Sujet 7 : DP proxy

Exercice 1 Proxy de protection

Un bar est spécialisé dans la dégustation de vodka. Lorsqu'un client commande une boisson, le barman doit vérifier que le client est majeur avant de le servir. S'il n'est pas majeur, le barman doit refuser de le servir.

1. Identifiez les classes dont vous avez besoin et adaptez le diagramme de classe général fourni dans le cours à ce cas. À quoi sert le proxy ici ?

2. Codez l'interface et les classes correspondant dans le cours à *SujetProxy* et *SujetRéel*.

Les clients du bar sont rangés dans un fichier dont le nom sera passé en paramètre du constructeur du bar. Le fichier contient les différents clients à servir : chaque ligne correspond à un client et a la forme *identifiant nom âge* comme ci-dessous.

```
1 Dominique 19
2 Claude 17
3 Camille 25
```

3. Créez une classe *Client* pour gérer les clients.

4. Codez ensuite une classe *Bar* avec les clients lus dans le fichier. Les clients seront entrés dans une table de hachage (*HashMap* ; vous regarderez la documentation). Pour lire les clients dans le fichier, vous utiliserez la classe *Scanner* (voir la documentation). Cette classe doit implanter les méthodes *void veutBoire(Client client)* qui fait boire le client passé en paramètre et la méthode *void boireTous()* qui fait boire tous les clients du bar.

5. Créez un fichier client puis une classe de test qui fait boire tous les clients de votre fichier et testez votre code.

Exercice 2 Proxy virtuel

On souhaite écrire une petite interface graphique qui télécharge et affiche des pochettes d'album. L'utilisateur va choisir la pochette à afficher dans un petit menu. Le problème est que le téléchargement des pochettes peut être long et on souhaite pendant ce temps afficher le message « en cours de chargement ». Si l'illustration a déjà été chargée, on souhaite ne pas la télécharger à nouveau.

1. L'interface que doit implémenter le proxy est ici l'interface *Icon*. Regardez sa documentation. Quelles sont les méthodes à implémenter ?

2. Notre classe réelle existe déjà, il s'agit de la classe *ImageIcon*. Regardez sa documentation.

Nous allons donc introduire une nouvelle classe *ProxyImage* qui va réduire l'utilisation du téléchargement. Cette classe doit implémenter les méthodes citées plus haut.

3. Récupérez les deux classes fournies : la classe de test et la classe *ComposantDImage*. À quoi sert cette dernière classe ? Regardez la documentation de *JComponent*.

4. Ajoutez la classe *ProxyImage* et implémentez les méthodes nécessaires. Le code pour télécharger une image disponible à une URL donnée est disponible en page suivante.

5. Testez votre programme grace à la classe de test fournie.

```
threadChargement = new Thread(new Runnable() {
 public void run() {
   HttpURLConnection connexion;
   int nbLu;
   try {
     if (proxy != null)
       connexion=(HttpURLConnection) urlImage.openConnection(proxy);
     else
       connexion=(HttpURLConnection) urlImage.openConnection();
       byte[] bytes=new byte[connexion.getContentLength()];
       int  totalLu=0;
       while (totalLu < bytes.length) {
         nbLu=connexion.getInputStream().read(bytes,totalLu,bytes.length-totalLu);
         totalLu+=nbLu;
       }
       image = new ImageIcon(bytes, "Pochettes de CD");
       c.repaint();
       chargement = false;
   } catch (Exception e) {
     e.printStackTrace();
   }
 }
});
threadChargement.start();
```