Java avancé - TP 2 - Révisions

Romain Péchoux

8 février 2010

Problème : le parc automobile

Une voiture possède un numéro d'immatriculation, une marque, un âge, un âge maximum au delà duquel elle part à la casse. Elle peut être en panne ou non. On doit pouvoir :

- afficher les caractéristiques d'une voiture,
- faire vieillir une voiture (par défaut, d'un an),
- changer son statut (elle devient épave et part à la casse),
- faire vrombir une voiture,
- connaître à tout moment le nombre de voitures en circulation (celles qui ne sont pas à la casse).

Par ailleurs, on ne peut pas créer directement une voiture. On peut seulement créer une voiture appartenant à une famille donnée (son âge est alors initialisé à 0). Dans ce parc automobile, il y a deux familles :

1. Les citadines :
   - l'âge maximum d'une citadine est de 15 ans,
   - le moteur d'une citadine fait «boum».
2. Les monospaces :
   - caractérisés par une capacité maximale (de places)
   - l'âge maximum d'un monospace est de 20 ans,
   - le moteur d'un monospace fait «vroum vroum».

Ecrire le programme modélisant le parc automobile. Il devra :

- implémenter l'interface suivante :

```
/** permet de définir des types d'éléments
* pouvant tester leur appartenance à une marque donnée
*/
interface Copyright{ public boolean estDeLaMarque(Marque o); }
```

- redéfinir `equals` de sorte à ce que deux voitures de même marque et même âge soient égales
- inclure une classe de test où vous définirez un tableau de Voitures incluant des citadines et des monospaces. Pour chaque voiture du tableau, faire vrombir son moteur. Quelles sont les fonctionnalités de Java utilisées à cet effet ? La redéfinition est-elle statique ou dynamique ?

- dans la classe test, créer une méthode statique qui affiche “voiture” si l'objet passé en argument est un objet de type voiture. Surcharger cette métode afin qu'elle affiche “mono” si l'objet passé en argument est de type monospace. Tester cette méthode sur les éléments du tableau créé à la question précédente. Que constater ? La surcharge est-elle statique ou dynamique ?
- créer une classe imbriquée locale Mini dans votre fichier Test qui hérite de Citadine et qui possède un attribut pour représenter le nombre de portes. Tester son utilisation. Créer un objet d'une classe anonyme de telle sorte qu'il hérite de Mono et que son moteur fasse “bling bling”.
- définir des accesseurs si nécessaire
- ajouter des levées/captures d'exception à votre programme (en créant de nouvelles exceptions si nécessaire). Par exemple, si une voiture ne peut plus vieillir.
- Créer une archive `jar` en utilisant la commande `jar cvfm Fichier.jar FichierManifest Fichiers.class` (un fichier Manifest permet de spécifier l'exécutable de l'archive en ajoutant la ligne suivante `Main-Class: Test`). Exécuter l'archive avec la commande `java -jar fichier.jar`