TD4 : Composite, Decorator, Flyweight, Proxy

Exercice 1 – UML...

Le but de cet exercice est de modéliser un système d'achat en ligne de type Amazon réalisé avec J2EE.

Tout d'abord, télécharger le fichier tdamazon.zip (http://igm.univ-mlv.fr/ens/Master/M1/2011-2012/POO/tdamazon.zip) . et intégrer le à votre projet courant

Nous utiliserons l'éditeur UML Violet (très simple et autonome) ou ...

1. Faire le diagramme de classe avec simplement les classes, leurs attributs et méthodes.
2. Maintenant, vous allez modifier le diagramme. Faire apparaître les relations d'héritage sur le diagramme: CD, DVD & Book implémentent l'interface Item.
3. La prochaine étape va être de créer des associations. Modéliser le fait qu'un Kart ne puisse avoir qu'un client, qui ne doit pas forcément être renseigné au départ.
   Regarder maintenant les apports à la classe Kart. Faites apparaître le champs Customer sur le diagramme Kart.
4. Ajouter les associations suivantes avec les cardinalités appropriées en fonction des indications ci dessous :
   - Un Kart peut avoir plusieurs Item. Ajouter une association en utilisant une ArrayList.
   - Ajouter une classe abstraite Product commune à CD, DVD et Book.
   - Un Product doit avoir un Editor. Un Editor peut avoir plusieurs Product.
   - Un Book doit avoir un ou plusieurs Author. Un Author peut avoir écrit plusieurs Book.
   - Un Book peut avoir une BookCover
   - Un Dvd possède plusieurs Language's.

Exercice 2 - Composite et Decorator

Vous regrettez l'époque où vous aviez le temps de jouer à Command & Conquer ? Voici l'occasion de créer votre version du jeu ! Commencer par télécharger le fichier td1war.zip (http://igm.univ-mlv.fr/ens/Master/M1/2007-2008/GenieLog/td1war.zip) Importer-le dans le répertoire source de votre projet courant dans Eclipse.

Le but de cet exercice est de modéliser des unités militaires. Certaines unités (AircraftCarrier, ArmyTruck, Helicopter et Destroyer) peuvent jouer un rôle de transport de troupe et contenir d'autres unités.

Dans le paquetage fr.umlv.gl.td2.war se trouvent toutes sortes d'unités militaires sous forme de classes.

1. Organiser les classes pour que les unités effectuant du transport de troupe en utilisant le design pattern composite. Les transporteurs devront donc possèder des méthodes permettant d'ajouter ou de retirer d'autres unités et de visualiser la liste des unités.
2. Modifier votre implantation pour qu'il soit impossible de transporter une unité étant elle même un transporteur.
3. Faites en sortes que la puissance de feu, la vie et la vitesse d'un transporteur soit la somme des caractéristiques des unités de même armes qu'il transporte. Par exemple si un destroyer transporte un soldat et deux marines, ses caractéristique ne dépendront que des deux marines.
4. Nous souhaitons maintenant ajouter un système de bonus qui permet de booster les fonctionnalités de feu, de vie et de vitesse des soldats.

Sachant que l'on souhaite avoir des bonus qui ajoute de la puissance de feu, de la vite ou augmente la vitesse ainsi que des bonus multipliant par un facteur l'ensemble des caractéristiques d'un soldat. Proposer une solution d'implantation en utilisant le design pattern decorator.

Enfin, faites en sorte que l'on puisse aussi appliquer le système de bonus aux transporteurs. A vos claviers !

Exercice 3 - Flyweight

Le centre ville va être refait. Le logiciel de conception doit pouvoir représenter les rues et les différents pavés qui vont être posés. 3 types de pavés vont être posés :

- le standard, décrit par ses dimensions, sa couleur, l'endroit où il va être posé (supposons une position terrestre donné par une paire de Double), son poids, etc.
- les pavés de bordures, un peu plus jolis, qui seront mis en bordure des routes. Leur description est donc quasi identique à celle des pavés standards.
- les pavés fantaisies, avec des initiales ou des dessins et qui seront disséminés dans la ville. Les pavés fantaisie sont décrits comme les pavés standards avec, en plus, une indication d'initiale ou un numéro de dessin.

Question : rappeler le principe du design pattern Flyweight

Implémentation : on souhaite implémenter, en utilisant le pattern Flyweight, le cœur de ce logiciel de conception.

Dans notre conception, un Pave a ses différentes caractéristiques (tel que décrit pour les 3 types de pavés) et a aussi une position terrestre (l'endroit où il sera placé).

Avant d'implémenter le DP Flyweight, l'interface de base était :

```
interface Pavement {
      // inclut les dimensions, la position
      //et les éventuelles caractéristiques particulière (dessin, ...)
      String getDescription() ;
}
```

- La position des pavés sera traitée comme la *donnée extrinsèque* de nos objets Pave
  - modifiez l'interface en conséquence
- Implémentez les 3 classes de Pavés
- Implémentez la Factory qui gérera les Pavés standards comme des Flyweights (partagés)
- Implémentez la classe qui gérera les données extrinsèques
- testez ...

Coder votre solution et un petit programme de tests.