TP02
Les java server pages (JSP)

Objectifs

Gérer des pages web dynamiquement (JSP)

1. **Produire une page HTML riche et dynamique**
2. **Distinguer les différents éléments d'une page JSP**
3. **Gérer les formulaires avec JSP**
4. **Communication Servlet/JSP (passage des attributs)**
5. **Incorporer du java avec du HTML**
6. **Portée des variables**

1. Création de la première page JSP

a. Créer, dans le workspace « **D:\Atelier_JEE\workspace** », un nouveau projet web dynamique nommé « **web_app_tp02** ».

b. Sélectionner le dossier « **WebContent** » et choisir la commande « **New/JSP File** ».

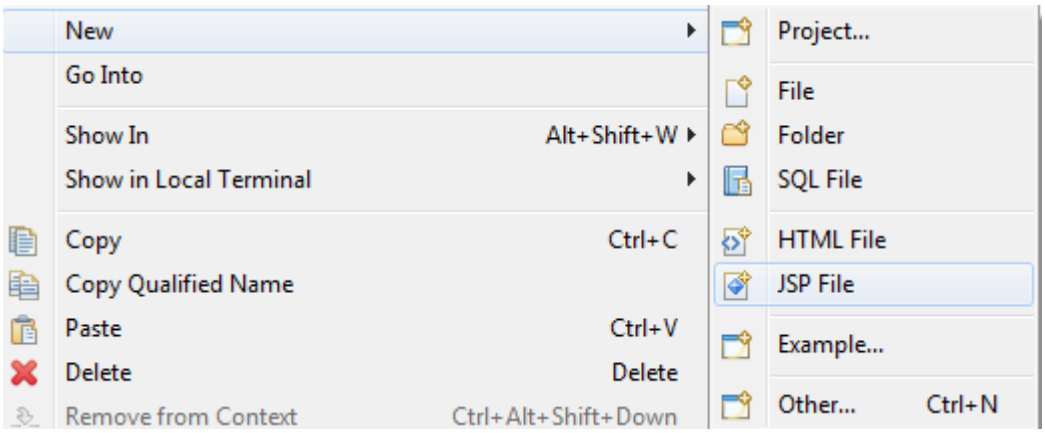

Figure 1 : nouvelle JSP

c. Attribuer le nom « **premier.jsp** » à cette page et cliquer sur « **Finish** ».

- **Un code JSP est un code HTML dans lequel nous ajoutons du code JAVA entre <% et %>.**
- **Au contraire du java script, JSP est exécuté côté serveur (en utilisant le moteur de Servlet (Ex : Tomcat).**

d. Ajoutons en premier lieu un simple message d'**aspect statique** comme suit :

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
                    Première JSP
</body>
</html>
```

- **@page s'appelle une directive placée en premier lieu dans un code JSP et qui présente éventuellement les attributs suivants :**
- **language : pour spécifier le langage de programmation « java ».**
- **contentTpe : pour préciser le type de contenu à afficher**
- **import : les packages à importer**

e. Pour lancer l'exécution, sélectionner, dans le volet « **Project Explorer** », la JSP et choisir la commande « **Run As/ Run On Server** » avec le bouton droit.

f. Ajoutons maintenant du code JAVA (**aspect dynamique**). Par exemple, afficher la date système du serveur. Pour se faire, ajouter, dans le corps de page (balise body>, le code suivant :

```
<h2><% out.println (new Date()); %></h2>
```

- **Ce code mis entre <% et %> est un code JAVA qui utilise la méthode « println() » l'objet « out » pour afficher pour afficher la date système.**

g. Ce code génère une erreur : la classe **Date** n'est pas reconnue par le compilateur. Pour spécifier le package de cette classe ajouter, tout en haut de la page, le code suivant :

```
<%@page import="java.util.Date"%>
```

h. L'affichage de la date peur être réalisé autrement avec la notion d'**expression**. En effet, avec JSP, il est possible d'afficher les valeurs des variables sans utiliser « **out.print** » :

Il suffit de mettre la variable entre **<%=** et **%>** **sans point-virgule** comme suit :

```
<h2><%= new Date() %></h2>
```

•L'utilisation d'une expression permet d'alléger le code et le rendre plus lisible et surtout plus simple en évitant les appels à la méthode « out.println()»

2. HTML et JSP

•Parmi les avantages d'utilisation des JSPs, nous citons la possibilité d'alterner entre le HTML et le JAVA.

• Un bloc JAVA peut être découpé en plusieurs parties délimitées par <% et %>. Entre ces parties, le développeur peut insérer un code HTML statique. Ainsi, il est apte de produire un résultat dynamique et complexe.

- **Exemple1 : réaliser le code suivant :**

