Université Constantine 2 Abdelhamid Mehri

Faculté des Nouvelles Technologies

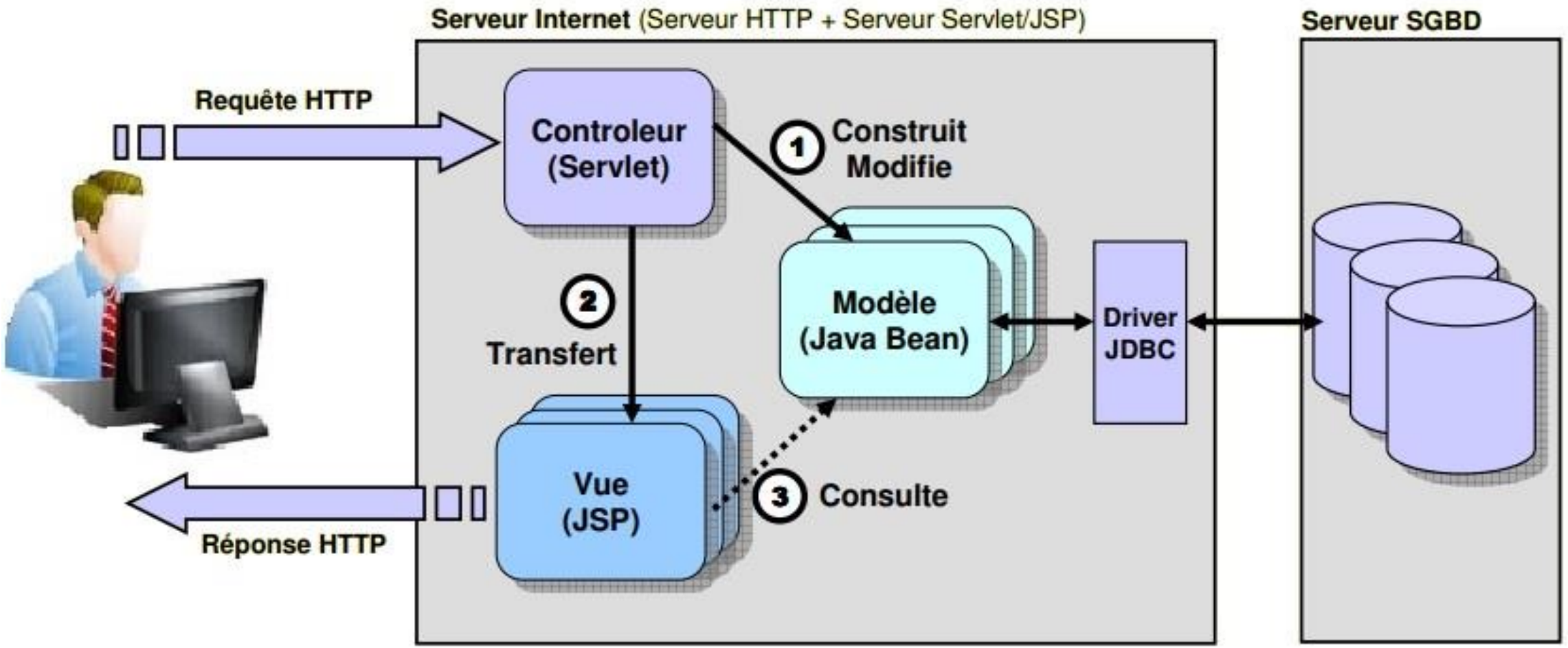
Département Technologies des Logiciels et Systèmes d'Information

JAVA SERVER PACE
JSP

DR. zakaria LAKHDARA / zakaria.lakhdara@univ-constantine2.dz

Faculté	Département	Niveau	Spécialité
NTIC	TLSI	L3	GL

VUE : JSP (JAVA SERVER PAGE)



JSP

- Une JSP est un fichier contenant un mélange de **code HTML** (statique) et des **fragments de code Java**:
  - Les parties statiques de la page web sont écrites en HTML
  - Les parties dynamiques de la page web sont écrites en Java
- Une **JSP est exécutées** sur le moteur de Servlets(Tomcat par exemple) pour **générer les page HTML** à envoyer au client.

POURQUOI UTILISER LES JSP ?

- Le contrôle (Servlet) génère la réponse à envoyer au client (La page HTML).
- Le contrôle combine des données (objet java) avec du code HTML statique pour créer les pages web.

