Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

M2 MIAGE FC / FA

MIAGE LILLE

Nouvelles Technologies et Systèmes d'Informations

JSP

Loïc Nogues

Octobre 2012

People matter, results count.

Présentation générale et mécanisme de compilation

Les JSP : Java Server Pages

objectif : générer des pages HTML dynamiques

permet de séparer la présentation des traitements

- JSP, les technologies concurrentes :
  - ASP
  - PHP

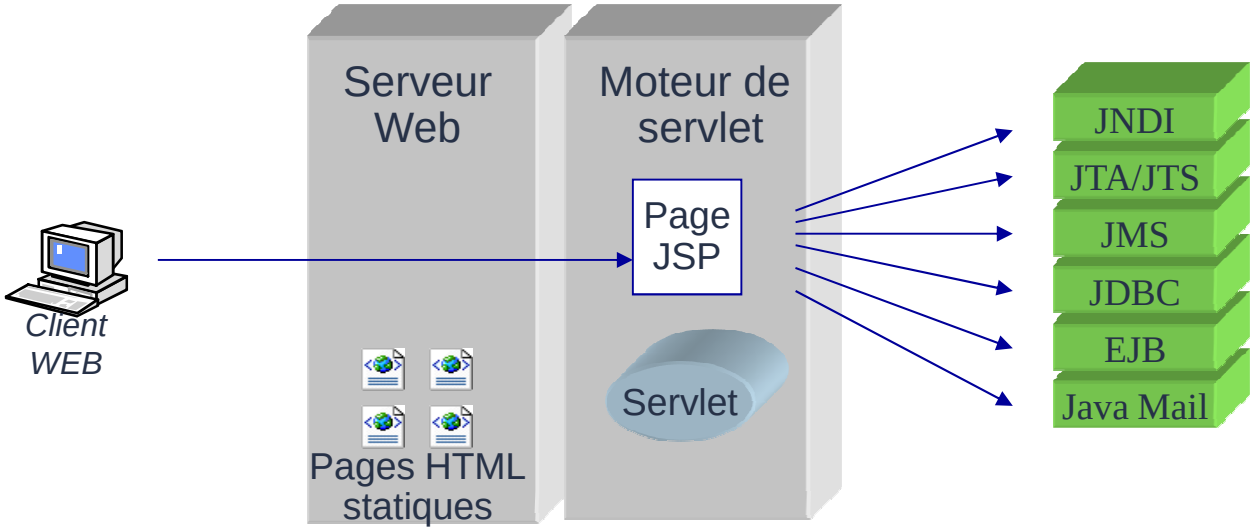
- Les fichiers JSP contiennent du code Java inséré dans des balises spécifiques, ainsi que du code HTML

- SUN fournit une spécification pour les JavaServer Pages

Présentation générale et mécanisme de compilation

Mécanisme des JSP :

- On parle de « compilation à la volée » des JSP :
  - Le serveur Web reçoit une requête pour un fichier JSP
  - Ce serveur envoie cette requête au moteur de JSP
  - Ce moteur analyse le fichier JSP demandé
  - Un fichier source (.java) est généré (à partir du JSP) et est compilé (.class)



Présentation générale et mécanisme de compilation

- Le moteur de JSP est un moteur de servlet. Une page JSP compilée est une servlet.
- Une page JSP est compilée à la première demande uniquement (puis exécutée) ; aux demandes suivantes, et si le source de la JSP n 'a pas changé, le fichier .class existant est exécuté.
- Le code source des fichiers JSP (format *servlet*) se trouve sous le répertoire du moteur de *servlets* et porte le nom de « jsp » ou « pageCompile » (utile pour déboguer).

Les tags JSP

- Les tags de directives <%@page … %>
  - la directive page : permet de définir des options de configuration
    - import : permet d'importer des classes JAVA
    - content-type : définit le language de script
    - errorPage : indique la page à afficher si une exception est lancée
  - la directive include : permet d 'inclure des fichiers dans la JSP (le code est inséré avant l 'exécution).

