



Le but de ce TP est de comprendre comment faire des tests d'intégration via des classes factices écrites à la main ou générées automatiquement. d'utiliser un outil de couverture de test et d'en comprendre le fonctionnement et les limites. Pour faciliter la prise en main, ma première partie correspond à l'exemple du cours.

## 1 Remplacer une classe par une classe factice

La classe *EuroCalc* dispose de méthodes qui donnent la valeur en euro d'une somme donnée dans une devise par exemple *Double dol2euro(Double somme)* à partir d'un attribut privé *conv* de type *Conversion*. Le constructeur de la classe *EuroCalc* prend en argument un élément de type *Conversion* et l'affecte à son attribut *conv*. La classe *Conversion* possède une méthode publique *Double getRate(String from, String to)* qui retourne le taux de change de la devise *from* dans la devise *to*. Cette valeur est calculée en consultant un fichier *tauxDeChange* dans laquelle chaque ligne est de la forme *devise,devise,taux*. On supposera que le fichier peut être modifié aléatoirement dans le temps (le système met à jour ce fichier toutes les minutes).

1. Ecrire la classe de test *EuroCalcTest*.
2. Ecrire les classes *EuroCalc* et *Conversion*.
3. Pour tester de manière fiable la classe *EuroCalc*, il n'est pas possible d'utiliser *Conversion* qui utilise une donnée dont la valeur varie dans le temps. Donc on demande d'écrire une classe *MockConversion* qui hérite de *Conversion*, mais redéfinit la fonction *getRate* de manière à ce qu'elle renvoie des valeurs déterminées à l'avance. Expliquer comment définir constructeur, attributs et méthode de cette classe. La réaliser et lancer les tests de la classe *EuroCalcTest*.

## 2 Le schéma d'inversion de controle

On remplace la classe *Conversion* par une interface *ICConversion*.

1. Ecrire le schéma de dépendance entre les classes.
2. Que faut-il modifier dans les classes précédentes ?
3. Réaliser les tests avec les nouvelles classes.

## 3 Utiliser un générateur d'objets factices

Deux générateurs classiques d'objets factices en java sont *easymock* et *JMock*. Chacun a ses avantages : syntaxe plus proche de l'invocation usuelle de méthodes, plus de précision

pour JMock. Voir <http://www.jmock.org/easymock-comparison.html> pour plus d'information.

1. Récupérer *JMock2* et l'installer dans son répertoire. Modifier son environnement pour pouvoir accéder les .jar. Lire la documentation de *JMock*.
2. Utiliser *JMock* pour générer une classe factice utilisée dans les tests *JUnit* (sans avoir à écrire explicitement la classe).
3. Utiliser toutes les constructions de *JMock*, en particulier pour faire plusieurs appels à une méthode, un nombre fixe, vérifier les valeurs retournées, le traitement d'exception, ...

Les classes *EuroCalc.java*, *EuroCalcTest.java*, *IConversion.java* du site du cours donnent des exemples d'utilisation de *JMock*.

## 4 Travail à rendre

La classe *Dicho* définit un dictionnaire français-italien. Un mot français pourra avoir plusieurs traductions. La classe *Dicho* possède les fonctions *isEmpty* (teste si le *Dicho* est vide), *addLine(AbstractLine line)* ajoute les traductions correspondant à une ligne de la forme *mot=mot1,...,motn* qui ajoute *mot* et ses traductions *mot1,...,motn* au dictionnaire - sans duplication si l'entrée *mot* existe-, *String getTrans(String mot)* qui affiche la traduction de *mot* (si plusieurs sont possibles on les concatène en séparant avec ,), *Dicho(AbstractLine [] text)* qui crée un dictionnaire à partir d'un tableau de lignes et *Dicho()* qui crée un dictionnaire vide.

La classe *AbstractLine* définit une suite de mots (ou token) séparés par = (pour signifier une traduction) ou , pour séparer des traductions du même mot et terminé par un symbole de fin. Elle possède les fonctions *int getNbToken()* qui donne le nombre de token de la ligne, *String getToken(int i)* qui renvoie le ieme token, *boolean isDefSep(String)* qui teste si un token est le séparateur de définition '=', (idem avec *isSep* pour ',') *isEmpty()* teste si la ligne est vide. **On n'écrira pas cette classe !**

1. Réaliser la classe de tests *DichoTest* pour *Dicho*.
2. Transformer la classe *AbstractLine* en une interface *IAbstractLine* et donner la manière dont *AbstractLine* serait modifiée.
3. Ecrire la classe *Dicho* et utiliser un générateur d'objets factices pour réaliser les tests.
4. Vous donnerez un fichier *ChoixdesTest.txt* qui explique le choix des tests et la manière de les réaliser.

Format de rendu : une archive .zip contenant les fichiers programmes et texte explicatifs. L'identifiant du projet est TPMock.