Java server pages Standard Tag Library

JSTL

- JSTL : Java server page Standard Tag Library.

  http://java.sun.com/products/jsp/jstl/

- Ensemble standard d'actions personnalisées (Custom Tags) développé par la JSR (Java Specification Request) 052.
- Un ensemble de marqueurs standard permettant d'éviter le mélange du **code Java** et des **marqueurs XHTML**

- Les actions possibles :

1. Affectation d'une valeur (objet).
2. Capture des exceptions.
3. Conditions et des Itérateurs.
4. Accès aux bases de données.

- Une bibliothèque de marqueurs est une collection de fonctions pouvant être utilisées dans une page JSP ou JSF.

Bibliothèques de marqueurs JSTL

Domaine	*URI*	*Préfixe classique*
Noyau	http://java.sun.com/jsp/jstl/core	c
Traitement XML	http://java.sun.com/jsp/jstl/xml	x
I18N et formatage	http://java.sun.com/jsp/jstl/fmt	fmt
Accès aux BD	http://java.sun.com/jsp/jstl/sql	sql
Fonctions	http://java.sun.com/jsp/jstl/functions	fn

- la JSP doit importer l'URI de la bibliothèque et choisir un préfixe.

- *Deux Possibilité : soit on utilise une directive JSP avec le système de marqueurs de JSP, soit on utilise une syntaxe XML*

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
// ou
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
          xmlns:c="http://java.sun.com/jstl/core" version="1.2">
```

- On pourra ensuite utiliser ttes les actions de la bibliothèque des marqueurs fondamentaux en utilisant le préfixe c :

```
<c:set var="upperLimit" value="20"/>
```

Actions Fondamentales (Noyau)

- fournissent des marqueurs permettant de :

1. Manipuler des variables
2. Traiter les erreurs
3. Effectuer des tests
4. Exécuter des boucles et des itérations.

- Voici ces actions :

Action	*Description*
`<c:out>`	Évalue une expression et affiche son résultat.
`<c:set>`	Initalise la valeur d'un objet.
`<c:remove>`	Supprime une variable.
`<c:catch>`	Capture une exception `java.lang.Throwable` lancée par l'une de ses actions imbriquées.
`<c:if>`	Teste si une expression est vraie.
`<c:choose>`	Fournit plusieurs alternatives exclusives.
`<c:when>`	Représente une alternative dans une action `<c:choose>`.
`<c:otherwise>`	Représente la dernière alternative d'une action `<c:choose>`.
`<c:forEach>`	Répète son corps pour chaque élément d'une collection ou un nombre fixé de fois.
`<c:forTokens>`	Itère sur une liste de tokens (lexèmes) séparés par des virgules.
`<c:import>`	Importe une ressource.
`<c:url>`	Encode une URL.
`<c:param>`	Ajoute des paramètres de requête à une URL.
`<c:redirect>`	Redirige vers une URL précise.

Exemple1 :

```
<%@ taglib uri="http://java.sun.com/jstl/core" prefix="c" %>
<html>
  <head>
    <title>Count to 10 Example (using JSTL)</title>
  </head>
  <body>
    <c:forEach var="i" begin="1" end="10" step="1">
    <c:out value="${i}" />
    <br />
    </c:forEach>
  </body>
</html>
```

Exemple1/2 :

Dans la Servlet

```
ArrayList<Etudiant> list = new ArrayList<Etudiant>();
request.setAttribute("key", list);
```

Dans la jsp

```
<body>
<c:forEach items ="${key}" var="obit">
nom :<c:out value="${obit.nom}"></c:out><br>
prenom <c:out value="${obit.prenom}"></c:out><br>
<br><br>
</c:forEach>
```

Exemple 2:

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
          xmlns:c="http://java.sun.com/jstl/core" version="1.2">
<html>
  <body>
    <c:set var="upperLimit" value="20"/>
    <c:forEach var="i" begin="3" end="${upperLimit - 5}">

      <c:choose>
        <c:when test="${i%2 == 0}">
          <c:out value="${i} is even"/><br/>
        </c:when>
        <c:otherwise>
          <c:out value="${i} is odd"/><br/>
        </c:otherwise>
      </c:choose>

    </c:forEach>
  </body>
</html>
```

Remarque :

- Traitement s'effectue grâce aux marqueurs, que l'exemple est conforme à XML.
- Doit être compris par les développeurs qui ne connaissent pas Java.

Actions Formatages

- fournissent des marqueurs :

  1. formater des dates, des nombres, des valeurs monétaires et des pourcentages.

