TP, troisième série d'exercices

Benoît Valiron
benoit.valiron@lipn.univ-paris13.fr

29 novembre 2010

Introduction

Encore une fois, vous êtes sensés me rendre ce que vous avez réalisé.
Dans tous les exercices, vous placerez les portées comme on l'a vu en cours.

Exercice 1.

Vous trouverez la classe `Personne` dans le dossier
`/home/usager/TPGTR/secret/Personne.java`

1. Créez une sous-classe `Etudiant` à la classe `Personne`. Cette sous-classe comporte un champs `numEtudiant` de type `long`. Le constructeur attend les mêmes arguments que celui de la classe `Personne`, plus un, pour le numéro d'étudiant.
2. Sur le même principe, créez une classe `Prof`, sous-classe de `Personne`. Elle possède un champ `bureau` de type `String`. Le constructeur doit aussi initialiser ce champ.
3. Redéfinissez la méthode `toString` de `Personne` pour prendre en compte les nouveaux champs.
4. À la place des commentaires, dans la classe suivante :

   ```
   public class Test {
     public static void main(String[] args) {
       // DU CODE
       // DU CODE
     }
   }
   ```

   créez deux nouvelles variables de type `Personne` : un nouvel étudiant `Bob` et un nouveau prof `Ben`.
5. Ajoutez une méthode abstraite `String quelRole()` à la classe `Personne`. Mettez à jour les classes `Etudiant` et `Prof` pour que leurs méthodes `quelRole()` rende respectivement les valeurs `"étudiant"` et `"enseignant"`.
6. Dans la méthode `main` de la classe `Test`, pouvez-vous créer un nouvel objet de type `Personne` ? Pourquoi ?

Exercice 2.

La documentation java est en ligne : `http://download.oracle.com/javase/6/docs/api/`

1. Comment feriez-vous pour que la classe `Prof` implémente l'interface `Runnable` ? Faites-le.
2. Dans la classe `Test`, créez une variable `r` de type `Runnable` et placez un nouvel objet de type `Prof` dedans. Pouvez-vous accèder au champ `bureau` de la variable `r` ? Pourquoi ?

Exercice 3.

1. Comment feriez-vous pour que la classe `Personne` implémente l'interface `CharSequence` ? Vous prendrez comme chaine de caractère sur laquelle travailler la valeur de la méthode `quelRole()`.
   Note : La classe `String` implémente-elle l'interface `CharSequence` ? Si tel est le cas, vous pouvez directement réutiliser les méthodes qui vous intéresse.
   N'oubliez pas de prendre en compte les exceptions.
2. Utilisez les nouvelles méthodes sur un objet de type `Etudiant`.

Exercice 4 (bonus).

1. Créez la classe abstraite `GenPointCouleur` :

   ```
   public abstract class GenPointCouleur {
     private String couleur;
     GenPointCouleur(String s){ couleur = s; }
     public void changeCouleur(String s) { couleur = s; }
     public abstract void translate(double x, double y);
     protected String toString() { return(couleur); }
   }
   ```

   et une classe `PointCouleur` sur le modèle de la première série d'exercices qui étends la classe abstraite.
   Modifiez les portées de la classe abstraite pour que ça compile.
2. Créez une classe `PointCouleurND` qui implémente un point en dimension quelconque : la classe possède un champ `double[] coords` qui contient les coordonnées du point. Le constructeur attends un tableau de `double`. La méthode `translate` modifie uniquement les deux premières coordonnées du point.
3. Créez une exception `DimensionException`.
4. Modifiez les classes de l'exercice pour que la méthode `translate` puisse lancer une exception `DimensionException`.
5. Modifiez la méthode `PointCouleurND.translate` pour lancer l'exception si la dimension du point est inférieur à 2.
6. Créez une classe `TestPoint` avec une classe `main` qui instancie un objet `p` de type `GenPointCouleur` et place dedans un point de dimension 1. Utilisez la méthode `translate` sur `p`, et attrapez l'exception lancée.