Syntaxe : <%@directive attribut="valeur"… %>
Exemple :

- <%@page import="java.util.*" %>
- <%@include file= "cheminRelatifDuFichier.jsp" %>
- Remarque: Déclarer en début de page

Les tags JSP

- Les tags de scripting permettent d ’intégrer du code Java dans les JSP
  - le tag de déclaration <%! … %> : déclaration de variables d ’instances, de classes, de méthodes

```
<%! private Employee emps[] = Employee.find()
<% for (int i = 0; i < emps.length; i++) { %>….
<%   } %>
```

  - le tag d ’expression <%= … %> : évalue le résultat et l ’insère sous forme de String dans la page (équivalent à out.print(...))
  - le tag de scriplet <% … %> : contient du code Java qui sera dans la méthode service de la servlet correspondante

- Exemple :
  - <% String name = request.getParameter("nom »); %>
  - <%= name %>

Les tags JSP

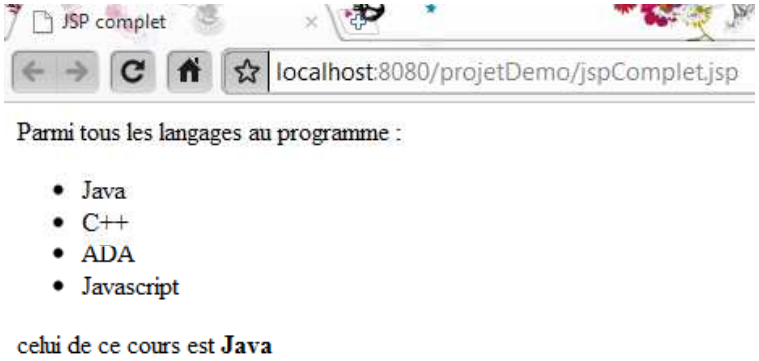
- Les tags de commentaires
  - les commentaires HTML <!-- … -->
  - les commentaires cachés <%-- … --%>

Les variables implicites

- out (javax.servlet.jsp.JspWriter) : flux en sortie de la page HTML générée
- request (javax.servlet.http.HttpServletRequest) : contient les informations de la requête
- response (javax.servlet.http.HttpServletResponse) : contient les informations de la réponse
- session (javax.servlet.http.HttpSession) : gère la session

Exemple

```
<html><head><title>JSP complet</title></head>
<body>
<%! String[] langages = {"Java", "C++", "ADA", "Javascript"};
   int random4() {
                return (int) (Math.random() * 4);
   }
%>
<p>Parmi tous les langages au programme :</p>
<ul>
<%
for (int i=0; i < langages.length; i++) {
   out.println("<li>" + langages[i] + "</li>");
}
%>
</ul>
<p>celui de ce cours est <b><%= langages[random4()] %> </b></p>
</body>
</html>
```



Les tags JSP

- Les tags d ’action
  - le tag <jsp : useBean> : permet de localiser ou d ’instancier un bean
  - le tag <jsp : setProperty> : permet de mettre à jour la valeur d’une propriété d’un bean
  - le tag <jsp : getProperty> : permet d’accéder à la valeur d’une propriété d ’un bean
  - le tag de redirection <jsp : forward> : permet de rediriger la requête vers une autre url (équivalent au sendRedirect de la servlet)
  - le tag <jsp : include> : permet d’inclure le contenu d’un fichier dynamiquement

