NTIC - Examen n°1 : JSTL et JSF

Objectif :

Le but de ce TP est de créer un site bancaire

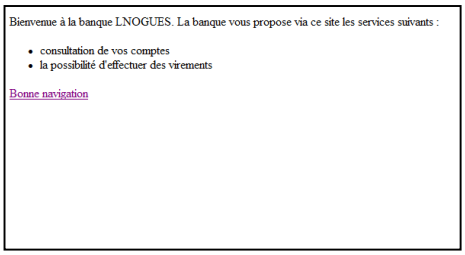
Technologies :

Java EE 5 : JSTL, JSF

IDE : Eclipse

Serveur : Tomcat

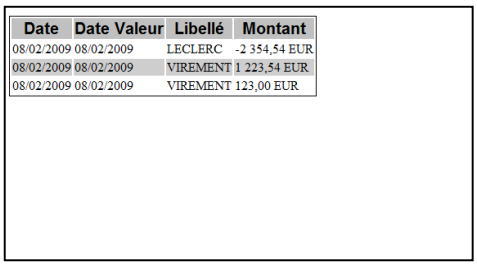
Maquette :



Accueil

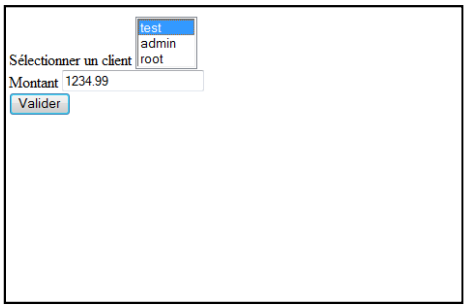
Login

Success

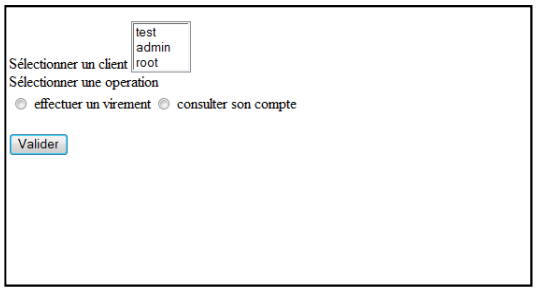


Date	Date Valeur	Libellé	Montant
08/02/2009	08/02/2009	LECLERC	-2 354,54 EUR
08/02/2009	08/02/2009	VIREMENT	1 223,54 EUR
08/02/2009	08/02/2009	VIREMENT	123,00 EUR

ConsulterCompte



EffectuerVirement



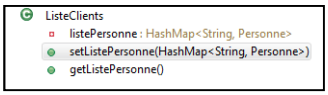
Administration

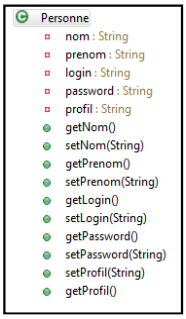
Déroulement du TP

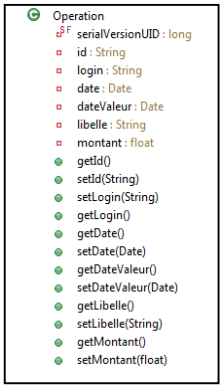
- Créez un nouveau projet « web » avec Eclipse, avec comme configuration « Dynamic web module 2.5 » et « JavaServer Faces v1.2 »
- **Nommez ce projet de la manière suivante : exam1_XYYY (avec X = 1ère lettre du prénom et YYY = trois 1ères lettres du nom.**
- Ajoutez les librairies nécessaires pour JSF
- Ajoutez dans un répertoire WebContent/css le fichier styles.css

Les beans

- Créez dans src un package bean
- Créez dans ce package les objets suivants : Operation, Personne, ListeClients, ListeOperations







Les classes métier

Deux classes métier vont permettre la gestion des opérations et des clients.

