Université d'Orléans L2 INFORMATIQUE
SLA4IF04 – Modélisation objet

TP designs patterns

On souhaite journaliser les opérations effectuées sur les comptes bancaires d'une agence.
1 – Dans le package `modele` fourni, créer une classe `Journalisation` en utilisant le pattern singleton pour gérer la journalisation. On enregistrera deux types de log, les logs pour les opérations qui se sont déroulées sans problème et les logs pour les opérations qui n'ont pas pu s'effectuer (débit impossible).
`Journalisation` aura deux champs privés de type `String`, `log` et `logErr` pour enregistrer ces deux types de logs. Ajouter les getters.
La classe fournira deux méthodes :
`public void ajouterLog(String leLog)` qui complète le log avec le nouveau log passé en paramètre précédé de la date du jour.
`public void ajouterLogErr(String leLog)` qui complète le logErr avec le nouveau log passé en paramètre précédé de la date du jour.

2 - Creér un package `controleur` contenant une classe `Controleur` qui va intercepter les demandes de l'utilisateur et définir les actions à effectuer.
Controleur contiendra deux attributs :
```
private Console console = new Console();
private AgenceBancaire agence = new AgenceBancaire();
```
Dans le contrstructeur du `Controleur`, après avoir affiché les informations de tous les comptes, on demandera en boucle à l'utilisateur de saisir un numéro de compte, un type d'opération (C pour crédit ou D pour débit) et un montant. Le système effectuera l'opération demandée si elle est possible. Sinon on affichera un message.
On demandera à l'utilisateur s'il souhaite quitter ou effectuer une autre operation.
Lorsque l'utilisateur demande de quitter, on affiche le log des opérations.

3 - Créer un package `vue` contenant la classe `Console`.
La classe `Console` permettra d'interagir avec l'utilisateur. Elle fournira les méthodes suivantes:
```
public int demanderNumCompte()
public void afficherInformations(String s)
public String demanderOperation()
public int demanderMontant()
public String demanderQuitter()
```

4 - Créer un package `application` contenant la classe `Main`.
La classe `Main` lancera l'application en créant un objet de type `Controleur`.

4 - Tester votre application.

5 - On souhaite maintenant afficher les logs à chaque ajout d'une nouvelle information concernant une opération.
On envisage deux affichages possibles. `Console` affichera les logs sans erreurs en utilisant `System.out`.
Une autre classe `ConsoleErr` affichera les logs des erreurs en utilisant `System.err`.
Si le dernier log modifié est `log`, c'est `log` qui sera affiché, sinon c'est `logErr`.
On pourrait envisager d'autres élements qui devraient être tenus au courant des modifications des logs.

Proposer une solution évolutive pour prendre en compte le fait qu'on doit pouvoir facilement ajouter d'autres éléments devant se mettre à jour en fonction de la modification de l'état des logs.