Faculté des Sciences
Aix*Marseille Université

Département Informatique et Interactions
Année 2016-2017

Master 2 FSIL

Fiabilité Logicielle 1

TP Couverture de Tests

Le but de ce TP est d'utiliser l'outil de couverture de test *Emma* et d'en comprendre le fonctionnement et les limites.

1 Utilisation d'un outil de couverture

Installer le plugin emma [1] via la marketplace d'eclipse s'il n'est pas déjà installé.

1. Reprendre vos tests du TP1 sur la fonction *isPresentArticle* (pour chacune des classes données) et calculer la couverture obtenue. Votre suite de tests est-elle une bonne suite de tests ?

2. Récupérer les classes *StringArray.java* et *StringArrayTest.java* sur le site du cours.

   (a) Vérifier que les tests ne causent pas d'erreur *JUnit*.

   (b) Utiliser *eclemma* pour vérifier la couverture des tests. Conclusion ?

   (c) Ajouter un test qui couvre la méthode *getString* et un test avec le tableau {"ab", "ab"} couvrant la duplication. Résultat ?

   (d) Ajouter un test qui prend le i^{ème} élément de la liste *slist1* et vérifie que c'est bien le i^{ème} élément de *array1*.

   (e) La classe *StringArray* contient un bug. (a) À partir de l'analyse de la couverture de code, étendre les tests pour le faire apparaitre (b) Identifier le problème, (c) Corriger le problème et ajouter si-besoin des tests afin d'atteindre un taux de couverture de 100%.

2 Couverture du triangle

Il s'agit de reprendre et étendre l'exercice du triangle pour étudier la couverture de code.

Spécification de la classe:

- La classe `Triangle` contient un constructeur `Triangle(double a, double b, double c)` qui initialise un objet de côtés a, b, c et lève une exception de type `IllegalArgumentException` lorsque les paramètres ne forment pas un triangle.

[1] voir `http://eclemma.org/` pour la documentation

- La classe contient une fonction publique `int type()` qui renvoie un des 7 types suivants:

  0. scalène (3 côtés de longueur $\neq$) acutangle (que des angles aigus $< 90°$)
  1. scalène obtusangle (avec un angle $> 90°$)
  2. scalène rectangle (avec un angle droit $= 90°$)
  3. isocèle (exactement deux côtés égaux) acutangle
  4. isocèle obtusangle
  5. isocèle rectangle
  6. équilatéral (tous les côtés de même longueurs)

Vous pouvez utiliser les formules vues en cours et la page wikipédia `https://en.wikipedia.org/wiki/Triangle` pour détecter les différents cas.

Tests

- Écrire une suite de tests pour la classe *Triangle*, puis écrire la classe *Triangle* et enfin exécuter les tests (selon la méthode du *Test Driven Development*).
- Vérifier la couverture des tests avec Emma. Ajouter les tests nécessaires afin d'atteindre une couverture proche de 100%. Étudier la redondance éventuelle de vos tests en supprimant ou refactorisant un ou plusieurs tests puis en recalculant la couverture.

Entrée-Sortie Lorsque vous aurez terminé la partie précédente, complétez la classe Triangle avec les fonctions d'entrée-sortie [2] suivantes :

- `public static Triangle read(String filename) throws IOException` qui lit un fichier texte contenant les résultats des mesures des trois cotés d'un triangle (une valeur par ligne) et affecte les attributs privés *coteA, coteB, coteC* de type *double* de la classe aux valeurs lues. Si le fichier n'existe pas ou s'il ne correspond pas à trois valeurs de type double, la méthode déclenche une exception de type *IOException*.
- `public static void write(Triangle t, String filename) throws IOException` qui écrit dans le fichier *filename* les côtés du triangle *t*, de manière à être compatible avec la fonction Triangle.read()

Ajoutez autant de tests que nécessaires afin de vérifier les deux fonctions ajoutées. Essayer de conserver une couverture de code proche 100%.

[2] Remarque: le tutorial Java d'Oracle donne toutes les informations sur Java 8 (par exemple sur les entrées-sorties avec java.nio, voir `http://docs.oracle.com/javase/tutorial/essential/io/file.html` en anglais et `http://www.jmdoudoux.fr/java/dej/chap-nio2.htm` en français).

3 Travail à rendre

À la fin du TP (3/10 à 12h), vous enverrez par mail un zip contenant vos sources (StringArray.java et son test et Triangle.java et son test). Vous pouvez ajouter un rapport en pdf répondant aux questions du tp, expliquant votre démarche et/ou des points techniques.

- Destinataire : romaric.duvignau@lif.univ-mrs.fr
- Sujet : [FIAB1] NOM prénom
- Pièces jointes "NOM prénom.zip", "NOM prénom.pdf", ...