- Créez dans src un package metier
- Créez dans ce package une nouvelle classe MetierPersonnes

La methode getPersonnes initialise la liste des clients référencés :

```
public static ListeClients getPersonnes() {
        if (personnesInstance == null) {
                personnesInstance = new ListeClients();

                Personne personne1 = new Personne();
                personne1.setLogin("test");
                personne1.setPassword("test");
                personne1.setNom("Test");
                personne1.setPrenom("Test");

                Personne personne2 = new Personne();
                personne2.setLogin("root");
                personne2.setPassword("root");
                personne2.setNom("Root");
                personne2.setPrenom("Root");

                Personne personne3 = new Personne();
                personne3.setLogin("admin");
                personne3.setPassword("admin");
                personne3.setNom("Admin");
                personne3.setPrenom("Admin");
                personne3.setProfil("ADMIN");

                HashMap<String, Personne> listePersonne = new HashMap<String, Personne>();
                listePersonne.put(personne1.getLogin() + personne1.getPassword(),
                                personne1);
                listePersonne.put(personne2.getLogin() + personne2.getPassword(),
                                personne2);
                listePersonne.put(personne3.getLogin() + personne3.getPassword(),
                                personne3);

                personnesInstance.setListePersonne(listePersonne);

        }
        return personnesInstance;
}
```

La classe contient également une methode qui retourne une Personne à partir de son login et password. La signature de cette méthode est la suivante :

```
public static Personne getPersonne(String login, String password)
```

- Créez dans ce package une nouvelle classe MetierOperations

La methode getListeOperations initialise la liste des opérations référencés :

```
public static ListeOperations getListeOperations() {
        if(listeOperations == null){
                listeOperations = new ListeOperations();

                Operation operation1 = new Operation();
                operation1.setId("1");
                operation1.setLogin("test");
                operation1.setDate(new Date());
                operation1.setDateValeur(new Date());
                operation1.setLibelle("JULES");
                operation1.setMontant(-150.99f);

                Operation operation2 = new Operation();
                operation2.setId("1");
                operation2.setLogin("test");
                operation2.setDate(new Date());
                operation2.setDateValeur(new Date());
                operation2.setLibelle("LECLERC");
                operation2.setMontant(-334.54f);

                Operation operation3 = new Operation();
                operation3.setId("3");
                operation3.setLogin("test");
                operation3.setDate(new Date());
                operation3.setDateValeur(new Date());
                operation3.setLibelle("BOUCHER");
                operation3.setMontant(-122.21f);

                Operation operation4 = new Operation();
                operation4.setId("4");
                operation4.setLogin("test");
                operation4.setDate(new Date());
                operation4.setDateValeur(new Date());
                operation4.setLibelle("PHARMACIE");
                operation4.setMontant(-13.87f);


                Operation operation5 = new Operation();
                operation5.setId("5");
                operation5.setLogin("test");
                operation5.setDate(new Date());
                operation5.setDateValeur(new Date());
                operation5.setLibelle("COIFFEUR");
                operation5.setMontant(-1120f);

                Operation operation6 = new Operation();
                operation6.setId("6");
                operation6.setLogin("root");
                operation6.setDate(new Date());
                operation6.setDateValeur(new Date());
                operation6.setLibelle("LECLERC");
                operation6.setMontant(-2354.54f);

                ArrayList<Operation> operations = new ArrayList<Operation>();
                operations.add(operation1);
                operations.add(operation2);
                operations.add(operation3);
                operations.add(operation4);
                operations.add(operation5);
                operations.add(operation6);

                listeOperations.setListeOperations(operations);
        }
        return listeOperations;
}
```

La classe contient également deux autres méthodes. Les signatures de ces méthodes sont les suivantes :

```
public static ArrayList<Operation> getOperations(String login)
```

```
public static void setListeOperations(ListeOperations o)
```

Page d'accueil

La page d'accueil (index.jsp) présentera en quelques mots les services de la banque (utilisez pour cela les tags JSTL et un fichier externe Message_fr.properties placé à la racine de src) elle proposera un lien pour accéder à la page de login.

Page de login

Pour accéder à leur compte les personnes doivent s'authentifier.

Vous créerez donc un formulaire login.jsp en utilisant les composants JSF suivants :

- `<html:form>`
- `<html:outputText>`
- `<html:inputText>`
- `<html:inputSecret>`
- `<html:commandButton>`

Pour remplir le formulaire, vous utiliserez un managedBean.

Pour cela, créez un package managedBean qui contiendra une classe Login.java. Cette classe sera un bean qui aura deux attributs « login » et « password », les getters et setters de ces attributs, ainsi qu'une méthode « validate » qui vérifiera si la personne existe dans la liste des clients et redirigera vers une page success ou error.

