UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Licence 1 INFO
INFO0201

TP n°9

Classe Object, clonage et introspection

En plus du CM, pour comprendre et apprendre à vous servir de la classe `Object`, une fiche **"La classe Object et ses méthodes"** est mise à disposition dans la section CM sur *Moodle*. Pour approfondir vos connaissances sur la classe `Object`, n'hésitez pas à consulter la page officielle : API Java 8 (`http://docs.oracle.com/javase/8/docs/api/`).

Essayez de répondre aux différentes questions, si vous n'êtes pas sûr de votre réponse n'hésitez pas à questionner votre intervenant de TP.

1 Personnes, étudiants et enseignants (encore ?)

Pour cet exercice, nous utiliserons les classes du package `personne` à télécharger sur *Moodle*. Placez ces fichiers dans votre répertoire `packages` (contenant tous les packages développés depuis le début d'Info0201), puis compilez-le.

Nous supposons maintenant la classe de test `TestPersonne1.java` disponible sur *Moodle*. Elle contient trois méthodes permettant de créer depuis le clavier, une personne, un étudiant et un enseignant. Dans la méthode principale, un tableau est créé et initialisé. Placez ce fichier dans le répertoire du `TP9`, compilez-le puis exécutez le programme.

1. Comment expliquez-vous l'affichage obtenu dans le résumé ?

Nous souhaitons que les affichages soient les suivants :

- classe `Personne` : par exemple "Smith John"
- classe `Etudiant` : par exemple "Etudiant Smith John, numero 12345"
- classe `Enseignant` : par exemple "Enseignant Smith John, salaire 1234,56 euros"

2. Modifiez les classes du package pour obtenir l'affichage attendu.
3. Est-on obligé de recompiler la classe de test lors de la modification des classes du package ?

Nous souhaitons maintenant modifier le `main` et réaliser une copie profonde du tableau nommé `tableau`.

4. Pour cela, ajoutez un constructeur par copie dans la classe `Personne` puis utilisez-le dans le `main` pour créer un autre tableau et réaliser la copie profonde du premier tableau. Affichez le contenu du nouveau tableau.
5. Expliquez l'affichage.
6. Ajoutez les constructeurs par copie dans les classes `Etudiant` et `Enseignant`.
7. À l'aide du mot-clef `instanceof`, modifiez la copie profonde de votre `main` pour qu'elle copie intégralement les éléments du premier tableau et que l'affichage soit identique à celui de la question 2.

2 L'attaque des clones

Nous souhaitons ajouter la classe `Vacataire` dans notre package `personne`. Un vacataire est un enseignant qui appartient à une entreprise (une chaîne de caractères). Dans l'affichage, son entreprise doit être affichée.

1. Ajoutez cette classe puis modifiez le `main` pour pouvoir créer des vacataires dans nos tableaux (vous pouvez ajouter une méthode `creerVacataire`). Affichez ensuite le tableau résultant et vérifiez que les vacataires sont affichés correctement.

Pour chaque classe ajoutée, la classe de test devra être modifiée et un nouveau test ajouté avec un `instanceof`. Pour éviter cela, nous proposons d'ajouter la méthode `clone` dans chaque classe. Pour rappel, la signature de la méthode est la suivante (lors de la redéfinition, n'oubliez pas de déclarer cette méthode `public` au lieu de `protected`) :

```
protected Object clone()
```

2. Ajoutez cette méthode dans les quatre classes du package.
3. Modifiez le `main` pour utiliser `clone` à la place des `instanceof` et des appels aux constructeurs par copie. Vérifiez ensuite que l'affichage du tableau soit identique à celui de la question 1.
4. Si une nouvelle classe fille d'une des classes du package est ajoutée, quelles modifications devront être apportées au `main` ?
5. Que devra contenir une telle classe fille ?

3 L'égalité entre personnes

Nous utilisons maintenant la classe de test `TestPersonne2.java`.

1. Copiez-la dans le répertoire du `TP9`, compilez-la puis exécutez-la.
2. Comment expliquez-vous l'affichage obtenu ?

La méthode `equals` est proposée dans la classe `Object`. Sa signature est la suivante :

```
public boolean equals(Object obj)
```

3. Modifiez le `main` et, à la place de l'égalité `(p1 == p2)`, utilisez la méthode `equals` pour comparer les deux objets.
4. Qu'est-ce que cela change ? Pourquoi ?
5. Redéfinissez la méthode `equals` dans la classe `Personne`, puis vérifiez son bon fonctionnement en exécutant la classe de test.
6. Si nous modifions `p2` en un `Etudiant`, la comparaison est-elle possible ? Expliquez.
7. Ajoutez la redéfinition de la méthode equals dans les trois autres classes.
8. Dans la classe de test, testez les différents cas :
   - Egalité entre deux personnes
   - Egalité entre deux étudiants
   - Egalité entre un étudiant et une personne
   - Egalité entre un étudiant et un enseignant

4 La classe `Object`

1. Consultez l'API Java en ligne et cherchez le contenu de la classe `Object`.
2. La méthode `getClass` permet de récupérer des informations concernant la classe d'un objet. Comment afficher le nom de la classe correspondant à un objet ?
3. Comment afficher les noms des méthodes d'un objet ?

5 Retour sur l'égalité entre personnes

Avez-vous bien respecté la symétrie pour vos redéfinitions de la méthode `equals` ?

Pour rappel, la méthode `equals` doit être symétrique , c'est-à-dire :
Si `x.equals(y)` est vrai
Alors `y.equals(x)` doit être vrai
Vous pouvez relire la fiche sur la classe `Object`.

Afin de vérifier, ajoutez le code suivant à votre classe de test et vérifiez que vous obtenez le résultat attendu, sinon modifiez le code des méthodes `equals` afin de l'obtenir.

```
Personne personne = new Personne("Doe", "John");
Etudiant etudiant = new Etudiant("Doe", "John", 2000);
Enseignant enseignant = new Enseignant("Doe", "John", 2000.0);
if(personne.equals(etudiant) == etudiant.equals(personne))
   System.out.println("Top !");
else
   System.out.println("Il faut revoir la symetrie.");
if(personne.equals(enseignant) == enseignant.equals(personne))
   System.out.println("Top !");
else
   System.out.println("Il faut revoir la symetrie.");
```