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    response.setContentType("text/html") ;

    PrintWriter out = response.getWriter() ;
    out.println("<html>") ;
    out.println("<head>") ;
    out.println("<title>Bonjour le monde !</title>") ;
    out.println("</head>") ;
    out.println("<body>") ;
    out.println("<h1>Bonjour le monde !</h1>") ;
    out.println("</body>") ;
    out.println("</html>") ;
}
```

POURQUOI UTILISER LES JSP ?

- Le contrôle (Servlet) génère la réponse à envoyer au client.

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    response.setContentType("text/html") ;

    PrintWriter out = response.getWriter() ;
    out.println("<html>") ;
    out.println("<head>") ;
    out.println("<title>Bonjour le monde !</title>") ;
    out.println("</head>") ;
    out.println("<body>") ;
    out.println("<h1>Bonjour le monde !</h1>") ;
    out.println("</body>") ;
    out.println("</html>") ;
}
```

Exécution

```
<html>
<head>
    <title>Bonjour tout le monde !</title>
</head>
<body>
    <h1>Bonjour tout le monde !</h1>
</body>
</html>
```

- Est-ce que c'est pratique ?
- Dans le cas d'une page web complexe ?
- Est- ce que cela respecte le modèle MVC ? Ou le contrôle et la vue doivent être séparés ?