N'oubliez pas d'ajouter à votre fichier faces-config.xml votre managed-bean et votre navigation-rule.

Créez des fichiers error.jsp et success.jsp.

Consulter son compte

Modifiez votre fichier success.jsp en y ajoutant des liens vers les pages consulterCompte.jsp et effectuerVirement.jsp grace aux balises :

- `<html:outputLink>`
- `<core:verbatim>`

Créez une page consulterCompte.jsp qui contiendra une datatable pour présenter les lignes d'opérations.

Afficher les colonnes suivantes : Date, Date de valeur, Libellé, Montant

Utilisez notamment les balises suivantes pour la mise en forme des dates et montant :

- `<core:convertDateTime>`
- `<core:convertNumber>`

Créez un managedBean Operations qui aura une méthode :

```
        public ArrayList<Operation> getOperations() {
                String login = getLoginValue();
                return MetierOperations.getOperations(login);
        }
```

Pour récupérer le login dans le managedBean, vous pourrez utiliser la méthode suivante :

```
    public String getLoginValue() {
                FacesContext facesContext = FacesContext.getCurrentInstance();
                Login login = (Login) facesContext.getExternalContext().getSessionMap().get("login");
                return login.getLogin();
    }
```

Ajouter votre managedBean à votre fichier faces-config.xml

Effectuer un virement

Créez une jsp effectuerVirement.jsp qui contiendra une liste des clients, un champ de saisie pour le montant et un bouton de validation.

Pour créer la liste vous utiliserez les balises suivantes :

```
<html:selectOneListbox value="#{virement.oneListBoxValues}" >
        <core:selectItems value="#{virement.manyAndOneListBoxItems}" />
</html:selectOneListbox>
```

Dans votre ManagedBean, il faudra déclarer les attributs et méthodes suivantes :

```
        private String oneListBoxValues = "";

        public Collection<SelectItem> getManyAndOneListBoxItems() {
                Collection<SelectItem> manyAndOneListBoxItems = new ArrayList<SelectItem>();
                ListeClients listeClients = MetierPersonnes.getPersonnes();
                Iterator<Personne> it = listeClients.getListePersonne().values().iterator();
                while (it.hasNext()){
                        manyAndOneListBoxItems.add(new SelectItem(it.next().getLogin()));
                }
                return manyAndOneListBoxItems;
        }

        public String getOneListBoxValues() {
                return this.oneListBoxValues;
        }

        public void setOneListBoxValues(String p) {
                oneListBoxValues = p;
        }
```

Votre méthode de validation du formulaire devra ajouter une ligne sur chacun des 2 clients concernés. Une avec un montant positif pour la personne destinataire du virement et une avec un montant négatif pour la personne à l'origine du virement.

Solde du compte

Ajoutez dans l'écran de consultation du compte (consulterCompte.jsp), sous la datatable, une ligne indiquant le solde du compte. Le solde sera un nouvel attribut à ajouter à l'objet Personne. Il sera initialisé à la création des personnes avec des valeurs de votre choix. Le solde sera ensuite mis à jour à chaque saisie d'opération.

Administration

Pour créer la page d'administration de la banque, vous allez permettre la redirection vers la page administration.jsp quand l'utilisateur qui se connectera aura un profil « ADMIN » de renseigné.

La page d'administration proposera la liste des clients de la banque (comme sur la page de virement) ainsi que 2 actions possibles (boutons radio) :

- Consulter les comptes de la personne sélectionnée.
- Effectuer un virement à la place de la personne sélectionnée.

Pour créer des boutons radios, vous utiliserez les tags suivants :

- `<html:selectOneRadio>`
- `<core:selectItems>`

Leur utilisation dans le bean managé est la même que pour les listbox.

Vous modifierez le fonctionnement de la consultation des comptes et des virements pour que ces pages prennent en compte non plus la personne connectée mais la personne sélectionnée dans le cas d'un utilisateur administrateur.

Création de compte

Dans la page d'administration vous allez ajouter un lien vers une page de création d'un compte client. Cette page sera composée d'un formulaire contenant les champs Nom, Prénom, Login, Mot de passe et solde initial. A la validation de ce formulaire, un nouvel utilisateur sera créé.

BON COURAGE…