CNAM

M-NFA032 Algorithmique et programmation : bibliothèques et patterns

Septième 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.

Thème de la semaine

Cette semaine, nous allons commencer l'interface graphique de l'application par la fenêtre d'identification.

Exercice 1 : Fenêtre principale

La fenêtre principale, celle qui nous permettra d'interagir avec l'application, sera développée plus tard. Pour l'instant, nous allons nous contenter de la définir.

1. Créer un package ui.main.
2. Créez dans ce package « MainFrame » classe héritière de JFrame.
3. Créer son constructeur en
   - donnant un titre à la fenêtre,
   - pensant à arrêter le programme à la fermeture de cette fenêtre,
   - la dimensionnant au mieux,
   - la positionnant au centre.

Exercice 2 : Administrateur

Nous allons créer une classe Administrateur, héritière de Client. Pour l'instant, les deux classes auront exactement le même comportement. Elles se distingueront un peu plus tard.

1. Créer une classe nommée «Administrateur» héritière de Client.
   - Y Créer uniquement un constructeur se contentant d'appeler le constructeur de Client.
   - Penser à ajouter le serialVersionUID.

Exercice 3 : Classe principale

Cette classe servira de point d'entrée du programme. Nous allons la définir partiellement. Elle sera complétée par la suite.

Création de la classe

1. Créer la classe « Systeme » dans le package application.

Variables de classe

1. Définir une constante statique et publique « FILE_NAME » et l'initialiser avec un nom de fichier que vous choisirez selon votre bon gré. Les sauvegardes et restitutions des données se feront sur ce fichier.
2. Ajouter une variable privée et statique « lf » de type « LoginFrame ». Cette classe sera définie par la suite. Cette variable nous servira à mémoriser la fenêtre d'identification tant qu'aucune personne ne se sera connectée.
3. Ajouter une variable privée et statique « mf » de type « MainFrame ». Cette variable nous permettra de retrouver facilement la fenêtre principale de l'application quand elle aura été créée.
4. Ajouter enfin une variable privée et statique « utilisateur » de type « Client ». Cette variable nous permettra de retrouver la personne connectée (client ou administrateur).

Initialisation des structures de données

1. Définir une méthode « initSysteme » privée, statique, sans résultat et sans paramètre, dont le rôle va consister à initialiser les structures de données lors du premier lancement de l'application. Elle devra
   - appeler la méthode getInstance sur les 4 classes appliquant le patron singleton et nécessitant une sauvegarde : Compteur, Catalogue, CarnetCommandeMagasin et CarnetClientele ; il ne sera pas nécessaire de mémoriser l'instance de chacune des classes ;
   - créer un administrateur, « root » par exemple, et l'enregistrer dans le carnet de clientèle ;
   - sauver la structure de données par l'appel à
     ```
     GestionnaireSauvegarde.getInstance().sauver(FILE_NAME);
     ```

Getter et setter pour utilisateur

1. Prévoir un getter et un setter statiques pour la variable « utilisateur ».
2. Dans le setter,
   - si un utilisateur est déjà connecté (variable d'instance utilisateur différente de null), lever une RuntimeException « UtilisateurDejaConnecteException » ; ceci ne devrait jamais arriver, mais il vaut mieux prévoir le cas ;
   - modifier la variable de classe utilisateur avec le paramètre ;
   - créer une nouvelle instance de MainFrame, la mémoriser dans « mf » et l'ouvrir avec `SwingUtilities.invokeLater` comme dans le programme principal des exemples du cours ;
   - fermer la fenêtre d'identification avec
     ```
     lf.dispose();
     ```
   - remettre lf à null.

Programme principal

1. Créer si ce n'est pas déjà fait le main de l'application.
2. Ajouter l'instruction suivante qui permet de donner aux fenêtres Swing un aspect identique à celui du système d'exploitation sur lequel sera exécutée l'application.

```
UIManager.setLookAndFeel(
                    UIManager.getSystemLookAndFeelClassName());
```

3. Essayer de restituer les données depuis le fichier avec l'instruction

```
GestionnaireSauvegarde.getInstance()
      .restituer(new File(FILE_NAME));
```

   - si une exception FileNotFoundException est levée, faire appel à la méthode d'initialisation de l'application définie plus haut ;
   - dans le cas où une autre exception est produite, on ouvre une fenêtre d'alerte avec le bloc catch qui suit ; l'exception est levée de nouveau pour provoquer l'arrêt du programme.

```
catch (Exception e) {
      // 1er paramètre = fenêtre à laquelle est rattaché ce dialogue
      // 2ème paramètre = texte dans la fenêtre
      // 3ème paramètre = titre de la fenêtre
      // 4ème paramètre = type de message
      JOptionPane.showMessageDialog(null,
                  "Le fichier de données est corrompu.",
                  "Fichier corrompu", JOptionPane.ERROR_MESSAGE);
      throw e;
}
```

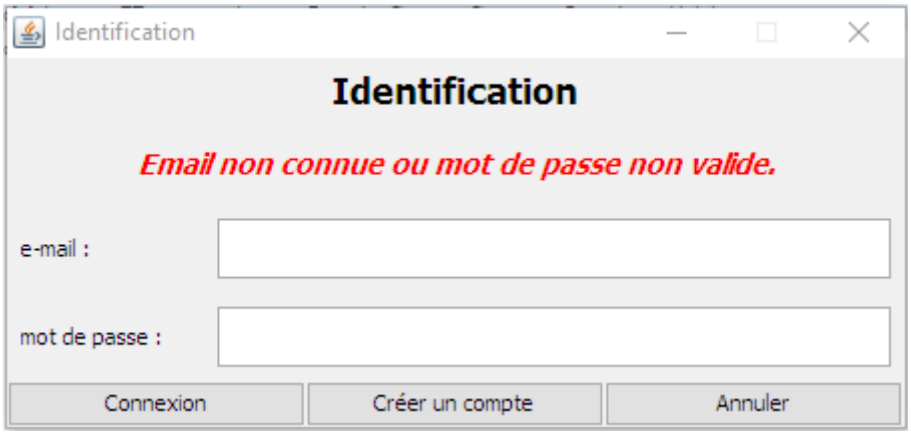
4. Créer une instance de LoginFrame et la mémoriser dans lf.
5. Afficher cette fenêtre.

Exercice 4 : Fenêtre d'identification

Concevoir la fenêtre d'identification dans un package ui.loginframe en la structurant comme il est montré dans le cours pour lui donner l'aspect suivant.

Pour que tous les titres soient identiques dans toutes les fenêtres de l'application, il faudrait concevoir une classe étendant JPanel et nommée ui.commun.TitrePane, par exemple, dont le constructeur prendra en paramètre le titre à afficher sous forme de String.

Pour le message d'erreur, prévoir la zone mais la laisser vide. Le message sera inséré au besoin quand nous définirons les actions des boutons. Pour l'instant, il n'y aura aucune fonctionnalité autre que la fermeture de la fenêtre avec la croix en haut à droite.



Exercice 5 : Capture des exceptions

En vous inspirant du point 3 de la partie « Programme Principal » de l'exercice 3, capturez les exceptions générées par l'ajout d'un client et la sauvegarde dans le fichier pour afficher un panneau d'avertissement. Les exceptions seront levées de nouveau pour provoquer la sortie du programme.

Vous pouvez tester le résultat. Pour l'instant, vous ne devez pouvoir que fermer la fenêtre d'identification.