Syntaxe JSP

B. Mermet

Préambule

- 2 syntaxes possibles
  - Syntaxe "JSP"
  - Syntaxe "XML"
- Choix fait : on ne présente que la syntaxe JSP

Syntaxe de base

- Scriptlet (code de base)
  - Syntaxe : <% ... %> ou <jsp:scriptlet>...</jsp:scriptlet>
  - Utilisation : code java inséré dans la méthode service()
- Commentaires
  - Syntaxe : <%-- ... --%>
  - Utilisation : commentaire de code non renvoyé
- Déclaration
  - Syntaxe : <%! ... %>
  - Utilisation : tout code or de la méthode service()
- Expression
  - Syntaxe : <%= ... %> ou <jsp:expression>...</jsp:expression>
  - Utilisation : affichage de la valeur d'une expression java

Directive page

- Syntaxe
  - Syntaxe : <%@ page directive="valeur" %>
- Directives possibles
  - language="**java**" : une seule valeur possible pour le moment
  - extends="package.class" : classe mère de la jsp ; usage à limiter
  - import="package" ou import ="listepackage"
    Par défaut : java.lang.*, javax.servlet.*, javax.servlet.jsp.*, javax.servlet.http.*
  - session="**true**|false" : si une session est requise pour utiliser la page
  - buffer="none|**8kb**|sizekb"
  - autoFlush="**true**|false"
  - isThreadSafe="**true**|false" : si false, SingleThreadModel utilisé
  - info="text"
  - errorPage="relativeURL" : url de la page d'erreur vers laquelle aller
  - contentType="mimeType [; charset=characterSet]"
    Par défault : "text/html ; charset=ISO-8859-1"]
  - isErrorPage="true|**false**" : s'agit-il d'une page d'erreur (objet exception)
  - pageEncoding="characterSet | **ISO-8859-1**"
  - isELIgnored="true|false" : valeur par défaut dépend version

Quelques autres directives

- Inclusion statique
  - Syntaxe :
    - <%@ include file="URLrelative" %>
    - <jsp:directive.include file="URLrelative" />
  - Principe
    - Le fichier est inclus avant la compilation
    - Fichier peut être du html, du texte, une jsp, du source java
- Utilisation de bibliothèques de balises
  - Syntaxe
    - <%@ taglib {uri="URI" | tagdir="/WEB-INF/tags[/subdir]+"} prefix="tagPrefix" %>
  - Principe
    - Voir section sur les balises personnalisées

Syntaxe du Langage d'Expression JSP

- ${Expression}
- Expression = {(ChoiceExpression | (Expression BinaryOp Expression)
  (UnaryOp Expression) | Value }
- ChoiceExpression = Expression ? Expression : Expression
- BinaryOp (JSP syntax) = and | && | or | || | '+' | '-' | * | / | did
  | % | mod | > | gt | < | lt | >= | ge | <= |
  le | == | eq | != | ne
- BinaryOp (XML syntax) = and | && | or | || | '+' | '-' | * | / | div
  | % | mod | gt | lt | ge | le | == | eq | ne
- UnaryOp = {'-' | ! | not | empty }
- Value = {ValuePrefix | (Value ValueSuffix) }
- ValuePrefix = {Literal | '('Expression')' | ImplicitObject |
  Java language identifier | FuncInvocation }
- ValueSuffix = { . Identifier | '['Expression']'}
- FuncInvocation = [Identifier :]Identifier '('[Expression [','
  Expression]+]')'
- Literal = booléen | nombre | chaine | null

EL (suite)

- Utilisation
  - Dans le texte
  - Dans les paramètres des balises
- Intérêt
  - Faciliter notamment l'accès aux propriétés des beans (voir partie sur les beans)
- Utilisation d'une variable dans EL
  - Recherche de l'attribut dans page, requête, session, application
- Notations "." Et [] : synonymes : a.b <=> a[b]
  - Utilisations : tableaux, "maps" et beans
- Quelques objets implicites :
  - pageContext.[servletContext|session|request|response]
  - Param, paramValues, header, headerValues, cookie, initParam
- Pour aller plus loin
  - http://java.sun.com/products/jsp/syntax/2.0/syntaxref207.html#1010522

Utilisation des beans

- Syntaxe simpifiée
  - <jsp:useBean id="*nomInstance*" scope="page|request|session|application" class="paquetage.class" />
  - <jsp:useBean id="*nomInstance*" scope="page|request|session|application" class="paquetage.class"> ... </jsp:useBean>
- Principe
  - Si le bean existe, sa référence est récupérée dans la variable nomInstance.
  - Sinon, il est créé, sa réf. est stockée dans la variable *nomInstance*, et les éventuelles actions incluses dans la balise (syntaxe 2) sont exécutées.
- Actions éventuelles (syntaxe 2)
  - Des "setProperty"

Manipulation des beans

- Balise setProperty

  Syntaxe :

