

TP2 Fiabilité logicielle – couverture de tests avec JaCoCo

ARTAUD Cyril, DRAOU Meryem

1. Premiers pas avec JaCoCo

1.1. Calcul de la couverture par JaCoCo

1) Après lancement des tests on en déduit que :

- La couleur verte indique que l'instruction est couverte (instruction exécutée).
- La couleur rouge indique que l'instruction n'est pas couverte (instruction qui n'a pas été exécutée).
- La couleur jaune indique que la condition n'est pas totalement couverte (instruction partiellement exécutée). Si une condition est un jaune c'est que toutes les branches ne sont pas couvertes.

2) Les taux de couverture pour une classe, les méthodes, les lignes et les branchements correspondent aux taux de ligne de code testé.

En effet, si une méthode n'est pas entièrement couverte alors la classe en question ne le sera pas.

Le taux de couverture des branchements correspond aux différents cas possibles qui ont été couverts.

Il est possible de couvrir toutes les instructions de la classe Palindrome, il suffit de tester le cas où on a un mot de taille impaire. Le test qui est @Disabled permettra de couvrir cette instruction ce qui permettra de couvrir à 100 % les instructions de Palindrome.

Pour PartialCoverage il est impossible de couvrir ses instructions à 100 %. Malgré que nous donnons des valeurs pour x et y en paramètre ces valeurs seront changées au début de la méthode x vaudra 1 et y vaudra -1, la boucle for qui utilisera i comme indice commencera à la valeur de x, donc 1, et nous ne passons pas dans la boucle car $i \geq y$. Donc l'instruction dans la boucle ne sera pas couverte car la condition de sortie de boucle sera vraie au premier parcours.

4) Certaines instructions booléennes sont jaunes : elles ne sont pas couvertes entièrement. En effet, certaines branches ne sont pas couvertes.

5) Il est possible de couvrir toutes les instructions de la classe Palindrome : il faut couvrir le

cas possible d'un mot de longueur 1, et l'instanciation de la classe Palindrome en vérifiant le fait qu'une instance de type Palindrome est bien de ce type.

6) La couverture est bien meilleure, nous observons que notre test couvre le cas où un mot est longueur 1 (il est initialement de longueur impaire et récursion utilisée ici).

2. Etude de couverture de code

2.1. Couverture des classes de l'application Complex

Lors de l'analyse de couverture de la classe Complex nous remarquons que seulement 2 méthodes ne sont pas 100 % couvertes. La méthode `isZero()` est partiellement couverte car toutes les possibilités de sa décision ne sont pas couvertes et la méthode `toString()` n'étaient pas couverte.

Nous ajoutons ensuite le test `testZeroAndFalse()` qui permet que `isZero()` soit couvert à 100 %, et nous ajoutons le test `testToString()` pour couvrir `toString()`. Avec ces tests nous obtenons un taux de couverture de 100 % sur la classe Complex.

2.2 Analyse d'un code inconnu

1) Tous les tests passent ici. Un objet de type `StringArray` est créé pour tester la classe `StringArray`, il faut les initialiser au début des tests (sinon redondance de code). La méthode `size()` est utilisée pas été testée ici.

2) On constate que la couverture n'est pas totale. Nous observons que toutes les méthodes ne sont pas testées et/ou les méthodes testées ne le sont pas à 100 %.

Les méthodes `indexOf()` et `size()` sont couvertes. On constate que les méthodes `getString()`, `getList()` et notre constructeur ne sont pas totalement couverts.

5) Nous observons que le taux de couverture du constructeur avec cette suite de tests est de 89 %.

6) La couverture du constructeur n'est pas totale : le constructeur tel qu'il est ici n'est pas optimal (possibilité de le modifier pour une meilleure couverture). En effet, la couverture est toujours la même que précédemment.

7) Ce n'est pas suffisant, il serait éventuellement possible d'améliorer cela en modifiant plusieurs lignes de codes. En effet, il n'est pas vérifié ici si les éléments sont des doublons ou non. Il faudrait ajouter une logique supplémentaire pour vérifier cela dans notre code.

2.3 Files à double extrémité

3) On observe que la couverture n'est pas maximale.

5) Nous avons effectué les modifications nécessaires : notre taux de couverture est à 100 %.