  2. Obtenir ou modifier les locales (variables de langue) et les zones horaires (i18n)

  3. Obtenir l'encodage de la page web.

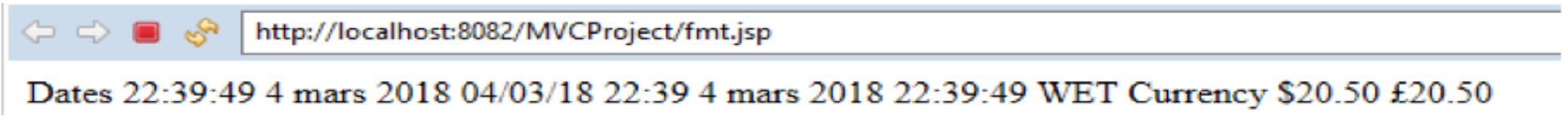
- Voici ces actions :

Action	*Description*
`<fmt:message>`	Internationalise une JSP en extrayant un message en fonction de la langue.
`<fmt:param>`	Fournit un paramètre à `<fmt:message>`.
`<fmt:bundle>`	Indique le paquetage contenant les messages par langue.
`<fmt:setLocale>`	Fixe la langue à utiliser.
`<fmt:requestEncoding>`	Fixe l'encodage des caractères de la requête.
`<fmt:timeZone>`	Précise la zone horaire du format de l'heure.
`<fmt:setTimeZone>`	Stocke la zone horaire indiquée dans une variable.
`<fmt:formatNumber>`	Formate une valeur numérique (nombre, monnaie, pourcentage) selon la locale.
`<fmt:parseNumber>`	Analyse la représentation textuelle des nombres, des valeurs monétaires et des pourcentages.
`<fmt:formatDate>`	Formate les dates et les heures selon la langue.
`<fmt:parseDate>`	Analyse la représentation textuelle des dates et des heures.

Exemple 3:

```
<%@pagelanguage="java"contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@tagliburi="http://java.sun.com/jstl/core_rt"prefix="c"%>
<%@tagliburi="http://java.sun.com/jstl/fmt_rt"prefix="fmt"%>
<!DOCTYPEhtmlPUBLIC"-//W3C//DTD HTML 4.01
Transitional//EN""http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
        Dates
        <c:set  var="now"  value="<%=newjava.util.Date() %>"/>
        <fmt:formatDate    type="time"    value="${now}"/>
        <fmt:formatDate    type="date"    value="${now}"/>
        <fmt:formatDate    type="both"    dateStyle="short"  timeStyle="short"
                           value="${now}"/>
        <fmt:formatDate     type="both"  dateStyle="long"  timeStyle="long"
                                         value="${now}"/>
        Currency
        <fmt:setLocalevalue="en_us"/>
        <fmt:formatNumbervalue="20.50"type="currency"/>
        <fmt:setLocalevalue="en_gb"/>
        <fmt:formatNumbervalue="20.50"type="currency"/>

</body>
</html>
```



Actions SQL

- Les actions SQL permettent d'effectuer :

1. Requêtes sur une base de données (insertions, modifications et suppressions).
2. Accéder aux résultats de ces requêtes.
3. mettre en place un contexte transactionnel.

- On a parfois besoin d'accéder à une base à partir d'une page web (exemple : application web d'administration non critique utilisée occasionnellement par un unique utilisateur).

Action	*Description*
`<sql:query>`	Interroge une base de données.
`<sql:update>`	Exécute une instruction SQL `INSERT`, `UPDATE` ou `DELETE`.
`<sql:transaction>`	Établit un contexte transactionnel pour les marqueurs `<sql:query>` et `<sql:update>`.
`<sql:setDataSource>`	Indique la source de données.
`<sql:param>`	Fixe les valeurs des marqueurs d'emplacements `(?)` d'une instruction SQL.
`<sql:dateParam>`	Fixe les valeurs des marqueurs d'emplacements `(?)` d'une instruction SQL pour les valeurs de type `java.util.Date`.

Exemple 4 :

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<html>
  <head>
    <title>Lists all the books</title>
  </head>
  <body>
    <h1>Lists all the books</h1>
    <hr/>
    <sql:setDataSource dataSource="jdbc/__default"/>
    <sql:query var="books">
      select * from book
    </sql:query>
```

Exemple 5 :