  - <jsp:setProperty name="*nomInstance*" property="*" />
  - <jsp:setProperty name="*nomInstance*" property="nomProp" param="nomParam" />
  - <jsp:setProperty name="*nomInstance*" property="nomProp" value="{chaine | '${' Expression '}' | <%= expression %>} />
- Balise getProperty

  Syntaxe :

  <jsp:getProperty name="*nomInstance*" property="*nomProp*" />
- Autres utilisations
  - Dans le code java via une variable *nomInstance*
  - Dans les expressions EL

Jeu de balise JSTL core (1)

- Syntaxe standard

  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
- Balise URL
  - <c:url var="*nomVar*" scope="*portée*" value="*url*" context="contexte/>
  - <c:url var="*nomVar*" scope="*portée*" value="*url*" context="contexte>

    <c:param name="nomParam" value="valeur"/>

    ...

    </c:url>
- Représentation des URL :
  - Absolues : *protocole://adresse*
  - Relatives à l'application : */adresse*
  - Relative à la page : *adresse*

Jeu de balise JSTL core (2)

- Balise forEach
  - Itération sur :
    - Un tableau
    - Une collection
    - Une énumération
    - Un itérateur
  - Syntaxe simplifiée
    - <c:forEach var="variable" items="liste">...</c:forEach>
    - Paramètres supplémentaires
      - begin : indice du premier élément parcouru
      - end : indice du dernier élément parcouru
      - step : intervalle entre 2 éléments
    - Rmq : utiliser begin et end sans items permet une boucle for

Jeu de balise JSTL core (3)

- Balise forTokens
  - Découpage d'une chaîne de caractères
  - Syntaxe simplifiée
    - <c:forTokens var="mot" items="chaine" delims="sep">

      ...

      </c:forTokens>
    - Paramètres supplémentaires
      - begin : indice du premier mot parcouru
      - end : indice du dernier mot parcouru
      - step : intervalle entre 2 mots considérés

Jeu de balise JSTL core (4)

- Balise if
  - Traitement conditionnel
  - Syntaxe simplifiée
    - <c:if test="*LeTest*" var="*resultat*" scope="*portée*">

      ...

      </c:if>
  - Remarques :
    - Seul le paramètre "test" est obligatoire
    - La portée par défaut est "page"

Jeu de balise JSTL core (5)

- Balise choose
  - Equivalent du "case" de Java
  - Syntaxe simplifiée
    - <c:choose>

      <c:when test="leTest">...</c:when>

      <c:when test="leTest">...</c:when>

      ...

      <c:otherwise>...</c:otherwise>

      </c:choose>
  - Remarque :
    - Les traitements des branches ne sont pas enchaînés comme avec le case (break automatique)

Jeu de balise JSTL core (6)

- Balise catch
  - Ignorer/stocker les exceptions générées dans le bloc encadré
  - Syntaxe simplifiée
    - <c:catch var="*variable*">

      ...

      </c:catch>
- Balise remove
  - Supprime une variable
  - Syntaxe :
    - <c:remove var = "*variable*" scope="*portée*"/>

Jeu de balise JSTL core (7)

- Balise set
  - Stocke le résultat d'un calcul dans une variable ou dans une propriété d'un bean
  - Syntaxe simplifiée :
    - <c:set var="*variable*" value="*valeur*" scope="*portée*"/>
    - <c:set target="*bean*" property="*prop*" value="*valeur*"/>
- Balise redirect
  - Envoie un ordre de re-direction au client avec ou sans paramètre
  - Syntaxe : <c:redirect url="cible"/> ou <c: redirect url="cible">... </c:redirect>

Jeu de balise JSTL core (8)

- Balise import
  - Récupération du contenu d'une autre page (pouvant être externe à l'application)
  - Syntaxes possibles
    - <c:import url="*adresse*"/>
    - <c:import url="*page*" context="*appli*"/>
    - <c:import url="*adresse*"> *params* </c:import>
    - <c:import url="*adresse*" var="*variable*"/>
    - <c:import url="*adresse*" varReader="*nomFlux*">*lecture*</c:import>
  - Paramètres supplémentaires
    - scope
    - charEncoding
  - Rmq : si varReader, l'url doit contenir dès le départ ses paramètres

Autres packages de jstl

- fmt (formatage, préfixe usuel : fmt)
  - Gère l'internationalisation (langue, fuseau horaire)
  - Interpréation des chaînes pour dates, nombres,
  - Affichage formaté d'une date, d'un nombre
- sql (SQL !, préfixe usuel : sql)
  - Définition de la connexion à la base
  - Exécution de requêtes (interrogation et modification) éventuellement paramétrées et de transactions
- xml (XML !, préfixe usuel : x)
  - Lecture de documents XML
- functions (fonctions EL, préfix usuel : fn)
  - Rajouts de fonctions sur les chaînes de caractères à EL