JSP/JSTL

Les Java Server Pages

Ecrire une interface HTML dans une string Java n'est pas très pratique. C'est pourquoi il est possible d'utiliser pour ce faire les JSP. Les Java Server Pages sont des templates de vues qui sont rendus coté serveur. Lorsque vous renvoyez une JSP depuis une Servlet, elle va être interprétée et le code HTML résultant sera placé dans le Body de la réponse HTTP. Ces vues sont templatables par du code Java mais c'est une mauvaise pratique car du code Java exprimant de la logique ne devrait pas résider dans une vue. C'est pourquoi il est recommandé d'utilise JSTL, la JSP Standard Tag Library.

Pour créer une JSP qui sera renvoyées par une servlet, créer un fichier `.jsp` à dans le dossier `src/main/webapp/WEB-INF`. Les fichiers de styles et de scripts sont à mettre directement dans le dossier `src/main/webapp` et sont accessibles à partir de l'URL racine du projet.

Il est possible de passer des objets Java à une JSP depuis une servlet :

```
request.setAttribute("nom", monObjet);
```

Pour renvoyer une JSP depuis une servlet, utilisez le code suivant :

```
this.getServletContext().getRequestDispatcher("/WEB-INF/maJsp.jsp").forward(request, response);
```

JSP Standard Tag Library

Installation

Pour installer JSTL, ajouter la dépendance au `pom.xml` de votre projet :

```
<dependency>
    <groupId>org.glassfish.web</groupId>
    <artifactId>jakarta.servlet.jsp.jstl</artifactId>
    <version>2.0.0</version>
```

```
</dependency>
```

Pour importer la JSTL dans une JSP utilisez les balises suivantes :

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
```

Balise de la JSTL

Afficher une variable

La balise `<c:out>` permet d'afficher une variable passée par la Servlet :

```
<c:out value="${ variable }">Valeur par défaut si la variable est null</c:out>
```

Cette balise évalue en fait une expression écrite dans un langage appelé EL pour Expression Language. C'est un langage simple qui permet de faire des petites opérations sur les variables. Il permet d'accèder par leur nom aux variables passés à la JSP par la servlet.

Il permet également d'accèder à un champs d'un objet Java si celui ci possède un Getter correctement nommé en utilisant l'opérateur `.`. EL supporte également les opérateurs arithmétiques et logique ainsi que les expressions ternaires. Le mot clé `empty` permet de vérifier si une valeur est nulle.

Itération sur une liste

La balise `<c:forEach>` permet d'itérer sur les éléments d'une liste :

```
<c:forEach items="${ maListe }" var="monElement" varStatus="status">
   <p>N°<c:out value="${ status.count }" /> : <c:out value="${ titre }" /> !</p>
</c:forEach>
```

Ici la variable `monElement` dansl'attribut `var` de la boucle correspond à l'élément courant de la liste. Quant à la variable `status` (optionnelle) définie dans l'attribut `varStatus` elle permet d'obtenir des informations sur l'élément courant de la boucle et possède des propriétés comme :

- `index` : l'indice de l'élément
- `first` : vrai si l'élément est le premier
- `last` : vrai si l'élément est le dernier

Rendu conditionnel

La balise `<c:if>` permet de faire du rendu conditionnel :

```
<c:if test="${ variable == '1' }">
    C'est vrai !
</c:if>
```

La contenu de la balise n'est rendu que si l'expression EL dans l'attribut `test` est évaluée à `true`.

Rendu conditionnel à choix multiple

La balise `<c:choose>` permet de faire du rendu conditionnel à choix multiples :

```
<c:choose>
    <c:when test="${ variable = '1' }">C'est égal à 1</c:when>
    <c:when test="${ variable = '2' }">C'est égal à 2</c:when>
    <c:when test="${ variable = '3' }">C'est égal à 3</c:when>
    <c:otherwise></c:otherwise>
</c:choose>
```

Revision #7
Created 21 November 2020 23:18:21 by Arsène Lapostolet
Updated 21 February 2022 09:05:29 by Arsène Lapostolet