```
        <table border="1">
          <tr>
            <th>ISBN</th>
            <th>Title</th>
            <th>Price</th>
            <th>Description</th>
            <th>Number of pages</th>
            <th>Illustrations</th>
          </tr>
          <c:forEach var="row" items="${books.rows}">
            <tr>
              <td><c:out value="${row.isbn}"/></td>
              <td><c:out value="${row.title}"/></td>
              <td><c:out value="${row.price}"/></td>
              <td><c:out value="${row.description}"/></td>
              <td><c:out value="${row.nbOfPage}"/></td>
              <td><c:out value="${row.illustrations}"/></td>
            </tr>
          </c:forEach>
        </table>
        <hr/>
    </body>
</html>
```

Actions XML

- On pourra dire que ca ressemble aux marqueurs fondamentaux, elles permettent :

1. Effectuer une analyse XML
2. Itérer sur les éléments des collections.
3. Effectuer des opérations reposant sur les expressions XPath.
4. Effectuer des transformations à l'aide de documents XSL.

- Voici les actions de cette bibliothèque.

Action	*Description*
`<x:parse>`	Analyse un document XML.
`<x:out>`	Évalue une expression XPATH et produit son résultat.
`<x:set>`	Évalue une expression XPATH et stocke son résultat dans une variable.
`<x:if>`	Évalue l'expression XPATH si l'expression est vraie.
`<x:choose>`	Fournit plusieurs alternatives exclusives.
`<x:when>`	Représente une alternative d'une action `<x:choose>`.
`<x:otherwise>`	Représente la dernière alternative d'une action `<x:choose>`.
`<x:forEach>`	Évalue une expression XPATH et répète son contenu sur le résultat.
`<x:transform>`	Applique une feuille de style XSLT à un document XML.
`<x:param>`	Fixe les paramètres de la transformation `<x:transform>`.

Exemple 6:

```
<?xml version='1.0' encoding='UTF-8'?>
<books>
  <book isbn='1234-234' price='12' nbOfPage='241'
        illustrations='true'>
    <title>H2G2</title>
    <description>Scifi IT book</description>
  </book>
  <book isbn='564-694' price='18.5' nbOfPage='317'
        illustrations='true'>
    <title>Robots</title>
    <description>Best seller</description>
  </book>
  <book isbn='256-6-56' price='23.25' nbOfPage='529'
        illustrations='false'>
    <title>Dune</title>
    <description>The trilogy</description>
  </book>
</books>
```

Exemple 7:

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x" %>
<html>
  <body>
    <table border="1">
      <tr>
        <th>ISBN</th>
        <th>Title</th>
        <th>Price</th>
        <th>Description</th>
        <th>Number of pages</th>
        <th>Illustrations</th>
      </tr>

      <c:import url="books.xml" var="bookUrl"/>
      <x:parse xml="${bookUrl}" var="doc"/>

      <x:forEach var="b" select="$doc/books/book">
        <tr>
          <td><x:out select="$b/@isbn"/></td>
          <td><x:out select="$b/title"/></td>
           <td><x:out select="$b/@price"/></td>
           <td><x:out select="$b/description"/></td>
           <td><x:out select="$b/@nbOfPage"/></td>
           <td><x:out select="$b/@illustrations"/></td>
         </tr>
       </x:forEach>
     </table>
   </body>
  </html>
```

Fonctions

- Les fonctions ne sont pas des marqueurs mais sont quand même définies dans la spécification JSTL.
- peuvent être utilisées avec EL et sont principalement employées pour traiter les chaînes de caractères.
- Exemple :

```
${fn:contains("H2G2", "H2")}
```

```
${fn:length("H2G2")}
```

Fonction	*Description*
`fn:contains`	Teste si une chaîne contient la sous-chaîne indiquée.
`fn:containsIgnoreCase`	Idem, sans tenir compte de la casse.
`fn:endsWith`	Teste si une chaîne se termine par le suffixe indiqué.
`fn:escapeXml`	Protège les caractères pouvant être interprétés comme du XML.
`fn:indexOf`	Renvoie l'indice de la première occurrence d'une sous-chaîne dans une chaîne.
`fn:join`	Joint tous les éléments d'un tableau pour former une chaîne.
`fn:length`	Renvoie le nombre d'éléments d'une collection ou le nombre de caractères d'une chaîne.
`fn:replace`	Renvoie une chaîne où toutes les occurrences de la sous-chaîne indiquée ont été remplacées par une autre chaîne.
`fn:split`	Découpe une chaîne pour obtenir un tableau de sous-chaînes.
`fn:startsWith`	Teste si une chaîne commence par le préfixe indiqué.
`fn:substring`	Renvoie une sous-chaîne.
`fn:substringAfter`	Renvoie la sous-chaîne située après la sous-chaîne indiquée.
`fn:substringBefore`	Renvoie la sous-chaîne située avant la sous-chaîne indiquée.
`fn:toLowerCase`	Convertit une chaîne en minuscules.
`fn:toUpperCase`	Convertit une chaîne en majuscules.
`fn:trim`	Supprime les espaces aux deux extrémités d'une chaîne.

Exemple :

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fn" prefix="fn" %>
<html>
  <body>
    <c:out value="${fn:toLowerCase(sentence)}" />
    <c:if test="${fn:length('H2G2') == 4}">
      H2G2 is four caracters long
    </c:if>
  </body>
</html>
```