




 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... 🔍

Abstract class names should comply with a naming convention

Code Smell

Strings literals should be placed on the left side when checking for equality

Code Smell

Files should contain an empty newline at the end

Code Smell

Source code should be indented consistently

Code Smell

A close curly brace should be located at the beginning of a line

Code Smell

Close curly brace and the next "else", "catch" and "finally" keywords should be on two different lines

Code Smell

Close curly brace and the next "else", "catch" and "finally" keywords should be located on the same line

Code Smell

An open curly brace should be located at the beginning of a line

Code Smell

An open curly brace should be located at the end of a line

Code Smell

Tabulation characters should not be used

Code Smell

Functions should not be defined with a variable number of arguments

Code Smell

Mockito argument matchers should be used on all parameters

Analyze your code

BugMajor?tests mockito

Mockito provides *argument matchers* and *argument captors* for flexibly stubbing or verifying method calls.

Mockito.verify(), Mockito.when(), Stubber.when() and BDDMockito.given() each have overloads with and without argument matchers.

However, if argument matchers or captors are used only on some of the parameters, all the parameters need to have matchers as well, otherwise an InvalidUseOfMatchersException will be thrown.

This rule consequently raises an issue every time matchers are not used on all the parameters of a stubbed/verified method.

Noncompliant Code Example

```
@Test
public void myTest() {
    given(foo.bar(anyInt(), i1, i2)).willReturn(null); // Noncompliant
    when(foo.baz(eq(val1), val2)).thenReturn("hi"); // Noncompliant
    doThrow(new RuntimeException()).when(foo).quux(intThat(x - verify(foo).bar(i1, anyInt(), i2); // Noncompliant
    ArgumentCaptor<Integer> captor = ArgumentCaptor.forClass(Integer.class);
    verify(foo).bar(captor.capture(), i1, any()); // Noncompliant
}
```

Compliant Solution

```
@Test
public void myTest() {
    given(foo.bar(anyInt(), eq(i1), eq(i2))).willReturn(null);
    when(foo.baz(val1, val2)).thenReturn("hi");
    doThrow(new RuntimeException()).when(foo).quux(intThat(x - verify(foo).bar(eq(i1), anyInt(), eq(i2));
    ArgumentCaptor<Integer> captor = ArgumentCaptor.forClass(Integer.class);
    verify(foo).bar(captor.capture(), any(), any());
}
```

See

- Mockito documentation - argument matchers
- {rule:java:S6068} - Call to Mockito method "verify", "when" or "given" should be simplified

Available In:

sonarlint

sonarcloud

sonarqube

https://rules.sonarsource.com/java/RSPEC-6073

1/2

<div>Local-Variable Type Inference should be used</div> <div> Code Smell</div>
<div>Migrate your tests from JUnit4 to the new JUnit5 annotations</div> <div> Code Smell</div>
<div>Track uses of disallowed classes</div> <div> Code Smell</div>
<div>Track uses of "@SuppressWarnings" annotations</div> <div> Code Smell</div>

© 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](#)