Les formulaires

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en Programmation par contrainte (IA)
Ingénieur en Génie logiciel

elmouelhi.achref@gmail.com

Introduction

Un formulaire

- un outil graphique
  - que nous créons avec le langage de description HTML
  - que nous gérons avec un langage de programmation tel que PHP, Servlets Java...
- il permet à l'utilisateur de saisir des données
- et de les envoyer vers une autre page, vers une base de données...

Exemple

Page JSP d'ajout d'une personne `ajoutPersonne.jsp`

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Ajout</title>
  </head>
  <body>
    <form method="post" action="ajoutPersonne">
      <div>Formulaire d'ajout d'une Personne</div>
        <div><label for="nom">Nom *</label>
          <input type="text" id="nom" name="nom" value="" />
        </div>
        <div><label for="prenom">Prenom *</label>
          <input type="text" id="prenom" name="prenom" value="" />
        </div>
        <input type="submit" value="Ajouter" />
        </form>
    </body>
</html>
```

Explication

Déclaration d'un formulaire :

```
<form method="post ou get" action="url-pattern d'une
   servlet">
</form>
```

Les attributs d'un formulaire

- `action` : indique une url correspondant à l'appel d'une servlet (tel qu'on l'a déclarée dans le `web.xml`)
- `method` : concerne l'envoi de données et peut prendre deux valeurs.
  - `get` : c'est la méthode `doGet()` de la servlet qui sera appelée.
  - `post` : c'est la méthode `doPost()` de la servlet qui sera appelée.

Extrait de `web.xml`

```
<servlet-name>TestServlet</servlet-name>
<url-pattern>/ajoutPersonne</url-pattern>
```

Explication

Par rapport à l'exemple précédent

- Quand on saisit l'url `.../ajoutPersonne` dans le navigateur, c'est la méthode `doGet()` qui sera appelée.
- Quand on valide le formulaire, il est préférable d'appeler la méthode `doPost()`.

Exemple

Dans la servlet

```
protected void doGet(HttpServletRequest request, HttpServletResponse
   response) throws ServletException, IOException {
  this.getServletContext().getRequestDispatcher("/WEB-INF/ajoutPersonne
     .jsp").forward(request, response);
}
protected void doPost(HttpServletRequest request, HttpServletResponse
   response) throws ServletException, IOException {
  String nom = request.getParameter("nom");
  String prenom = request.getParameter("prenom");
  Personne p = new Personne();
  p.setNom(nom);
  p.setPrenom(prenom);
  PersonneDaoImpl daop = new PersonneDaoImpl();
  daop.save(p);
  this.getServletContext().getRequestDispatcher("/WEB-INF/confirmation.
     jsp").forward(request, response);
}
```

Explication

Contrôler la validation du contrôleur

- Interdire l'insertion vide (pour les deux champs nom et prénom)
- Pas de chaîne avec une longueur inférieure à deux
- Première lettre en majuscule
- Afficher un message en cas de format non-respecté
- Garder les valeurs saisies affichées dans le formulaire, en cas d'erreur

Exemple

Dans la servlet, une méthode pour contrôler la saisie

```
public boolean verifChaine(String s) {
  char c = s.charAt(0);
  if (s.length() < 2 || !(c >= 'A' && c <= 'Z'))
    return false;
  for(int i = 0 ; i< s.length(); i++) {
    c = s.charAt(i);
    if (!(c >= 'a' && c <= 'z')&& !(c >= 'A' && c <=
        'Z'))
      return false;
    }
  return true;
}
```

Exemple

Préparons le formulaire pour les nouvelles améliorations

```
<form method="post" action="ajoutPersonne">
  <div>Formulaire d'ajout d'une Personne</div>
    <div><label for="nom">Nom *</label>
    <input type="text" id="nom" name="nom" value="${
       nomSaisi }"/>
    ${ nomIncorrect }
    </div>
    <div><label for="prenom">Prenom *</label>
    <input type="prenom" id="prenom" name="prenom"
       value="${ prenomSaisi }" />
    ${ prenomIncorrect }
    </div>
    <input type="submit" value="Ajouter"  />
</form>
```

Exemple

Dans la servlet, une méthode pour contrôler la saisie

```
protected void doPost(HttpServletRequest request, HttpServletResponse
   response) throws ServletException, IOException {
  String nom = request.getParameter("nom");
  String prenom = request.getParameter("prenom");
  boolean verifNom = verifChaine(nom);
  boolean verifPrenom = verifChaine(prenom);
  request.setAttribute("nomSaisi", nom);
  request.setAttribute("prenomSaisi",prenom);
  if (!verifNom)
    request.setAttribute("nomIncorrect", "format␣incorrect");
  if (!verifPrenom)
    request.setAttribute("prenomIncorrect","format␣incorrect");

  if (!verifNom || !verifPrenom)
    this.getServletContext().getRequestDispatcher("/WEB-INF/
       ajoutPersonne.jsp").forward(request, response);
  else
    this.getServletContext().getRequestDispatcher("/WEB-INF/
       confirmation.jsp").forward(request, response);
}
```

Explication

Une deuxième solution

- Utiliser les exceptions pour afficher les erreurs

Explication

Une deuxième solution

- Utiliser les exceptions pour afficher les erreurs

- On ne modifie pas la vue

Exemple

La méthode de contrôle de saisie avec utilisation d'exception

```
public void verifChaine(String s) throws Exception {
  char c = s.charAt(0);
  if (s.length() < 2)
    throw new Exception("La chaine doit comporter au
       moins deux caracteres");
  if (!(c >= 'A' && c <= 'Z'))
    throw new Exception("La chaine doit commencer
       par une lettre en majuscule");
  for(int i = 0 ; i< s.length(); i++) {
    c = s.charAt(i);
    if (!(c >= 'a' && c <= 'z')&& !(c >= 'A' && c <=
       'Z'))
      throw new Exception("La chaine ne peut
         contenir que des lettres");
  }
}
```

Exemple

```
protected void doPost() {
  String nom = request.getParameter("nom");
  String prenom = request.getParameter("prenom");
  request.setAttribute("nomSaisi", nom );
  request.setAttribute("prenomSaisi",prenom);
  try {
    verifChaine(nom);
  } catch (Exception e) {
    request.setAttribute("nomIncorrect", e.getMessage() );
    this.getServletContext().getRequestDispatcher("/WEB-INF/
      ajoutPersonne.jsp").forward(request, response);
  return;
  }  try {
    verifChaine(prenom);
  } catch (Exception e) {
    request.setAttribute("prenomIncorrect", e.getMessage() );
    this.getServletContext().getRequestDispatcher("/WEB-INF/
      ajoutPersonne.jsp").forward(request, response);
    return;
  }
  this.getServletContext().getRequestDispatcher("/WEB-INF/confirmation.
     jsp").forward(request, response);
}
```

Explication

Une troisième solution

- déplacer les contrôles dans une classe intermédiaire entre le bean et le contrôleur (on l'appel généralement classe métier)