

Gestion d'une librairie

Le but de cet TP est d'écrire une application complète simulant un site de commerce électronique proposant la vente des livres.

Vous proposerez à chaque fois deux solutions : une solution en JSP et une autre en Servlets java.

1. L'objet Livre

1.1. Ecrire en java un objet Livre

Cet objet devra être écrit sous forme de bean et exploitable par une page jsp et une servlet.

Nous limiterons les attributs du livre à :

- ISBN
- Titre
- Auteur
- Année de parution
- Nombre de pages
- Prix

Ces attributs devront être encapsulés.

L'objet livre aura les méthodes suivantes

```
public void setIsbn(String isbn) ; // positionne l'isbn du livre
public void setTitre(String titre); // positionne le titre du livre
public void setAuteur(String auteur); // position le nom de l'auteur
public void setAnnee(int annee); // positionne l'année de parution
public void setPages(int pages) ; // positionne le nombre de pages
public void setPrix(int prix) ; // positionne le prix du livre

public String getIsbn() ; // récupère la valeur de l'isbn
public String getTitre(); //récupère la valeur du titre
public String getAuteur(); // récupère la valeur de l'auteur du livre
public int getAnnee(); // récupère l'année de parution
public int getPages() ; // récupère le nombre de pages
public int getPrix() ; // récupère le prix du livre
```

Figure 1

1.2. Ecrire une page JSP (livre.jsp)

Cette page permettant de saisir (figure 2) les différentes valeurs du livre et de les afficher (figure 3) après validation.

Saisie informations du livre en JSP

Titre	<input type="text" value="JAVA Servlets"/>
Isbn	<input type="text" value="2-84177-082-6"/>
Auteur	<input type="text" value="J. HUNTER & W. CRAWFORD"/>
Année	<input type="text" value="1999"/>
Nombre de Pages	<input type="text" value="503"/>
Prix	<input type="text" value="60"/>

Figure 2

Affichage informations du livre en JSP

Titre	Isbn	Auteur	Année	Pages	Prix
JAVA Servlets	2-84177-082-6	J. HUNTER & W. CRAWFORD	1999	503	60

Figure 3

1.3. Ecrire une servlet java qui aura les mêmes fonctionnalités que ci-dessus

La servlet devra utiliser le même objet Livre écrit au préalable.

2. Un manager de Livres

Vous avez constaté que les fonctionnalités de la première JSP étaient très réduites. On ne peut par exemple enregistrer qu'un seul livre. Ce serait dommage d'avoir qu'un seul livre à vendre.

Le but de cette partie donc est d'écrire un manager qui va gérer un ensemble de livres avec possibilité d'effectuer certaines opérations sur un Livre. Changer son prix par exemple.

2.1. Ecrire un objet java manager de Livres permettant de gérer et stocker un nombre illimité de livres.

Les livres seront stockés dans une Hashtable. Les principales méthodes de cet objet sont

```
Public class ManagerLivre {
    private Hashtable livresTable=new Hashtable();
    ...
    public void setLivre(String key, Livre livre) ; // Stocke un livre dans livreTable
    public Livre getLivre(String isbn) ; // récupère l'objet Livre dont l'ISBN est isbn
    public Hashtable getLivres() ; // renvoie livreTable contenant tous les livres
    ...
}
```

Figure 4

2.2. Ecrire une page JSP (managerLivre.jsp)

A la première invocation, cette page devra proposer de rajouter des livres en utilisant l'objet ManagerLivre écrit préalablement.

Prix:
 Nombre de Pages:
 Année:
 Auteur:
 Isbn:
 Titre:
 Ajouter un livre en JSP avec le Manager

Figure 5

Après validation, vous devez obtenir un tableau semblable à celui de la figure 6. Cette page devra aussi proposer :

- Une fonctionnalité d'ajout de livres : renvoie vers la page ci-dessus (Figure 5)
- Une fonctionnalité de modification du prix (en cliquant dessus) pour chaque livre . Dans cas, vous devez réafficher toutes les anciennes valeurs, tout en proposant un champs de saisie « Nouveau prix ». Illustration en Figure 7.