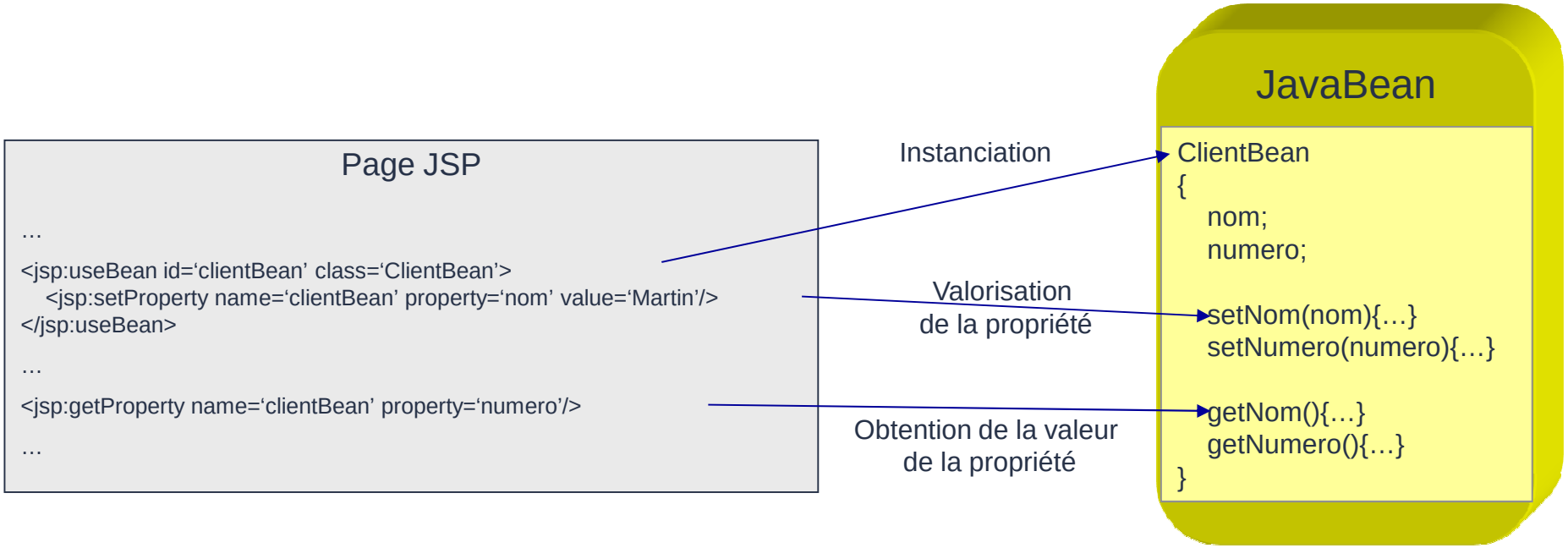
Utilisation des Beans dans une page JSP

- Présentation des JavaBeans
  - composants Java pour encapsuler le code Java (allègement des pages JSP)
  - Ces composants suivent des règles de développement (qui dépendent de l 'utilisation du Bean : composant graphique ou non, …).

- Les méthodes
  - Un Bean doit posséder un constructeur sans paramètre (qui doit initialiser les attributs du Bean)
  - Un Bean peut définir des propriétés : Il s'agit de méthodes (des accesseurs) dont le nom et la signature sont normalisés.

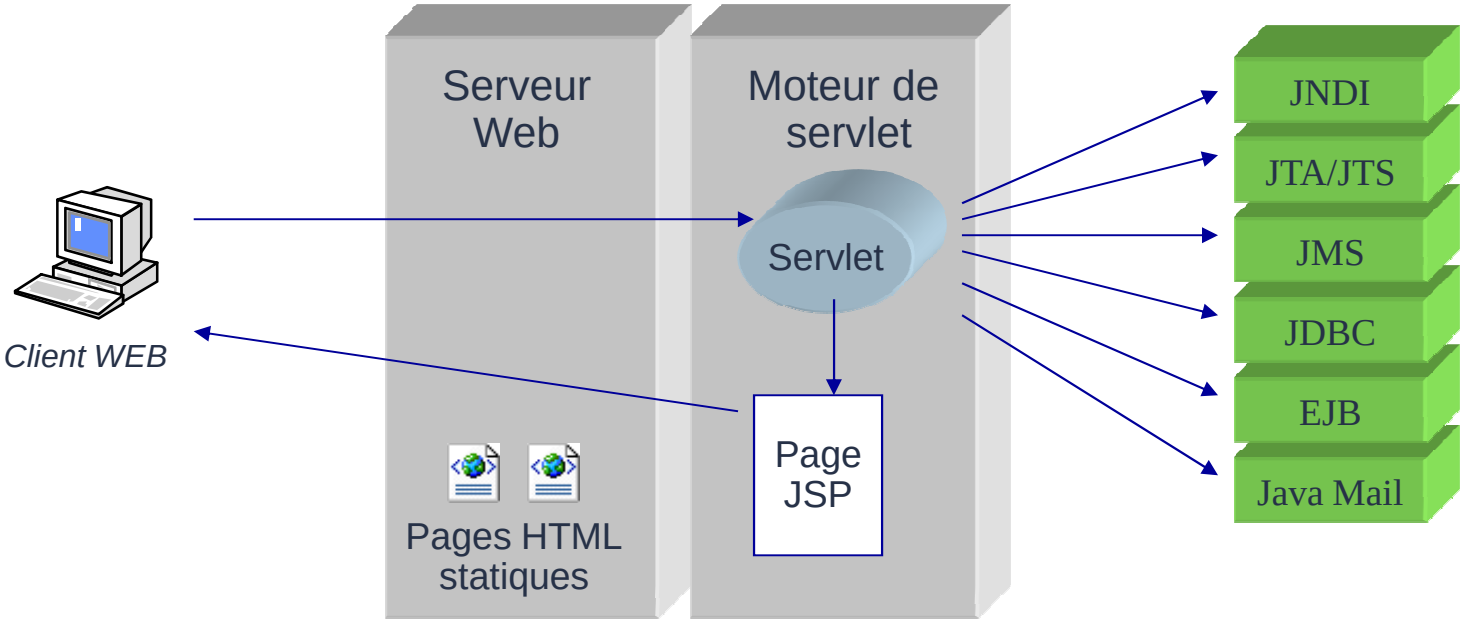
Utilisation des Beans dans une page JSP

Utilisation dans les pages JSP:



