TP3 - Les collections en Java

Septembre 2005

Objectif

L'objectif ce TP est de présenter d'une part l'utilisation des collections et d'autre part l'utilisation du plugin omondo (`http://www.omondo.com`) pour eclipse.

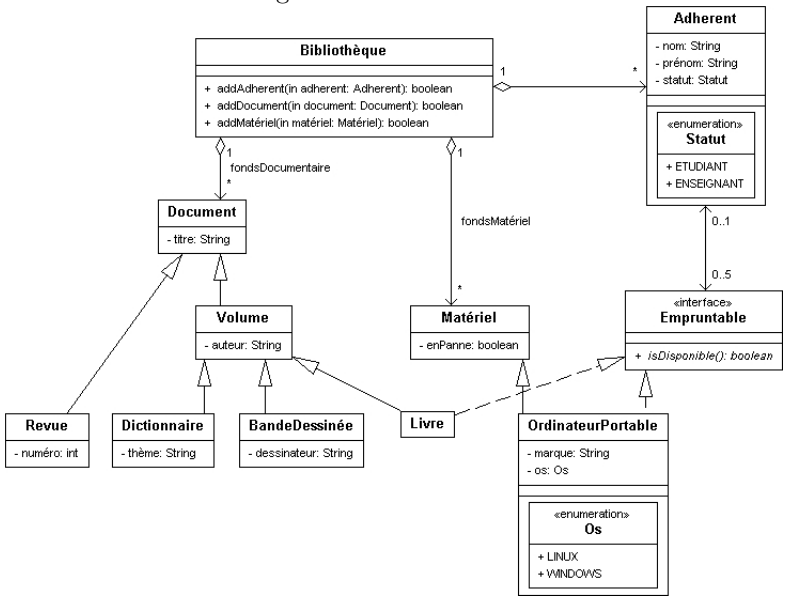
1 Utilisation "simple" des collections

Vous utiliserez les classes du TP précédent.

- Créez un **ensemble** d'animaux (sans utiliser de génériques) et peuplez le (physiquement un `HashSet`).
- On suppose que tous les animaux ont un nom, indiquez que deux animaux sont égaux s'ils ont le même nom.
- Vérifier que l'on ne puisse pas ajouter deux animaux ayant le même nom (par définition d'un ensemble)
- Affichez cet itérateur en utilisant un itérateur (puis avec le foreach)
- Afficher la liste des animaux trier par l'ordre alphabétique de leur nom.
- Transformez le support physique de cet ensemble en un `TreeSet`, que faut-il modifier d'autre ?
- Convertissez cet ensemble en un tableau d'animaux (attention au type)
- Modifier cet ensemble en ajoutant la généricité et définissez un ensemble d'animaux (supprimez les transtypages devenus inutiles).
- On suppose que l'on associe à chaque animal un tatouage unique composé de sa race et d'un nombre par exemple *aigle1*, *aigle2*, *chien1* et *chien2*. Créer une `Map` (avec la généricité) contenant des animaux et affichez les informations concernant l'animal dont l'identifiant est *chien1*.

2 Ecriture manuelle d'un programme à partir d'un diagramme de classe

L'objectif est d'écrire un programme de gestion d'un bibliothèque qui prête des livres et des ordinateurs à des enseignants et des étudiants.



2.1 Architecture de base

Ecrire (sans utiliser l'éditeur UML) les classes Java correspondant au diagramme UML donné décrivant une bibliothèque simple. Vous veillerez en particulier à

- redéfinir la méthode `toString()` des différentes classes pour que celles-ci puissent être affichées

- représenter correctement les associations multiples en utilisant des collections (il est conseillé d'utiliser les génériques).
- créer une classe `Test` dans laquelle vous pourrez :
  - créer une bibliothèque
  - ajouter des adhérents qui doivent avoir un nom, un prénom et un statut
  - ajouter au moins deux livres, deux revues, deux dictionnaires et deux BD
  - ajouter deux ordinateurs portables (un sous linux, un sous windows)

2.2 Gestion des emprunts

Ecrire le code permettant de gérer les emprunts en ajoutant aux adhérents les méthodes `boolean emprunter(Empruntable e)` et `boolean rendre(Empruntable e)`. Pensez à bien vérifier qu'un objet n'est pas emprunté deux fois (vérifier que cela marche en faisant un test dans la classe `Test`) et que l'on n'emprunte pas plus de cinq objets.

- Ajouter une méthode `void afficherEmprunts()` à la classe `Adherent`
- Ajouter une méthode `void afficherFonds()` à la classe `Bibliothèque` qui affiche les documents et le matériel (chacun affichant par qui il a été emprunté). (**Ajoutez les méthodes qui vous semble utiles pour cela**)
- Surcharger les méthodes `emprunter` et `rendre` pour les Livres et les Portables, pour vérifier que l'on emprunte au plus cinq livres mais un seul portable.

2.3 Recherche dans une collection

Ajouter une méthode `Collection rechercherTitre(String mot)`, qui retourne une collection contenant des références vers les documents qui comportent la chaîne `mot` dans leur titre.

3 Ecriture assistée d'un programme à partir d'un diagramme de classe

En vous inspirant de l'exercice précédent et du diagramme de classe représentant des vols d'avions, mettez en place une application de création et d'affichage des vols et des équipages.

Vous utiliserez le plugin omondo de eclipse, pour créer le diagramme UML, puis vous adapterez le squelette d'application obtenu.

4 Si vous avez fini...

Vous pouvez affinez vos connaissances en suivant le tutorail de Sun : `http://java.sun.com/docs/books/tutorial/collections/index.html`.