Affichage informations du livre en JSP avec le Manager

Titre	Isbn	Auteur	Année	Pages	Prix
JAVA Servets	2-84177-082-6	J. HUNTER & W. CRAWFORD	1999	303	60

[Ajouter un nouveau Livre en JSP](#)

Figure 6

La modification du prix doit proposer le tableau suivant

Modification du prix d'un livre en JSP

Titre: JAVA Servets
 Isbn: 2-84177-082-6
 Auteur: J. HUNTER & W. CRAWFORD
 Année: 1999
 Nombre de Pages: 303
 Nouveau Prix:
 Valider

Figure 7

Après validation, on obtient la page suivante

Affichage informations du livre en JSP avec le Manager

Titre	Isbn	Auteur	Année	Pages	Prix
JAVA Servets	2-84177-082-6	J. HUNTER & W. CRAWFORD	1999	303	100

[Ajouter un nouveau Livre en JSP](#)

Figure 8

NB : Après ajout d'un nouveau livre, on obtient le tableau suivant

Affichage informations du livre en JSP avec le Manager					
Titre	Isbn	Auteur	Année	Pages	Prix
JAVA Servets	2-84177-082-6	J. HUNTER & W. CRAWFORD	1999	303	100
Introductin au JSP	2-7440-0922-9	Larne PEKOWSKY	2000	320	25

[Ajouter un nouveau Livre en JSP](#)

Figure 9

Vous devez pouvoir continuer le procédé infiniment.

2.3. Ecrire une servlet proposant les mêmes fonctions que la JSP

La servlet devra utiliser les mêmes objets Livre et ManagerLivre écrits au préalable.

3. Persistance des données

Vous avez remarqué que tant qu'on a notre navigateur ouvert, on peut ajouter autant de livres qu'on veut, les visualiser et changer leur prix. Un fois qu'on a fermé le navigateur, on est obligé de tout ressaisir car on a tout perdu. C'est une fois de plus dommage.

En principe, pour rendre les données persistantes, on devrait utiliser une base de données. Pour ce TP, nous utiliseront un fichier texte pour « lire » les données et écrire les modifications.

Le fichier contiendra plusieurs lignes et chaque ligne correspond à un livre. Le format est le suivant

```
## Ligne de commentaire
## Format du fichier
## [ISBN] [TITRE] [AUTEUR] [ANNEE] [PAGES] [PRIX]
[2-84177-082-6] [JAVA Servets] [J. HUNTER & W. CRAWFORD] [1999] [303] [60]
...
```

Figure 10

3.1. Nouvelles fonctionnalité dans l'objet ManagerLivre

Rajouter dans l'objet ManagerLivre les méthodes permettant de lire le fichier de données et une autre permettant d'écrire les données dans le fichier.

Vous prendrez garde de sauvegarder l'ancien fichier avant de l'écraser.

NB : Vous pouvez également écrire un nouvel objet pour gérer ces fonctionnalités.

```
// Lit le fichier de données (fileName) contenant les livres et les stocke dans livreTable.
public void setLivre(String fileName);

// Ecrit la liste des livres dans le fichier de données (fileName)
public void writeDataFile(String fileName);
```

Figure 11

La figure 8 devriendra alors

Affichage informations du livre en JSP avec le Manager					
Titre	Isbn	Auteur	Année	Pages	Prix
JAVA Servets	2-84177-082-62-8	J. HUNTER & W. CRAWFORD	1999	303	60

[Ajouter un nouveau Livre en JSP](#) [Sauvegarder le fichier en JSP](#)

Figure 12

3.2. Ecrire une page JSP

Cette page devra lire le fichier de données, et afficher les livres comme à la figure 12. Toutes les fonctionnalités décrites au paragraphe 2 devront être maintenues.

3.3. Ecrire une servlet proposant les mêmes fonctions que la JSP

La servlet devra utiliser les mêmes objets Livre et ManagerLivre écrits au préalable.

3.4. Sérialisation des données

Si vous avez le temps, proposer une solution qui sérialise les données dans un fichier. Vous mettrez donc en place un objet qui va sérialiser L'objet ManagerLivre contenant tous les livres. De la même façon, vous écrirez un objet qui va le désérialiser.