```
<%  for (int i=1; i<=5; i++)
{ %>
      <h<%=i %>> Titre de niveau :<%=i %> </h<%=i %>>
 <%} %>
```

- **Exemple2 : réaliser le code suivant :**

```
<table border= "1">
    <%for (int i= 1; i<=L ; i++) {%>
    <tr>
        <%for (int j= 1; j<=C ; j++) {%>

        <td> <%=i*j %></td>
        <%} %>
    </tr>
    <%} %>
```

- **Exercice :**

Reprendre le code de l'exemple 2 et le modifier pour afficher une matrice de multiplication (**4 * 3**) qui :

✓Affiche en **bleu** le texte d'une cellule ayant une valeur paire

✓Et Affiche en **rouge** le texte d'une cellule ayant une valeur impaire

3. Les formulaires JSP et Servlet

a. Créer le formulaire suivant (nommé **connexion.jsp**) :

```
<%@ page language="java"
     contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Connexion</title>

<link rel ="stylesheet" type ="text/css"
href="<%=request.getContextPath()%>/css/style.css"/>

</head>
<body >
<div>
Veuillez saisir vos paramètres de connexion :
</div>
<div>
<form action="ConnexionAction" method="POST">

<table>
<tr>
<td  class ="label">Nom:</td>
<td><input type="text" name="login" value =""/></td></tr>
<tr>
<td  class ="label">Mot de passe:</td>
<td>
<input type="password" name="password" value ="" />
</td></tr>
<tr>
<td align="center" colspan="2">
<input type="submit" value="ok " />
<input type="reset" value="Annuler" />
</td></tr>
</table>
</form>
</div>
</body> </html>
```

Type du fichier de style

Nom du fichier de style

Servlet qui reçoit les paramètres du formulaire

Méthode qui traite la requête

Paramètres du formulaires envoyés avec la requête

b. La page « **connexion.jsp** » est associé à une feuille de style CSS nommée « style.css » située dans le sous-dossier « **css** » du « **WebContent** ». Le nom de l'application est spécifié par l'expression JSP « **<%=request.getContextPath()%>** » et le code du fichier CSS est le suivant :

```
div
{
 border : 1px dotted gray;
 margin: 10px;
 padding: 50px;
 color: blue;
}

.label {
    color: green;
    font-weight: bold;
}
```

c. Créer une servlet nommée « **ConnexionAction.java** » , dans un package « **controller** » qui récupère les valeurs des deux champs « **login** » et « **password** » , affiche leurs valeurs et affiche un message d'erreur si la valeur d'un paramètre du formulaire est vide. Voici le code de récupérations des paramètres et leur affichage dans la méthode « **doPost()** » :

```
        String login = (String) request.getParameter("login");
        String password = (String) request.getParameter("password");
        PrintWriter out = response.getWriter();
        out.println("Login:"+login);
        out.println("Pasword:"+password);
// validation
      if (login!=null && login.equals(""))
      {
      out.append("Champs login vide. Merci de spécifier une valeur..");
      }
      if (password!=null && password.equals(""))
      {
     out.append("Champs password vide. Merci de spécifier une valeur..");
      }
```

4. Le passage des attributs

- **Il est possible de passer des données (dans l'objet « HttpServletRequest » sous forme d'attributs définis par des couples (« key », « value ») :**
  - **Le premier argument est une chaine de caractères qui représente le nom de l'attribut**
  - **et le deuxième argument est un objet qui représente la valeur de l'attribut.**
- **Vous utilisez la méthode «setAttribute(String key, Object objet) » pour l'écriture et la méthode «Object getAttribute(String key)» pour la procédure lecture.**
- **Il existe d'autres méthodes pour gérer les attributs (suppression, modification,..(voir documentation)**

a. Modifier le code de la méthode « **doPost()** » pour réaliser les actions suivantes :

- ✓ Placer le message d'erreur comme un attribut dans l'objet « **request** »
- ✓ Placer les variables « **login** » et « **password** » comme attributs dans l'objet « **request** »
- ✓ Retourner à la page « **connexion.jsp** »

Voici le nouveau code de la méthode « **doPost()** » :

```
            PrintWriter out = response.getWriter();
            //Récupérer la valeur du paramètre login"
            String l = request.getParameter("login");
            //Afficher la valeur du paramètre
            out.println("La valeur du nom est:"+l);
            //Récupérer la valeur du paramètre password"
            String pwd = request.getParameter("password");
            //Afficher la valeur du paramètre
            out.println("La valeur du mot de passe est:"+pwd);

            // validation
            String message ="";
            if (l!=null && l.equals(""))
            {
                  message ="Champs login vide. Merci de spécifier une valeur..";
            }
            if (pwd!=null && pwd.equals(""))
            {
                  message="Champs password vide. Merci de spécifier une valeur..";
            }
//Placer le message d'err comme un attribut nommé "msg" dans l'objet "request"
            request.setAttribute("msg", message);
//Placer le login comme un attribut nommé "login" dans l'objet "request"
            request.setAttribute("login", l);
//Placer le password comme un attribut nommé "password" dans l'objet "request"
```

