Les JSP

Définition

- Les JSP permettent d'introduire du code java dans des tags prédéfinis à l'intérieur d'une page HTML. La technologie JSP mélange la puissance de java côté serveur et la facilité de mise en page d'HTML côté client.

- Une JSP est habituellement constituée :
  - de données et de tags HTML
  - de tags JSP
  - de scriptlets (code java intégré à la JSP)

- Les fichiers JSP possèdent par convention l'extension .jsp.

Définition

- Concrètement, les JSP sont basées sur les servlets. Au premier appel de la page JSP, le moteur de JSP crée et compile automatiquement une servlet qui permet la génération de la page web. Le code HTML est repris intégralement dans la servlet. Le code java est inséré dans la servlet.
- La servlet est sauvegardée puis elle est exécutée. Les appels suivants de la JSP sont beaucoup plus rapides car la servlet, conservée par le serveur, est directement exécutée.
- Il a plusieurs manières de combiner les technologies JSP, les beans/EJB et les servlets.
- Comme le code de la servlet est généré dynamiquement, les JSP sont relativement difficiles à debugguer.

Définition

- Cette approche possède plusieurs avantages :
  - l'utilisation de java par les JSP permet une indépendance de la plate-forme d'exécution mais aussi du serveur web utilisé.
  - la séparation des traitements et de la présentation : la page web peut être écrite par un designer et les tags java peuvent être ajoutés ensuite par le développeur. Les traitements peuvent être réalisés par des composants réutilisables (des java beans).
  - les JSP sont basées sur les servlets : tout ce qui est fait par une servlet pour la génération de pages dynamiques peut être fait avec une JSP.

Les Tags JSP

- Il existe trois types de tags :
  - tags de directives : ils permettent de contrôler la structure de la servlet générée
  - tags de scripting: il permettent d'insérer du code java dans la JSP
  - tags d'actions: ils facilitent l'utilisation de composants
- Attention : Les tags sont sensibles à la casse.

Les Tags JSP

- **Les tags de directives <%@ … %>**
  - Les spécifications des JSP définissent trois directives :
  - **page** : permet de définir des options de configuration
  - **include** : permet d'inclure des fichiers statiques dans la JSP avant la génération de la servlet
  - **taglib** : permet de définir des tags personnalisés
  - Leur syntaxe est la suivante :
- *<%@ directive attribut="valeur" … %>*

Les Tags de directives

La directive page

- Cette directive doit être utilisée dans toutes les pages JSP : elle permet de définir des options qui s'applique à toute la JSP.
- Elle peut être placée n'importe ou dans le source mais il est préférable de la mettre en début de fichier, avant même le tag <HTML>. Elle peut être utilisée plusieurs fois dans une même page mais elle ne doit définir la valeur d'une option qu'une seule fois, sauf pour l'option import.
- Les options définies par cette directive sont de la forme option=valeur.

Les Tags de directives

La directive page

Les options définies par cette directive sont de la forme option=valeur.

Option	Valeur	Valeur par défaut	Autre valeur possible
autoFlush	Une chaine	«true»	«false»
buffer	Une chaine	«8kb»	«none» ou «nnnkb» (nnn indiquant la valeur)
contentType	Une chaîne contenant le type mime		
errorPage	Une chaîne contenant une URL		
extends	Une classe		
import	Une classe ou un package.*		
info	Une chaîne		
isErrorPage	Une chaîne	«false»	«true»
isThreadSafe	Une chaîne	«true»	«false»
langage	Une chaîne	«java»	
session	Une chaîne	«true»	«false»

Exemple :

```
<%@ page import="java.util.*" %>
<%@ page import="java.util.Vector" %>
<%@page info="Ma premiere JSP"%>
```

Les Tags de directives

La directive include

- Cette directive permet d'inclure un fichier dans le source JSP. Le fichier inclus peut être un fragment de code JSP, HTML ou java. Le fichier est inclus dans la JSP avant que celle ci ne soit interprétée par le moteur de JSP.
- Ce tag est particulièrement utile pour insérer un élément commun à plusieurs pages tel qu'un en-tête ou un bas de page.
- Si le fichier inclus est un fichier HTML, celui ci ne doit pas contenir de tag <HTML>, </HTML>, <BODY> ou </BODY> qui ferait double emploi avec ceux présents dans le fichier JSP. Ceci impose d'écrire des fichiers HTML particuliers uniquement pour être inclus dans les JSP : ils ne pourront pas être utilisés seuls.
- La syntaxe est la suivante :
- <%@ include file="chemin relatif du fichier" %>
- Si le chemin commence par un '/', alors le chemin est relatif au contexte de l'application, sinon il est relatif au fichier JSP

Les Tags de directives

- La directive page ♦ Forme générale :
-
- <%@ page
- import="java.io.*;java.sql.*"
- session="true"
- isThreadSafe="true"
- errorPage="bug/erreur.html"
- isErrorPage="false"
- language="java"
- %>
- ♦ il peut y avoir plusieurs directives page dans une JSP :
- <%@ page import="java.io.*" %>
- <%@ page import="java.sql.*" %>
- <%@ page isThreadSafe="true" %>...

