CNAM

M-NFA032 Algorithmique et programmation : bibliothèques et patterns

Quatrième partie

Travail à effectuer

Rappels

Quelques éléments importants avant de commencer le fil rouge :

- à terme, votre programme comportera une interface graphique ; cette interface ne vous permettra pas de lire au clavier ni d'écrire à l'écran ; donc, ne pas mettre d'ordre de lecture ou d'écriture dans vos classes ; si vous souhaitez valider vos classes, prévoyez alors une petite bibliothèque temporaire pour les entrées-sorties.
- Dans vos classes, mettez systématiquement vos champs private ou, à la limite, protected ; vous pourrez ensuite donner des accès à vos variables d'instances via des getters et des setters.

Exercice 1 : Ligne de Panier

Nous allons nous préparer à définir un Panier. Un panier est un ensemble de paires (produit,quantité) que nous appellerons lignes de panier. En regardant un peu plus loin, un bon de commande sera constitué aussi de telles paires . Cependant, le prix des produits au catalogue pouvant changer et afin que le total d'un bon de commande soit fixe, il serait bon d'ajouter une troisième information : le prix du produit au moment de l'achat. Les deux notions étant assez semblables, mais une ligne de commande ne pourra pas être modifiée une fois créée. Donc, nous allons créer une classe pour une ligne de panier et nous créerons une classe ligne de commande sans dériver les lignes de panier.

En regardant les opérations que l'utilisateur peut faire sur son panier, nous en déduirons les services qu'une ligne de panier doit fournir :

- créer un panier
  - pas d'implication sur les lignes de panier
- ajouter une quantité d'un produit au panier
  - si le panier contient déjà une ligne de panier pour ce produit, il faut ajouter la quantité à la ligne de panier
  - sinon, il faut créer la ligne de panier et l'ajouter à la collection
- retirer une quantité d'un produit au panier
  - si le produit n'est pas présent dans le panier, il faut lever une exception spécifique
  - pour la ligne de panier concernant ce produit, retrancher la quantité à retirer. Si le résultat est inférieur ou égal à 0 (exception spécifique définie dans l'exercice précédent levée), il faudra enlever la ligne de panier du panier
- vider le panier
  - supprimer toutes les lignes de panier. Pas d'incidence sur les lignes de panier
- commander le panier
  - pour chaque ligne de panier, il faudra créer une ligne de commande similaire.

En résumé, les fonctions attendues pour une ligne de panier sont :

- créer une ligne de panier avec un produit et une quantité
- récupérer le produit
- récupérer la quantité
- ajouter un certain nombre de produits à la ligne
- retirer un certain nombre de produits à la ligne
  - signaler quand la quantité devient <= 0
- créer la ligne de commande équivalente.

1. Définir la classe LignePanier avec son constructeur et ses méthodes dans le package magasin. On laissera la création de la ligne de commande correspondante en attente. Notez que le produit ne changera jamais pour une ligne de commande !
2. Ajoutez une méthode calculant le prix total de la ligne de panier
3. Pensez à définir equals et hashCode. Deux lignes de panier seront égales si elles concernent le même produit que les quantités soient différentes ou non.

Exercice 2 : Panier

1. Définir un panier comme un ensemble de paires (produit, ligne de panier), produit étant la clé. Vous pourrez utiliser LinkedHashMap pour garder le contenu dans l'ordre où il a été enregistré.
2. Construisez l'ensemble des opérations décrit en introduction dans l'exercice précédent.
3. Comme pour les lignes de panier, laissez en attente la méthode qui permet de passer commande du panier.
4. Ajoutez une méthode retournant le montant total du panier.

Exercice 3 : Client

1. Toujours dans magasin, créez un client en vous reportant au sujet. L'adresse électronique, le genre, le nom et le prénom ne pourront pas être modifiés, par contre, les autres attributs le pourront.
2. Ajouter un panier au client avec la possibilité, par simple délégation, de le manipuler.
   1. Dans le menu Source, choisissez Generate Delegate Methods
   2. Dans la nouvelle fenêtre
      1. replier adresse (clic sur le symbole v à gauche)
      2. déplier panier
      3. choisir toutes les méthodes sauf equals, hashcode et toString.
      4. Cliquez sur ok

   Ainsi, tout ce qui concerne la manipulation du panier sera accessible via l'instance de Client mais, in fine, ce sera le panier qui gérera.
3. Ajouter une procédure de connexion qui prendra en paramètre un mot de passe, vérifiera que le mot de passe correspond à celui qui est enregistré et, en cas de succès, initialisera le panier du client à un nouveau panier. Si le mot de passe n'est pas valide, il faudra lever une exception spécifique.
4. ajouter une fonction de déconnexion qui mettra le champ panier à null.
5. Laisser en attente le carnet de commandes du client.

Exercice 4 : Ligne de Commande

Attention ! Une ligne de commande sera immuable ! C'est-à-dire qu'aucune variable d'instance ne sera modifiable après la création de l'instance.

1. Définissez une ligne de commande avec ses champs, son constructeur et les getters sur les champs. Pas de setter ! Dans le constructeur, le prix de vente effectif sera obtenu depuis le produit pris en paramètre. Il sera donc pas nécessaire de prendre le prix de vente effectif en paramètre.
2. Ajoutez une méthode qui permette de calculer le montant total de la ligne de commande.
3. Dans la classe LignePanier, finalisez la méthode de création d'une ligne de commande (on parle de patron de conception « monteur »)

Exercice 5 : Commande

Le contenu de la commande sera initialement vide, puis on ajoutera les lignes de commandes une à une. Il faudra veiller à ce qu'il ne puisse pas y avoir deux lignes de commande pour le même produit (Cf construction de la classe Panier). Attention cependant, une fois ajoutée au carnet de commande, à part la validation, la commande ne pourra plus être modifiée et seulement validée ! Vous pouvez vous inspirer des lots pour implémenter cette fonctionnalité.

1. Créez une commande avec :
   - un ensemble de lignes de commandes
   - le client
   - la date (utilisez la classe Calendar définie dans le package java.util et initialisez la variable d'instance avec l'expression Calendar.newInstance () qui retourne une instance de Calendar initialisée avec la date courante.
   - un numéro de commande (vous pouvez utiliser le compteur créé précédemment)
   - un booléen initialisé à false pour pouvoir marquer la commande comme validée
2. Pensez à créer un constructeur qui ne prendra que le client en paramètre. Les autres variables d'instance seront calculées
3. Pensez aux getters pour chacune des variables d'instance sauf pour l'ensemble des lignes de commandes. Pas de setter pour verrouiller (Cf plus bas)
4. Ajouter une méthode permettant d'ajouter une ligne de commande à l'instance courante. Une exception spécifique devra être levée si la commande est verrouillée et une autre si le produit est déjà présent dans la commande
5. Ajoutez une méthode pour calculer le montant total du bon de commande
6. Ajoutez une méthode pour verrouiller la commande lorsqu'elle sera enregistrée comme cela a été fait pour les lots
7. Finalisez dans la classe Panier, la méthode prévue qui construira un bon de commande à l'image du panier. Elle prendra le client en paramètre et elle devra vider le panier !
8. Pensez à créer une délégation de cette dernière méthode dans Client. Notez bien qu'il n'est pas nécessaire de prendre le client en paramètre dans cette délégation puisque le client qui passe commande est this…