Rapport sur les modifications du code

Table des matières

1. Introduction

Après avoir analysé le code de JUnit4 pour la première partie du rapport, j'ai apporté des modifications au code de l'application en utilisant SonarQube afin de déterminer les bugs ainsi que leurs degrés d'importance.

2. Travail réalisé

2.1. Code implémenté

Pour cette partie du projet, j'ai décidé de concentrer mes efforts sur la correction des bugs repérés par SonarQube ainsi que quelques 'Code smells'.

Sur la branche GareinGaetan du projet Git, on trouve 2 commits:

- le premier réfère à quelques code smells :
  - Dans une classe abstraite 'TestCase.java' -> passage de méthodes public en protected
  - Utilisation d'un logger à la place de 'System.out.println'

- le second réfère à la correction de la majorité des bugs majeurs repéré par Sonar
  - "Replace this assertion to not have the same actual and expected expression" : création d'objet distincts afin de les comparer
  - "Either re-interrupt this method or rethrow the "InterruptedException" that can be caught here" Compléter "InterruptedException" avec un autre catch

2.2. Piste d’amélioration

J'ai essayé de corrigé le bugs critique : "Don't use assertEquals() inside a try-catch catching an AssertionError", 18 bugs sont équivalents

Code d'origine :

```
try {
    assertEquals(new Object(), new Object());
} catch (AssertionFailedError e) {
    return;
}
fail();
```

Code testé :

```
AssertionFailedError thrown = Assertions.assertThrows(
        AssertionFailedError.class,
        () -> assertEquals(new Object(), new Object())
);
```

Le code test lève une erreur :
Erreur :
\junit4\src\test\java\junit\tests\framework\AssertTest.java:[58,19] error: lambda expressions are not supported in -source 7

Je n'ai malheureusement pas été en capacité de contourner ce problème malgré mes nombreux essais.

Pour le bug critique : "Change the assertion arguments to not compare dissimilar types" ayant 3 équivalents,
Les tests relevant ce bug comparent 2 arguments de types différents afin de vérifier qu'ils lèvent une erreur, n'ayant pas pu corrigé le bug énoncé précédemment,
Je n'ai pas pu faire la vérification.

3. Conclusion

Pour conclure, je me suis concentré sur les bugs majeurs et non les codes smells, SonarQube est un outil très utile pour corriger les bugs et les vulnérabilités d'une application, pour apprendre et appliquer les bonnes pratiques de développement.

Lien du Git : https://github.com/GaetanGarein/junit4/tree/main