```
            request.setAttribute("password", pwd);
//Se rediriger vers la page "connexion.jsp"
            request.getRequestDispatcher("connexion.jsp").forward(request,
response);
```

b. Modifier le code de la page « **connexion.jsp** » pour :

- ✓ Récupérer les attributs placés dans l'objet « **request** »
- ✓ Afficher le message d'erreur.
- ✓ Remplacer les valeurs nulles des attributs par une chaine de caractères vide.
- ✓ Remplir les champs du formulaire avec les valeurs des attributs correspondants :

Voici le nouveau code de la page « **connexion.jsp** » :

```
<%@ page language="java"
     contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<% String login = (String) request.getAttribute("login");
   String password = (String) request.getAttribute("password");
   String message = (String) request.getAttribute("msg");
    if (login==null) login="";
   if (password==null) password="";
   if (message==null) message="";
 %>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Connexion</title>
<link rel ="stylesheet" type ="text/css"
href="<%=request.getContextPath()%>/css/style.css"/>
</head>

<body >

<% if (message!=null && !message.equals("")) { %>
<div class ="erreur"><h1><%=message %>fff</h1></div>
 <%  }  %>

<div>
Veuillez saisir vos paramètres de connexion :
</div>
<div>
<form action="ConnexionAction" method="POST">
<table>
<tr>
<td  class ="label">Nom:</td>
<td><input type="text" name="login" value ="<%=login%>"/></td>
</tr>
```

```
<tr>
<td  class ="label">Mot de passe:</td>
<td>
<input type="password" name="password" value ="<%=password%>" />
</td>
</tr>
<tr>
<td align="center" colspan="2">
<input type="submit" value="ok " />
<input type="reset" value="Annuler" />
</td>
</tr>
</table>
</form>
</div>
</body> </html>
```

c. Dans le fichier «**style.css**», ajouter une nouvelle classe « **erreur** » ayant le code suivant :

```
.erreur {
color: red;
font-weight: bold;
}
```

d. Remarquer qu'un seul message d'erreur est affiché (en effet, le dernier message écrase les autres). Modifier le mode d'envoi des erreurs pour passer tous les messages d'erreur dans un objet « **ArrayList** ». En cas d'existence d'erreurs, la servlet redirige appelle la vue « **connexion.jsp** » en envoyant les attributs nécessaires. Sinon, la servlet appelle la page d'accueil « **accueil.jsp** » pour afficher un message de BienVenue.

Voici le nouveau code de la méthode « **doPost()** » :

```
        PrintWriter out = response.getWriter();
        //Récupérer la valeur du paramètre login"
        String l = request.getParameter("login");
        //Afficher la valeur du paramètre
        out.println("La valeur du nom est:"+l);
        //Récupérer la valeur du paramètre password"
        String pwd = request.getParameter("password");
        //Afficher la valeur du paramètre
        out.println("La valeur du mot de passe est:"+pwd);
        // Création d'un tableau vide d"erreurs
        ArrayList<String> erreurs =new ArrayList<String>();
        // validation
        if (l!=null && l.equals(""))
        {
```

```
            erreurs.add("Champs login vide. Merci de spécifier une valeur..");
            }
            if (pwd!=null && pwd.equals(""))
            {
            erreurs.add("Champs password vide. Merci de spécifier une valeur..");
            }
            if (erreurs.isEmpty())
            {// Aller à la page d'accueil
      request.getRequestDispatcher("accueil.jsp").forward(request, response);
            }else
            {
//Placer le login comme un attribut nommé "login" dans l'objet "request"
            request.setAttribute("login", l);
//Placer le password comme un attribut nommé "password" dans l'objet "request"
            request.setAttribute("password", pwd);
//Placer le tableau des erreurs comme attribut "tab_err" dans l'objet "request"
            request.setAttribute("tab_err", erreurs);
            //Se rediriger vers la page "connexion.jsp"
      request.getRequestDispatcher("connexion.jsp").forward(request, response);
      }
```

e. Voici le nouveau code de la page « **connexion.jsp** »

```
<%@ page language="java" import ="java.util.ArrayList, java.util.Iterator"
      contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<% String login = (String) request.getAttribute("login");
   String password = (String) request.getAttribute("password");
   ArrayList<String> err = (ArrayList<String>) request.getAttribute("tab_err");
    if (login==null) login="";
   if (password==null) password="";
%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Connexion</title>
<link rel ="stylesheet" type ="text/css"
href="<%=request.getContextPath()%>/css/style.css"/>
</head>
<body >
<% if (err!=null && !err.isEmpty()) { %>
<div class ="erreur">
<ul>
<%  for(Iterator<String> it =err.iterator(); it.hasNext();)
{%>
<li> <%=it.next()%></li>
<% } %>
</ul>
</div>
 <%  }  %>
<div>
Veuillez saisir vos paramètres de connexion :
</div>
<div>
<form action="ConnexionAction" method="POST">
```

