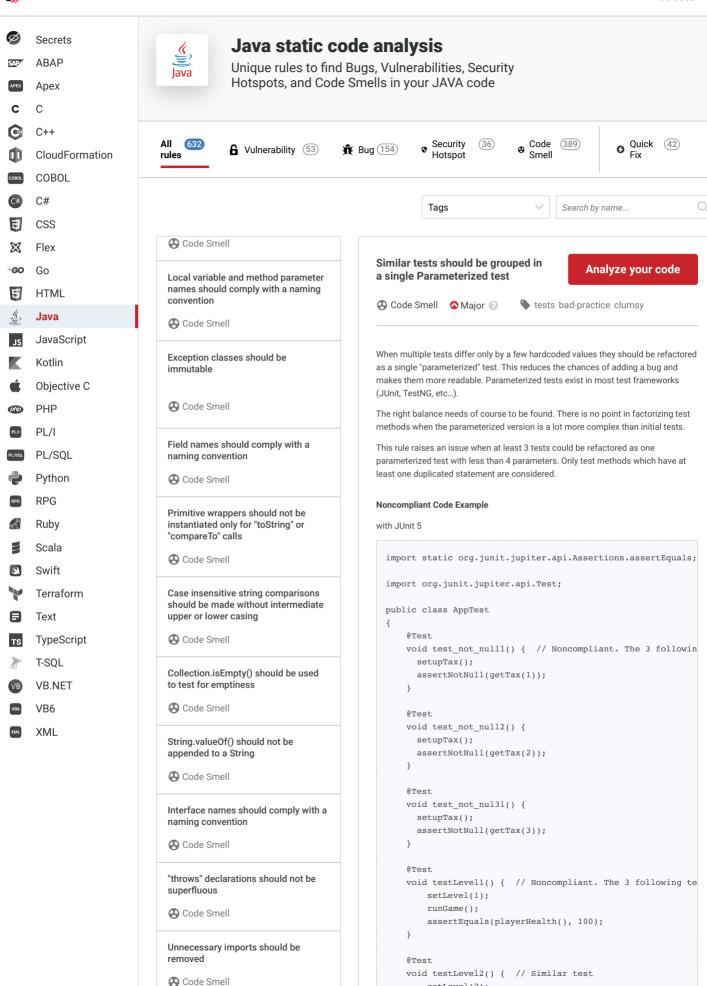


Products ✓



setLevel(2);
runGame();

assertEquals(playerHealth(), 200);

Return of boolean expressions should

not be wrapped into an "if-then-else" statement

A Code Smell

Boolean literals should not be redundant

Code Smell

Modifiers should be declared in the correct order

Code Smell

Empty statements should be removed

Code Smell

Class variable fields should not have

```
@Test
    void testLevel3() { // Similar test
       setLevel(3):
       runGame();
        assertEquals(playerHealth(), 300);
}
```

## **Compliant Solution**

```
import static org.junit.jupiter.api.Assertions.assertEquals;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;
public class AppTest
   @ParameterizedTest
   @ValueSource(ints = {1, 2, 3})
   void test_not_null(int arg) {
    setupTax();
     assertNotNull(getTax(arg));
    @ParameterizedTest
   @CsvSource({
        "1, 100",
        "2, 200",
       "3, 300",
   })
    void testLevels(int level, int health) {
        setLevel(level);
        runGame();
        assertEquals(playerHealth(), health);
}
```

## See

- Modern Best Practices for Testing in Java Philipp Hauer
- JUnit 5 documentation Parameterized tests
- Writing Parameterized Tests With JUnit 4
- TestNG documentation Parameters

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