Note : En java, vous pouvez sérialiser certains objets , du moment qu'il implémentent la classe **Serializable**. Lorsque vous récupérez un objet sérialisé, vous pouvez le manipuler comme vous voulez en lui envoyant les messages, car il reste dans l'état où vous l'avez sérialisé.

Exemple de sérialisation

```

...
public class ObjetASerialer {
    private File m_InputFile=null;
    private FileInputStream m_FileInputStream=null;
    private ObjectInputStream m_ObjectInputStream=null;
    private FileOutputStream m_FileOutputStream=null;
    private ObjectOutputStream m_ObjectOutputStream=null;
    private final static String FILENAME="file.dat"; //nom dufichier
    public ObjetASerialer () {}

    public void o_Serialise(Personne personne) {
        try {
            m_InputFile=new File(FILENAME);
            m_FileOutputStream = new FileOutputStream(m_InputFile);
            m_ObjectOutputStream =new ObjectOutputStream(m_FileOutputStream);
            m_ObjectOutputStream.writeObject(personne);
            m_ObjectOutputStream.close();
            m_FileOutputStream.close();
        }catch (Exception e) {System.out.println("Serialisation -> "+e.toString());}
    }

    public Personne o_Derialise() {
        try {
            m_InputFile=new File(FILENAME);
            m_FileInputStream = new FileInputStream(m_InputFile);
            m_ObjectInputStream =new ObjectInputStream(m_FileInputStream);
            return ((Personne)m_ObjectInputStream.readObject());
        }catch (Exception e) {return null;}
    }
}

```

Figure 13

```
public class Personne implements Serializable {
    ...
    private firstName=null;
    private Hashtable tablePersonnes=new Hashtable();
    ...
    public personne () {...}

    public setPersonne () {
        tablePersonnes.put(key, Personne);
    }

    public Personne getPersonnes(String key) { // retourne un objet Personne dont la clé est key
        return this.tablePersonne.get(key);
    }

    public Hashtable getPersonnes() { // retourne une hashtable contenant toutes les personnes
        return this.tablePersonnes;
    }
    ...
}
```

Figure 14

Pour sérialiser l'objet Personne, on fera:

```
...
ObjetASerialiser objetASerialiser = new ObjetASerialiser() ;
Personne personne=new Personne() ;
...
ObjetASerialiser.o_Serialise(personne) ;
```

Figure 15

Cette action écrira dans un fichier binaire (file.dat), l'objet Personne. Cette donnée restera persistante.

Pour récupérer l'objet sérialisé (dans l'état où il a été sauvegardé), on fera

```
...
ObjetASerialiser objetASerialiser = new ObjetASerialiser() ;
Personne personne=objetASerialiser.o_Derialise() ;
...
```

Figure 16

On pourra alors effectuer les opérations sur l'objet Personne comme si on l'avait instancié. Par exemple :

```
Hashtable table=personne.getPersonnes() ;
```

4. Simulation d'achats des livres

Après avoir écrit des outils nous permettant d'initier, de gérer nos livres, nous allons maintenant les proposer à la vente.

Cela suppose que nous sommes capables de gérer un panier : ajout des livres dans le panier, visualisation du contenu à tout moment, suppression d'un livre du panier.

Vous devez prévoir aussi une fonctionnalité de login / password qui vous permettra de fidéliser votre client, de suivre ses commandes et éventuellement de lui offrir des bons de réductions au bout d'un certain nombre de commandes. Dans ce cas, le client qui fera le pas de s'identifier sur votre site (par son login / password), pourra avoir accès à l'état des ses commandes.

Lorsque le client termine sa commande, vous devez lui proposer un récapitulatif des livres qu'il a achetés, avec éventuellement la possibilité d'en ajouter ou d'en supprimer.

Vous devez aussi lui proposer un texte le rassurant sur la confidentialités et la sécurisation des données qui seront transmises.

Lorsqu'il est d'accord, vous devez saisir ses coordonnées, puis lui proposer un login / password pour les raisons évoquées ci-dessus. En aucun cas, cette information devra être obligatoire.

Une fois que vous avez toutes les informations, vous devez les lui envoyer par mail au client.

4.1. Ecrire en java un objet d'envoi de mail