TP2 : Test en isolation avec Mockito

Master1 info S2
TP SVL – 2011-2012

Ce TP utilise l'outil Mockito, qui permet de générer automatiquement le code des doublures à partir de leur spécification. Toutes les informations requises sont sur la page *documents* de SVL, rubrique *Documents*, puis *concernant Mockito*.

Le travail est bien sûr à réaliser dans une approche TDD : astreignez-vous à écrire d'abord les tests, puis le code, et à progresser comportement par comportement en réusinant le code au fur et à mesure. Pour vous y aider :

- les premiers scénarios vous sont fournis ;
- aucun diagramme de classes ne vous est donné pour ne pas vous imposer une architecture testable a priori. Un des buts du TP est justement de vous faire trouver cette architecture par vous-même.

Exercice 1 : Jeu de dés

On s'intéresse à des jeux de hasard et d'argent (l'auteur du TP n'ayant jamais mis les pieds dans un casino, ce jeu n'est pas forcément très réaliste).

1.1 : Spécification informelle

Le joueur possède une somme d'argent qui lui permet de jouer à un jeu (la somme étant physiquement étalée devant lui, on suppose que le jeu n'a aucun contrôle à effectuer sur le montant de la mise).

Le jeu qui nous intéresse se joue avec 2 dés et une banque qui gère les pertes et les gains. Les dés sont du modèle classique à tirage aléatoire entre 1 et 6.

La banque est censée être toujours solvable. Néanmoins, il arrive que le casino soit débordé par un joueur chanceux et n'arrive plus à suivre : la banque "saute". Le gain est quand même donné au joueur, mais le jeu ferme immédiatement.

La règle du jeu est la suivante. On ne peut jouer qu'à un jeu ouvert. Le joueur qui joue fait un pari en misant une somme. Il est débité du montant de son pari qui est encaissé par la banque. Ensuite les 2 dés sont lancés. Si la somme des lancers vaut 7, alors le joueur gagne : la banque paye deux fois la mise, somme créditée au joueur. Si le pari a fait sauter la banque, le jeu ferme immédiatement. Si la somme des lancers ne vaut pas 7, le joueur a perdu sa mise.

1.2 : Sujet

Le sujet consiste à tester **en isolation** la méthode

```
public void jouer() throws ...
```

de la classe qui représente le jeu (paramètres de la méthode à adapter si besoin). Le test sera donc purement unitaire au niveau de la classe. **On ne demande pas d'écrire ni de tester les autres classes** : limitez-vous à des **interfaces** et utilisez des **doublures**. On ne demande pas non plus d'écrire les méthodes de la classe représentant le jeu qui ne seraient pas utilisées par `jouer` (argument désagréable : tout code non demandé vaudra des points en moins à la correction).

Les mocks devront rester les plus simples possibles : on peut lancer un dé, faire miser, créditer ou débiter un joueur, créditer ou débiter une banque (avec levée d'exception si la banque saute en cas de débit).

Commencer par le scénario le plus simple : le jeu est fermé. Écrire le test (il ne doit y avoir aucune interaction avec les autres acteurs). Ajouter les autres scénarios du même genre si vous en voyez.

Continuer par le scénario d'un joueur perdant : un test d'interaction est nécessaire. Vérifier notamment que le joueur a été débité de sa mise, laquelle a été créditée à la banque, et que le jeu reste ouvert.

Continuer avec les scénarios qui manquent.