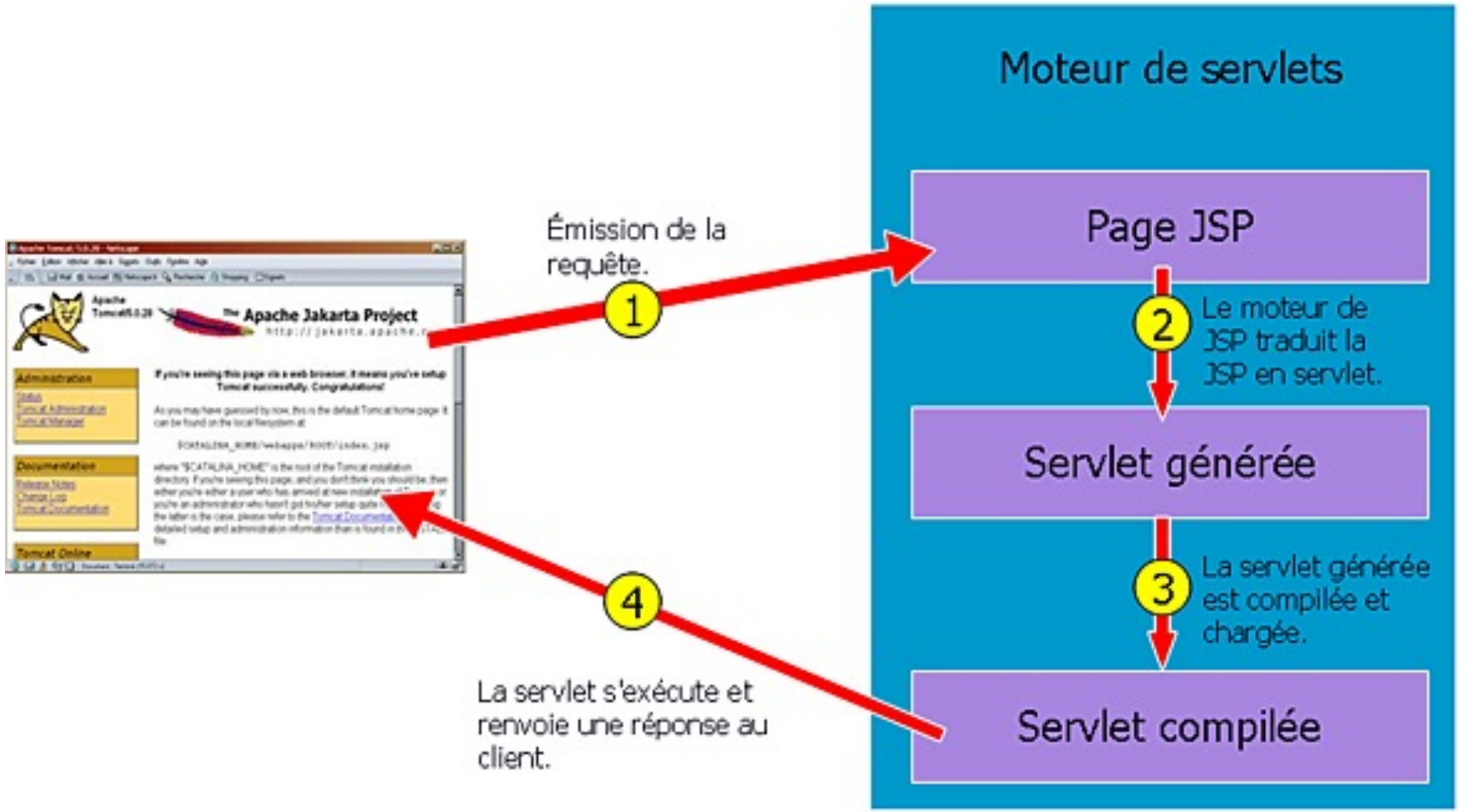
NON

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8">
    <title>My JEE webApp</title>
   <style><%@include file="/WEB-INF/style.css"%></style>

</head>

<body>
    <h1>Contact us</h1>
      <form name="myForm" action="formServlet"  method="post">
         <table class="form-style">
            <tr>
               <td>
                  <label>
                     Your name <span class="required">*</span>
                  </label>
               </td>
               <td>
                  <input type="text" name="name" class="long"/>
                  <span class="error" id="errorname"></span>
               </td>
            </tr>
            <tr>
               <td>
                  <label>
                     Your e-mail adress <span class="required">*</span>
                  </label>
               </td>
               <td>
                  <input type="text" name="email" class="long"/>
                  <span class="error" id="erroremail"></span>
               </td>
            </tr>
            <tr>
               <td>
                  <label>
                     Message <span class="required">*</span>
                  </label>
               </td>
               <td>
                  <textarea name="message" class="long field-textarea"></textarea>
                  <span class="error" id="errormsg"></span>
               </td>
            </tr>
            <tr>
```

JSP : PRINCIPE DE FONCTIONNEMENT

- Au lieu de mettre du HTML dans du code JAVA pour générer les pages web, on met des fragments de code JAVA (pour récupérer les données) dans le code HTML.
- Nous obtenons des JSP (Java Server Page).
- Le serveur exécute la JSP pour générer la page HTML à envoyer au client.
- Pour exécuter une JSP il faut:
  - Traduire la JSP en Servlet
  - Exécuter la Servlet résultante pour générer la page HTML

JSP : PRINCIPE DE FONCTIONNEMENT



JSP

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>Bonjour tout le monde !</title>
</head>
<body>
    <h1>Bonjour tout le monde !</h1>
</body>
</html>
```

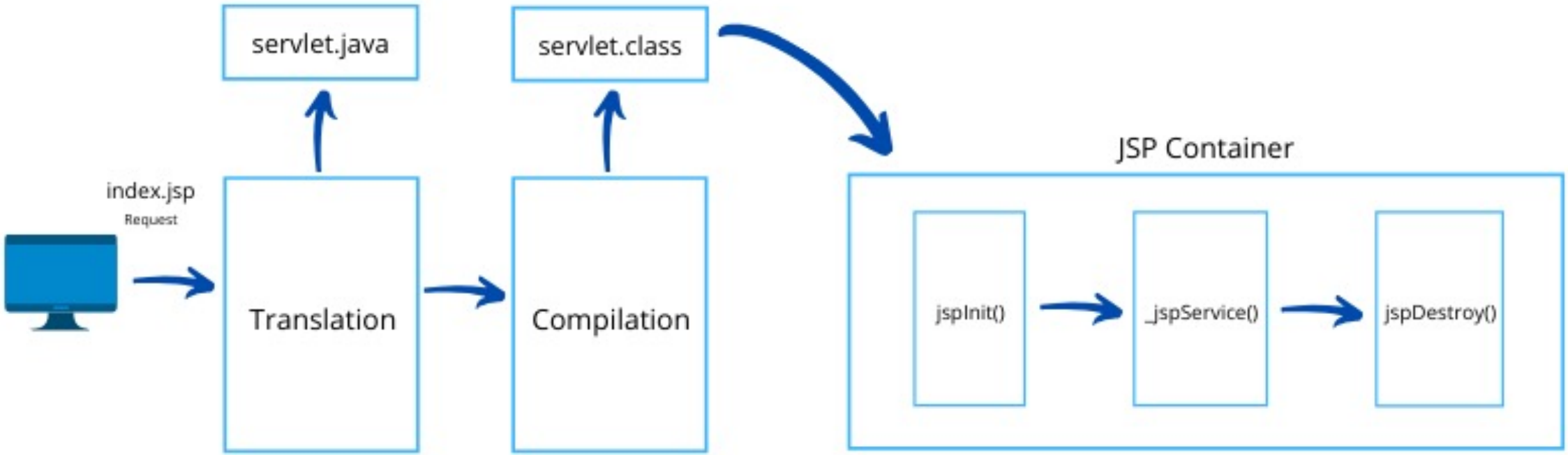
Traduir JSP en Servlet

```java
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    response.setContentType("text/html") ;

    PrintWriter out = response.getWriter() ;
    out.println("<html>") ;
    out.println("<head>") ;
    out.println("<title>Bonjour le monde !</title>") ;
    out.println("</head>") ;
    out.println("<body>") ;
    out.println("<h1>Bonjour le monde !</h1>") ;
    out.println("</body>") ;
    out.println("</html>") ;
}
```

