(...)` and `hashCode()` methods, it's legitimate to compare an object to itself. This rule does not raise an issue for `isEqualTo`, `assertEquals` or `hasSameHashCodeAs` when the unit test name contains (case insensitive): `equal`, `hash_?code`, `object_?method`. For example, in tests with the following names: `test_equals`, `testEqual`, `test_hashCode`, `test_hash_code`, `test_object_methods`.

```
class MyClassTest {
    @Test
    void test_equals_and_hash_code() {
        MyClass obj = new MyClass();
        assertThat(obj).isEqualTo(obj); // Compliant
        assertThat(obj).hasSameHashCodeAs(obj); // Compliant
    }
}
```

Available In:

sonarlint | sonarcloud | sonarqube


Static fields should not be updated in constructors  Code Smell
"Thread.sleep" should not be used in tests  Code Smell
"entrySet()" should be iterated when both the key and value are needed  Code Smell
"DateUtils.truncate" from Apache Commons Lang library should not be used