Les Tags de directives

- Les directives include et taglib
- ♦ La directive « include » (inclusion à la compilation) :
- <%@ include file="banniere.html" %>
- ♦ La directive « taglib » :
- <%@ taglib uri="monTag" prefix="jlm" %>
- ...
- <jlm:debut> ... </jlm:debut>

Les Tags de Scripting

- Ces tags permettent d'insérer du code java qui sera inclus dans la servlet générée à partir de la JSP.
- Il existe trois tags pour insérer du code java :
  - le tag de déclaration : le code java est inclus dans le corps de la servlet générée. Ce code peut être la déclaration de variables d'instances ou de classes ou la déclaration de méthodes.
  - le tag d'expression : évalue une expression et insère le résultat sous forme de chaîne de caractères dans la page web générée.
  - le tag de scriptlets : par défaut, le code java est inclus dans la méthode service() de la servlet.
- Il est possible d'utiliser dans ces tags plusieurs objets définis par les JSP.

Les Tags de Scripting

- **Le tag de déclarations <%! … %>**
- Ce tag permet de déclarer des variables ou des méthodes qui pourront être utilisées dans la JSP. Il ne génère aucun caractère dans le fichier HTML de sortie.
- La syntaxe est la suivante :
- <%! declarations %>
- Exemple :

```
<%! int i = 0; %>
<%! dateDuJour = new java.util.Date(); %>
```

Les Tags de Scripting

- **Le tag d'expressions <%= … %>**
- Le moteur de JSP remplace ce tag par le résultat de l'évaluation de l'expression présente dans le tag.
- Ce résultat est toujours converti en une chaîne. Ce tag est un raccourci pour éviter de faire appel à la méthode println() lors de l'insertion de données dynamiques dans le fichier HTML.
- La syntaxe est la suivante :
- <%= expression %>
- Le signe '=' doit être collé au signe '%'.
- Attention : il ne faut pas mettre de ' ;' à la fin de l'expression.

Les Tags de Scripting

- Les scriptlets
- Ces élements de code seront insérés dans la méthode _jspService() de la servlet crée.
- Exemple:

```
<html>
<body bgcolor="white">
<h3>Nombre de visiteurs :


<%@ page import = "java.io.*"%>
<%! static    RandomAccessFile fichier; %>
```

Les Tags de Scripting

- Les scriptlets
- Exemple (suite):

```
<%
    long compte = -1;
    String compteur = "page1";
    fichier = new RandomAccessFile("compteurs\\" + compteur,
"rw");
    synchronized(this) {
            compte = fichier.readLong();
            fichier.seek(0);
            fichier.writeLong(++compte);
            fichier.close();
  }
%>
<%= compte %>

</h3>
</body></html>
```

Les Tags de Commentaires

- Il existe deux types de commentaires avec les JSP :
  - les commentaires visibles dans le code HTML
    - **<!-- … -->**
  - les commentaires invisibles dans le code HTML
    - . **Les commentaires cachés <%-- … --%>**

Les tags d'actions

Le tag <jsp:useBean>

- Le tag <jsp:useBean> permet de localiser une instance ou d'instancier un bean pour l'utiliser dans la JSP.
- L'utilisation d'un bean dans une JSP est très pratique car il peut encapsuler des traitements complexes et être réutilisable par d'autre JSP ou composants. Le bean peut notamment assurer l'accès à une base de données. L'utilisation des beans permet de simplifier les traitements inclus dans la JSP.
- Lors de l'instanciation d'un bean, on précise la porté du bean. Si le bean demandé est déjà instancié pour la porté précisée alors il n'y pas de nouvelle instance du bean qui est créée mais sa référence est simplement renvoyée : le tag <jsp:useBean> n'instancie pas obligatoirement un objet.
- Ce tag ne permet pas de traiter des EJB.

Les tags d'actions

- **Le tag <jsp:useBean>**
  - La syntaxe est la suivante :
  - <jsp:useBean
    id="*beanInstanceName*"
    scope="**page**|request|session|application"
    class="*package.class*" | type="*package.class*" </>

- Exemple :
  - <jsp:useBean id="monBean" scope="session" class="test.MonBean"
    />

Les tags d'actions

Le tag <jsp:useBean>

- Dans cet exemple, une instance de MonBean est instancié un seul est unique fois lors de la session. Dans la même session, l'appel du tag <jsp:useBean> avec le même bean et la même portée ne feront que renvoyer l'instance créée. Le bean est accessible durant toute la session.