Exécution

```html
<html>
<head>
    <title>Bonjour tout le monde !</title>
</head>
<body>
    <h1>Bonjour tout le monde !</h1>
</body>
</html>
```

JSP : CYCLE DE VIE



VERSION SERVLET

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

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println(" <title> Servlet vs JSP </title>");
        out.println(" </head>");
        out.println(" <body>");
        out.println(" <h1>Bonjour tout le monde</h1>");
        out.println(" Nous sommes le " + (new java.util.Date().toString()));
        out.println(" </body>");
        out.println("</html>");
    }

}
```

VERSION JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<html>
<head>
    <title> Servlet vs JSP </title>
</head>
<body>
    <h1>Bonjour tout le monde</h1>
    Nous sommes le <%= new java.util.Date().toString() %>
</body>
</html>
```

LES BALISE TAG

JSP : BALISE TAG

JSP : BALISE TAG

- Les Tags sont des balises qui permettent de différencier le code HTML du code Java dans une JSP.
- Il existe trois types de tags :
  - **Tags de directives :** ils permettent de contrôler la structure de la servlet générée à partir de la JSP
  - **Tags de scripting:** ils permettent d'insérer du code Java dans la servlet
  - **Tags d'actions:** ils facilitent l'utilisation de composants(Java bean).

JSP : TAGS DE DIRECTIVES

- Les directives permettent de préciser des informations globales sur la page JSP.
- **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" … %>

JSP : TAGS DE DIRECTIVES

Le tag Page : : permet de définir des options de configuration

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
 pageEncoding="UTF-8"%>
```

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

JSP : TAGS DE DIRECTIVES

- **Le tag include** : permet d'inclure des fichiers statiques dans la JSP avant la génération de la servlet

Inclure une feuille de style

```
<style>
<%@include file="/WEB-INF/cssfiles/style.css"%>
</style>
```

Inclure une autre JSP

```
<body>
 <%@include file="/WEB-INF/header.jsp"%>
 <h2>Welcome this is Home page ....</h2>
</body>
```

JSP : TAGS DE DIRECTIVES

- **Le tag taglib** : permet de définir des tags personnalisés. Par exemple, utiliser la bibliothèque taglibs **JSTL (JSP Standard Tag Library)**.

<%@ **taglib** uri=*"http://java.sun.com/jsp/jstl/core"* prefix="c"%>

JSP : TAGS DE SCRIPTING

Tags de scripting: permettent d'insérer du code Java qui sera inclus dans la servlet générée à partir de la JSP.

- **tag de déclaration** : pour la déclaration de variables (<%! %>)
- **tag d'expression** : évalue une expression et insère le résultat sous forme de chaîne de caractères dans la page web générée <%= %>
- **tag de scriptlets** : blocs d'instructions Java inclus dans la page JSP entre des délimiteurs <% et %>

JSP : TAGS DE SCRIPTING

- **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.

```
<%! int i = 0; %>
```

```
<%! date = new java.util.Date(); %>
```

JSP : TAGS DE EXPRESSION

- **tag d'expression** : évalue une expression et insère le résultat sous forme de chaîne de caractères dans la page web générée.

```
<p> Date : <% = new java.util.Date() %> </p>
```

JSP : TAGS DE SCRIPTING

- **tag de scriptlets** : blocs d'instructions Java inclus dans la page JSP entre des délimiteurs <% et %>

```
<%@ page import="java.util.Date"%>
 <html>
 <body>
        <%!  Date date; %>
        <% date = new Date(); %>
        Date du jour : <%= date %>
 </body>
</html>
```

JSP : TAGS DE SCRIPTING

<%! int count =0; %> Déclaration

```
<HTML>
<BODY>
    Bonjour. <BR>
    <%
        count ++;
        if (count == 1)
    %>
        Vous êtes le premier visiteur.
    <%  else  %>
        Vous êtes le <%= count %> eme visiteur.
</BODY>
</HTML>
```

JSTL : JSP STANDARD TAG LIBRARY

- C'est un ensemble de **tags personnalisés** qui propose des fonctionnalités souvent rencontrées dans les JSP :
  - Tag de structure (itération, conditionnement …)
  - Internationalisation
  - Exécution de requêtes SQL
  - Utilisation de documents XML

JSTL : JSP STANDARD TAG LIBRARY

- Pour utiliser cette bibliothèque, il faut la déclarer dans le fichier web.xml du répertoire WEB-INF de l'application web.

