RPI

Année 2014 – 2015

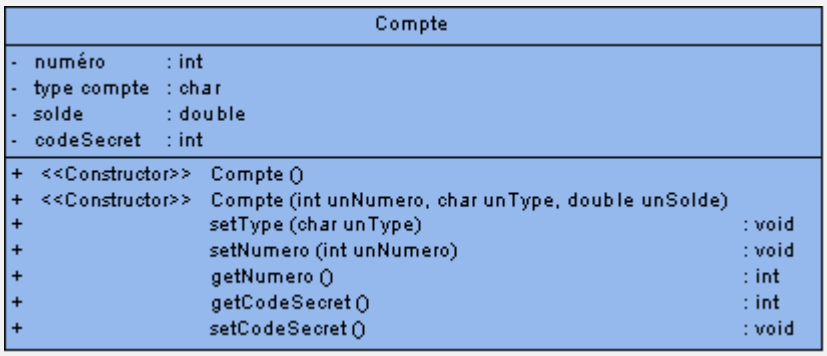
Java – TP2

Une petite banque régionale souhaite développer en Java un petit logiciel destiné à gérer ses comptes bancaires (exercice 1) et les opérations bancaires (Exercice2).

Remarque : Les exercices sont indépendants les uns des autres.

Exercice 1 : Constructeurs, getteurs, setteurs, surchage dans la classe Compte

La classe à créer est représentée dans le diagramme UML ci-dessous.



a) Créer la classe Compte avec ses attributs et un constructeur par défaut qui crée le compte avec :

- numero = 999999
- type de compte = ‘’
- solde = 0
- codeSecret = 0

b) Créer une méthode qui attribue un code secret automatiquement (un code secret est un chiffre généré aléatoirement compris entre 100 et 9999.
Conseil : utiliser la fonction la méthode random() de la classe Math.

Exemple : L’instruction « `int i = (int)(Math.random()*30+10)` ; » va affecter à la variable i, un nombre aléatoire compris entre 10 et 40 donc compris dans un intervalle : [10;40[

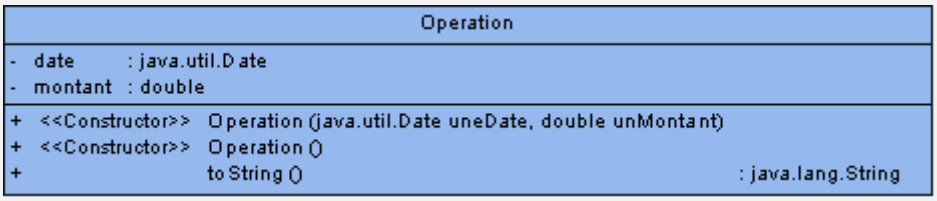
c) Créer un second constructeur (Surcharge) qui reçoit un numéro, un type de compte égal à ‘D’ (dépôt) ou ‘E’ (Epargne) et un solde. Ce constructeur déterminera un code secret automatiquement.
Si l’argument reçu est différent de ‘D’ et ‘E’, le type de compte prendra la valeur par défaut : ‘ ‘.

d) Créer des modificateurs (setteurs) pour les propriétés numéro et type de compte

e) Créer des accesseurs (getteurs) pour le numéro et le code secret.

f) Après avoir mis tous les commentaires utiles dans votre source, générer la documentation java pour la classe « Compte »

Exercice 2 : Un tableau d’opérations



Par souci de simplification et contrairement à ce diagramme UML, nous considérons une date au format chaîne de caractères (String)

a) Créer la classeOperation avec 2 constructeurs :

- le premier reçoit 2 arguments et valorise les attributs date et montant
- le second ne reçoit qu'un montant en argument et la date prend pour valeur la date du jour

b) Créez un programme qui déclare un tableau de 50 objets de classe « Operation » et affiche le menu suivant :

```
0. Quitter
1. Voir les opérations
2. Ajouter une opération
```

Tant que l'utilisateur ne choisit pas l'option 0, le menu lui est proposé de nouveau.

d) Codez l'option 1. Que se passe t'il ?
Remarque : Vous devriez voir apparaître 50 fois null. Toutes les cases ont été initialisées mais à null. Modifiez alors la condition d'arrêt de votre boucle afin que la boucle s'arrête dès lors qu'elle rencontre une case à null.

e). Codez l'option 2. Demandez à l'utilisateur les informations nécessaires pour créer un objet « Operation » puis pensez à stocker cet objet au sein de votre tableau. Testez l'option 1. Que se passe t'il ? Vous devriez voir l'adresse mémoire de cet objet et non ses valeurs, il faut afficher non pas la case contenant l'objet mais le résultat d'une méthode appliquée à la case (donc à l'objet).

f) Développez la méthode « toString » dans la classe « Operation » et modifiez l'option 2

2. Après avoir créé votre classe « Operation », développez un programme Java qui affiche le menu suivant :

0. Quitter
1. Voir les opérations
2. Ajouter une opération

Tant que l'utilisateur ne choisit pas l'option 0, le menu lui est proposé de nouveau.

3. Développez les différentes options du menu en mémorisant les opérations dans une collection (ArrayList) d'opérations.