- **Selon le moteur de JSP utilisé, les fichiers du bean doivent être placé dans un répertoire particulier pour être accessible par la JSP.**

Les tags d'actions

Le tag <jsp:setProperty >

- Le tag <jsp:setProperty> permet de mettre à jour la valeur d'un ou plusieurs attributs d'un Bean. Le tag utilise le setter (méthode getXXX() ou XXX est le nom de la propriété) pour mettre à jour la valeur. Le bean doit exister grâce à un appel au tag <jsp:useBean>.
- Il existe trois façon de mettre à jour les propriétés soit à partir des paramètres de la requête soit avec une valeur :
  - alimenter automatiquement toutes les propriétés avec les paramètres correspondants de la requête
  - alimenter automatiquement une propriété avec le paramètre de la requête correspondant
  - alimenter une propriété avec la valeur précisée

La syntaxe est la suivante :

```
<jsp:setProperty name="beanInstanceName"
   { property="*" |
   property="propertyName" [ param=" parameterName" ] |
   property="propertyName" value="{string | <%= expression%>}"
   }
   />
```

Les tags d'actions

Le tag <jsp:getProperty>

- Le tag <jsp:getProperty> permet d'obtenir la valeur d'un attribut d'un Bean. Le tag utilise le getter (méthode getXXX() ou XXX est le nom de la propriété) pour obtenir la valeur et l'insérer dans la page HTML généré. Le bean doit exister grâce à un appel au tag <jsp:useBean>.
- La syntaxe est la suivante :
- <jsp:getProperty name="*beanInstanceName*" property="*propertyName*" />
- L'attribut name indique le nom du bean tel qu'il a été déclaré dans le tag <jsp:useBean>.
- L'attribut property indique le nom de la propriété dont on veut la valeur.

Les tags d'actions

- **Le tag <jsp:getProperty>** Exemple :
-

```
<html>
  <HEAD>
  <TITLE>Essai d'instanciation d'un bean dans une JSP</TITLE>
  </HEAD>
  <body>
  <p>Test d'utilisation d'un Bean dans une JSP </p>
  <jsp:useBean id="personne" scope="request"
    class="test.Personne" />
  <p>nom initial = <jsp:getProperty name="personne"
    property="nom" /></p>
  <jsp:setProperty name="personne" property="nom" value="mon
    nom" />
  <p>nom mise à jour = <jsp:getProperty name="personne"
    property="nom" /></p>
  </body>
  </html>
```

- Attention : ce tag ne permet pas d'obtenir la valeur d'une propriété indexée ni les valeurs d'un attribut d'un EJB.

Les tags d'actions

Le tag de redirection <jsp:forward>

- Le tag <jsp:forward> permet de rediriger la requête vers une autre URL pointant vers un fichier HTML, JSP ou un servlet.
- Dès que le moteur de JSP rencontre ce tag, il redirige le requête vers l'URL précisée et ignore le reste de la JSP courante. Tout ce qui a été généré par la JSP est perdu.
- La syntaxe est la suivante :
- <jsp:forward page="{*relativeURL* | <%= *expression* %>}" />
  ou
  <jsp:forward page="{*relativeURL* | <%= *expression* %>}" >
  <jsp:param name="*parameterName*" value="{ *parameterValue* | <%= *expression* %>}" /> +
  </jsp:forward>
- L'option page doit contenir la valeur de l'URL de la ressource vers laquelle la requête va être redirigée.
- Cette URL est absolue si elle commence par un '/' ou relative à la JSP sinon. Dans le cas d'une URL absolue, c'est le serveur web qui détermine la localisation de la ressource.

Les tags d'actions

Le tag de redirection <jsp:forward>

- Il est possible de passer un ou plusieurs paramètres vers la ressource appelée grâce au tag <jsp :param>. Exemple :

```
<html>
<body>
<p>Page initiale appelée</p>
<jsp:forward page="forward.htm"/>
</body>
</html>
```

forward.htm

```
<HTML>
<HEAD>
<TITLE>Page HTML</TITLE>
</HEAD>
<BODY>
<p><table border="1" cellpadding="4" cellspacing="0" width="30%"
  align=center >
<tr bgcolor="#A6A5C2">
<td align="center">Page HTML forwardée</Td>
</Tr>
</table></p>
</BODY>
</HTML>
```

- Avec paramètres:

Les tags d'actions

```
<jsp:forward page="banniere.jsp">
  <jsp:param name="titre" value="Premier titre"/>
</jsp:forward>

<p>paragraphe ignoré !</p>
```

Les objets implicites en JSP

- request
- response
- pageContext -> context de pages = échange données

- session
- application
- out
- Config -> servletConfig
- Page -> équivalent de this
- exception

Gestion des erreurs en JSP

- ♦ Une page de gestion des erreurs peut ressembler à ceci :

```
<html>
<%@ page
   language="java"
   isErrorPage="true"%>
<body>
erreur : <%= exception.getMessage() %>
...
<jsp:forward page="point_d'entrée_normal" />
```

- Cette page devra avoir été référencée dans les pages JSP pour lesquelles on veut gérer des erreurs:

```
<%@ page
   errorPage="bug/erreur.html"
   isErrorPage="false" %>
```