```
<taglib>
        <taglib-uri> http://java.sun.com/jstl/core </taglib-uri>
        <taglib-location> /WEB-INF/tld/c.tld </taglib-location>
 </taglib>
```

- Pour chaque JSP qui utilise un ou plusieurs tags, la bibliothèque doit être déclarée avec une directive taglib

```
<%@ taglib uri="http://java.sun.com/jstl/core" prefix="c" %>
```

JSTL : JSP STANDARD TAG LIBRARY

Tag SET:

Le tag set permet de stocker une variable dans une portée particulière (page, requête, session ou application).

```
<c:set var="maVariable1" value="1" scope="page" />

<c:set var="maVariable2" value="2" scope="request" />

<c:set var="maVariable3" value="3" scope="session" />

<c:set var="maVariable4" value="4" scope="application" />
```

JSTL : JSP STANDARD TAG LIBRARY

Tag OUT:

Le tag out permet d'envoyer dans le flux de sortie de la JSP le résultat de l'évaluation de l'expression fournie dans le paramètre " value ". Ce tag est équivalent au tag d'expression <%= ... %> de JSP.

```
<c:out value='${pageScope.maVariable1}' />
<c:out value='${requestScope.maVariable2}' />
<c:out value='${sessionScope.maVariable 3}' />
<c:out value='${applicationScope.maVariable 4}' />
```

JSTL : JSP STANDARD TAG LIBRARY

Tag REMOVE:

Le tag remove permet de supprimer une variable d'une portée particulière.

```
<c:remove var="maVariable1" scope="page" />

<c:remove var="maVariable2" scope="request" />

 <c:remove var="maVariable3" scope="session" />

<c:remove var="maVariable4" scope="application" />
```

JSTL : JSP STANDARD TAG LIBRARY

Tag IF:

Ce tag permet d'évaluer le contenu de son corps si la condition qui lui est fournie est vraie.

```
<c:if test=" ${!empty  sessionScope.user}">
    <h1>Welcome ${user.firstname} ${user.lastname}</h1>
<c:if test="${empty sessionScope.user}">
    <h1>Welcome to My Web Application</h1>
</c:if>
```

JSTL : JSP STANDARD TAG LIBRARY

Tag ForEach:

Ce tag permet de parcourir les différents éléments d'une collection et ainsi d'exécuter de façon répétitive le contenu de son corps.

```
<c:forEach begin="1" end= »10" var="i">
        <c:out value="${i}"/><br>
</c:forEach>
```

LE LANGAGE EL (EXPRESSION LANGUAGE)

- JSTL propose un langage particulier constitué **d'expressions** qui permettent **d'utiliser et de faire référence à des objets Java accessibles** dans les **différents contextes de la page JSP**.
- Le but est de fournir **un moyen simple d'accéder aux données nécessaires** à une JSP.
- La syntaxe de base est **${ xxx }** où :
- **xxx** est le nom d'une variable ou d'un objet Java défini dans un contexte particulier (page, requête, session ou application).

LE LANGAGE EL (EXPRESSION LANGUAGE)

```
<c:if test=" ${!empty  sessionScope.user}">
    <h1>Welcome ${user.firstname}  ${user.lastname}</h1>
<c:if test="${empty sessionScope.user}">
    <h1>Welcome to My Web Application</h1>
</c:if>
```

LE LANGAGE EL (EXPRESSION LANGUAGE)

```
<c:forEach var="i" begin="0" end="${usersList.size()-1}" step="1">

 <div class="row">
    <div class="cell"> <c:out value="${usersList[i].firstname}" /> </div>
    <div class="cell">  <c:out value="${usersList[i].lastname}" /></div>
    <div class="cell" >  <c:out value="${usersList[i].adress}" /></div>
    <div class="cell" ><c:out value="${usersList[i].phone}" /> </div>
 </div>

 </c:forEach>
```