TD/TP1

REVISIONS – UTILISATION DES OUTILS ECLIPSE, JUNIT, JAVADOC

1. CLASSE MONEY

L'objectif de cette classe est de pouvoir considérer des montants de différentes devises. Le montant peut être négatif et la devise s'exprime comme une chaîne de 3 caractères (EUR : euro, USD : dollar, CHF : franc suisse, GBP : livre sterling, ...).

1.1. Copier la **classe Money** ci-dessous dans Eclipse dans un package nommé ***tp1*.**

```
package tp1;

public class Money {
	private int montant;
	private String devise;

	public Money(int mon, String dev) {
		this.montant = mon;
		this.devise = dev;
	}

	public int getMontant() {
		return this.montant;
	}

	public String getDevise() {
		return this.devise;
	}

	public Money add(Money m) {
		return new Money(this.getMontant()+m.getMontant(), this.getDevise());
	}
}
```

NB : Sous Eclipse, *CTRL A* suivi de *CTRL I* permet d'indenter correctement le code.

1.2. Ajouter une méthode ***equals*** sachant que deux instances de la classe Money sont égales si elles ont même montant et même devise. La signature de cette méthode devra être :

```
public boolean equals(Object ob)
```

1.3. Ajouter une méthode ***toString*** de signature `public String toString()`

1.4. Commenter cette classe afin de pouvoir générer une documentation avec la commande ***javadoc***.

1.5. Créer une classe de tests dans un package ***test*** avec JUnit pour valider les méthodes ***equals*** et ***add*** (bouton droit sur la classe Money, choisir *New* puis *JUnit Test Case*). Les initialisations peuvent être communes pour ces 2 méthodes, les placer après **@Before** dans une méthode d'initialisation.

----------**Faire valider la question 1.5. par votre enseignant** ----------

1.6. Vous avez du vous en rendre compte à la question précédente, le code de la méthode ***add*** n'est pas vraiment correct, que se passe-t-il si les deux montants ne sont pas de même devise ?

Créer une **classe DeviseException**, héritant de la classe **Exception** et modifier ensuite la méthode ***add*** pour qu'une instance de *DeviseException* soit lancée dans la méthode *add* si les devises des montants à additionner sont différentes.

1.7. Ecrire une méthode de test qui construit deux objets de la classe Money avec des devises distinctes et vérifier que, dans ce cas, une exception *DeviseException* est levée.

2. CLASSE MONEYLIST

Afin d'organiser des sommes d'argent de différentes devises, nous allons créer une classe **MoneyList** en utilisant une **ArrayList**.

2.1. Compléter la classe **MoneyList** ci-dessous. La méthode *public void ajouterSomme (Money m)* doit parcourir la liste *list*, si un élément possède la même devise que m, alors le montant de m est ajouté à cet élément, sinon, l'élément est ajouté à *list*.

```
public class MoneyList {

	private List<Money> list;

	public MoneyList() {
		list= new ArrayList<Money>();
	}

	public List<Money> getList() {
		return list;
	}

	public void ajouterSomme (Money m) throws DeviseException {
	// A compléter

	}
}
```

2.2. Compléter la classe **MoneyList** avec

- une méthode `public String toString()` afin d'afficher le contenu de list,
- une méthode `public boolean equals(Object obj)` qui retourne true si l'instance courante et celle passée en paramètre contiennent les mêmes devises avec les mêmes montants.

2.3. Ecrire une classe de test vérifiant la bonne cohérence de la méthode *`ajouterSomme`*.

2.4. Ajouter (utiliser la classe Collections) une méthode ***triMontant*** permettant de trier les éléments de list selon la valeur du montant et une méthode ***triDevise*** qui trie les éléments de list selon l'ordre alphabétique des devises. Compléter la classe de test pour valider ces deux méthodes.