TD3 : Couplage et Cohérence

Application de e-commerce

Considérons une application classique de eCommerce (déjà présentée en TD1).

Cette application offre deux types de fonctionnalités à ses usagers :

- La consultation des produits de son catalogue
- La vente de produits

Un groupe d'étudiants n'ayant pas bien assimilé les principes de la programmation objet propose la programmation suivante.

```
package td3;

public class Produit {
	String description;
	double prix;

	public Produit(String description , double prix) {
		this.description = description;
		this.prix = prix;

	}

	public double getPrix() {
		return this.prix;
	}

	public String getDescription() {
		return this.description;
	}
}
```

```
package td3;

import java.util.ArrayList;
import java.util.List;

public class Magasin {

	List<Produit> catalogue;

	List<Produit> panier;
	List<Integer> panierQ;
	boolean panierValid;

	public Magasin() {
		catalogue = new ArrayList<Produit>();
		panier = new ArrayList<Produit>();
```

```
		panierQ = new ArrayList<Integer>();
		panierValid = false;
	}

	public void nouveauPanier() {
		if (!panierValid) {
			panier = new ArrayList<Produit>();
			panierQ = new ArrayList<Integer>();
			panierValid = false;
		}
	}

	public void ajouterProduitDansPanier(Produit p, int quant) {
		if (!panierValid) {
			if (!panier.contains(p)) {
				panier.add(p);
				panierQ.add(new Integer(quant));
			}
		}
	}

	public void modifierQuantiteProduitDansPanier(Produit p, int quant) {
		if (!panierValid) {
			if (panier.contains(p)) {
				panierQ.set(panier.indexOf(p), new Integer(quant));
			}
		}
	}

	public void supprimerProduitDansPanier(Produit p) {
		if (!panierValid) {
			if (panier.contains(p)) {
				panierQ.set(panier.indexOf(p), new Integer(0));
			}
		}
	}

	public double calculerPrixPanier() {
		double res = 0;
		for (Produit p : panier) {
			res += panierQ.get(panier.indexOf(p)) * p.getPrix();
		}
		return res;
	}

	public void validerPanier() {
		panierValid = true;
	}

	public List<Produit> getCatalogue() {
		return catalogue;
	}

	public void addProduitDansCatalogue(Produit p) {
		catalogue.add(p);
	}

	public void supprimerProduitDansCatalogue(Produit p) {
		catalogue.remove(p);
	}
}
```

```java
package td3;

public class Main {

	public static void main(String[] args) {
		//Construction des produits
		Produit livre = new Produit("The Livre", 15);
		Produit cd = new Produit("The CD", 15);

		//Construction du magasin
		Magasin mag = new Magasin();
		mag.addProduitDansCatalogue(livre);
		mag.addProduitDansCatalogue(cd);

		//exemple de commande
		mag.ajouterProduitDansPanier(livre, 1);
		mag.ajouterProduitDansPanier(cd, 1);
		mag.validerPanier();

		//
		System.out.println("Le prix du panier est de : " +
mag.calculerPrixPanier());
	}
}
```

Q1 : Modifiez cette application afin d'assurer une meilleure cohérence. En particulier, séparez les deux types de fonctionnalités dans deux classes différentes.

Q2 : Modifiez cette application afin qu'il soit possible de gérer plusieurs panier en même temps. On en profitera pour supprimer la double liste panier et panierQ en introduisant les classes nécessaires à la programmation du Panier.

Q3 : On souhaite ajouter une nouvelle fonctionnalité permettant de gérer le fichier clientèle. Cette fonctionnalité doit permettre d'ajouter une nouvelle FicheClient puis de lister tous les paniers construits et validés par un client.

Q4 : Construisez des interfaces Java permettant de définir les différentes opérations utilisables à partir d'une interface graphique. L'objectif de cette interface est de minimiser les références entre l'interface graphique et les classes de l'application.

Fonctionnalité de gestion des stocks (Optionnel)

On souhaite ajouter une nouvelle fonctionnalité de gestion des stocks à notre application de eCommerce.

Le stock permet de savoir quelles références sont actuellement en stock et ainsi réellement vendable. Ainsi, si un client ne devrait pas pouvoir valider un panier alors qu'il manque une référence de son panier dans le stock.

Q1 : Ajouter les classes nécessaires à la gestion du stock.

Q2 : Faite en sorte qu'il ne soit pas possible d'ajouter un produit dans un panier si celui-ci n'est pas disponible en stock.

Q3 : Faite en sorte qu'il soit possible d'ajouter des demandes de produit non disponible en stock dans un panier. Ainsi, dès que les produits seront disponibles en stock, alors ils seront automatiquement ajoutés dans le panier.