```
<table>
<tr>
<td  class ="label">Nom:</td>
<td><input type="text" name="login" value ="<%=login%>"/></td>
</tr>
<tr>
<td  class ="label">Mot de passe:</td>
<td>
<input type="password" name="password" value ="<%=password%>" />
</td></tr>
<tr>
<td align="center" colspan="2">
<input type="submit" value="ok " />
<input type="reset" value="Annuler" />
</td></tr>
</table>
</form>
</div>
</body> </html>
```

5. Portée des variables (scope)

Une variable peut être visible de trois manières :

- **request : accessible au niveau d'une requête http, sa durée de vie est limitée à une seule requête. (HttpServletRequest)**
- **session : accessible par toutes les requêtes d'un même client « navigateur » (HttpSession)**
- **application : accessible par toute l'application (même pour des navigateurs différents (ServletContext)**

a. Tester la portée d'une variable « **prenom** » passée à une nouvelle servlet « **HelloWorldScope** » :

```
package test;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@WebServlet("/HelloWorldScope")
public class HelloWorldScope extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// Préciser le type du contenu à généré
response.setContentType( "text/html" );
```

```
// Récupérer l'objet d'écriture de la réponse
PrintWriter out = response.getWriter();
//Récupérer la valeur du paramètre prenom"
String prenom = request.getParameter("prenom");
//Référencer à la session
HttpSession session = request.getSession(true);
//Placer la variable dans la session si elle est non nulle
if (prenom !=null)
      session.setAttribute("prenom", prenom);
//référencer au contexte  (portée application)
ServletContext contexte = request.getServletContext();
//Placer la variable dans le contexte si elle est non nulle
if (prenom !=null)
      contexte.setAttribute("prenom", prenom);
//Paser à la page JSP "scope.jsp"
request.getRequestDispatcher("scope.jsp").forward(request, response);
      }
}
```

Voici le code de la page « **scope.jsp** » :

```
<%@ page language="java" contentType="text/html; charset=windows-1256"
    pageEncoding="windows-1256"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Scope</title></head>
<body>
BienVenue: <br>
<%
//utiliser les variables pré-définies (request, session, application)
  String prenom = request.getParameter("prenom");
//Afficher les valeus
out.println("depuis la requete :"+prenom+"<br>");
out.println("depuis la session:"+(String)session.getAttribute("prenom")+"<br>");
out.println("depuis le contexte:"+(String)application.getAttribute("prenom")+"<br>");
%>
</body></html>
```

b. Lancer en premier lieu l'url :
   http://localhost:8080/web_app_tp02/HelloWorldScope?prenom=Ali
c. Puis Lancer sur le même navigateur l'url :
   http://localhost:8080/web_app_tp02/HelloWorldScope
d. Puis Lancer sur un autre navigateur l'url :
   http://localhost:8080/web_app_tp02/HelloWorldScope
e. Interpréter les résultats

6. Mode connecté

a. Revenons à la servlet « **ConnexionAction** » pour mettre les deux paramètres « **login** » et « **password** » comme attributs dans la session avant de rediriger la requête vers la page d'accueil

Voici le code d'appel à la page d'accueil :

```
if (erreurs.isEmpty())
        {
            HttpSession session =request.getSession(true);
//Placer le login comme un attribut nommé "login" dans l'objet "session"
            session.setAttribute("login", l);
//Placer le password comme un attribut nommé "password" dans l'objet "session"
            session.setAttribute("password", pwd);
// Aller à la page d'accueil
      request.getRequestDispatcher("accueil.jsp").forward(request, response);
            }
```

Voici la nouvelle version de la page « accueil.jsp » :

```
<%@ page language="java"
      contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Connexion</title>
<link rel ="stylesheet" type ="text/css"
href="<%=request.getContextPath()%>/css/style.css"/>
</head>
<body>
<% String login = (String) session.getAttribute("login");
   String password = (String) session.getAttribute("password");
   //tester l'existence des attributs dans la session
    if ((login==null) || (password==null))
{
      //retourner à la page d'authentification
   request.getRequestDispatcher("connexion.jsp").forward(request, response);
}
%>
<div>
BienVenue: <%=login %>
</div>
</body>
</html>
```

b. Ajouter un lien « **Deconnexion** » qui appelle une Servlet « **DeconnexionAction** » qui mets la valeur « **null** » à chacun des attributs « **login** » et « **password** » de la session et puis redirige l'affichage vers la vue « **connexion.jsp** »