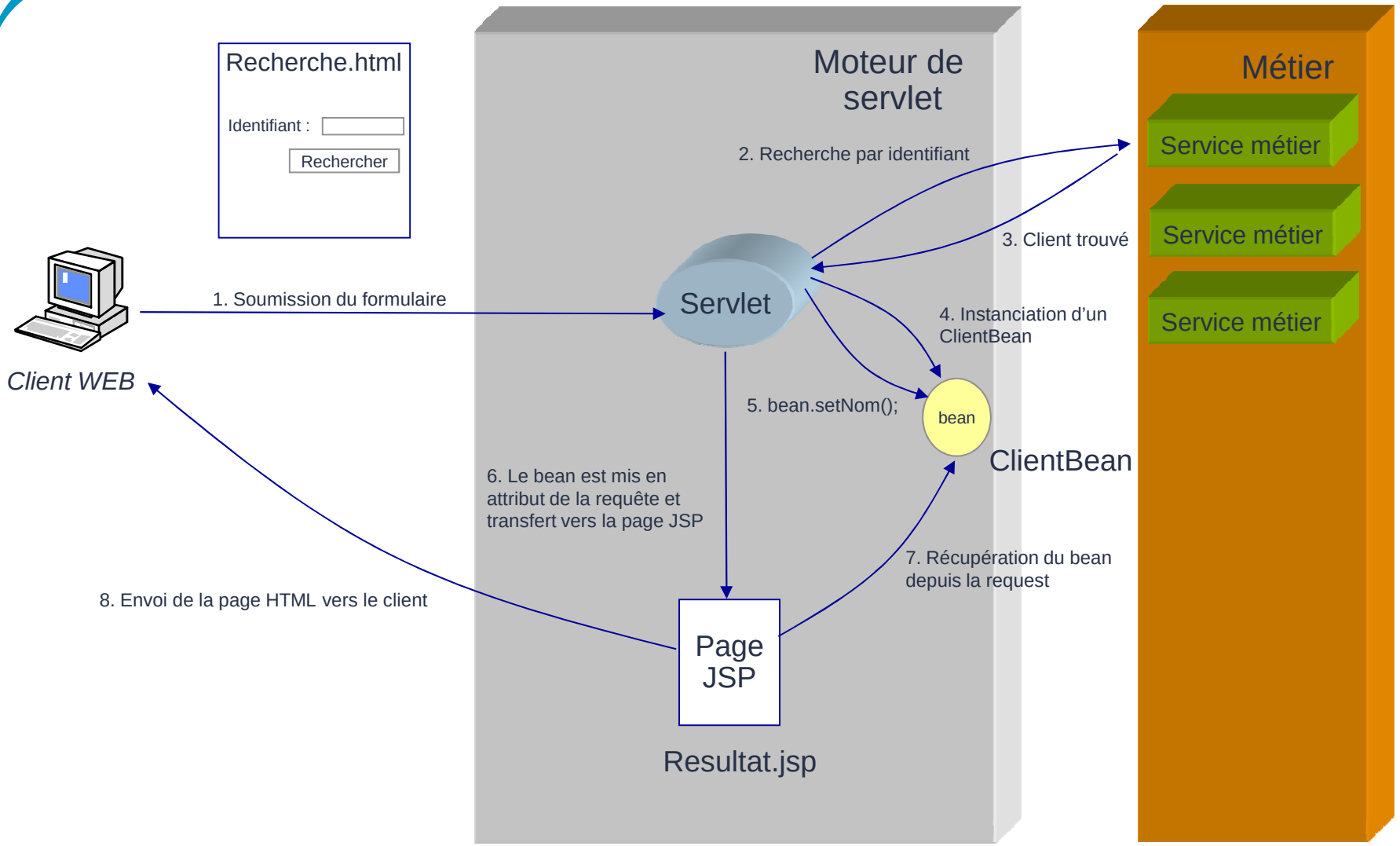
Couplage JSP / Servlet

- Le couplage des servlets et des JSP permet de séparer clairement la partie « traitement » de la partie « affichage » du résultat.



- Ainsi, la page JSP va s'appuyer sur des JavaBean et Taglib pour présenter (sous forme de code HTML) les résultats de la requête au client.

Couplage JSP / Servlet



Couplage JSP / Servlet

- 1. Le client saisit l'identifiant dans le formulaire et lance la recherche : la Servlet reçoit la requête (HttpServletRequest)
- 2. La Servlet récupère l'identifiant passé en paramètre de la requête (req.getParameter(«identifiant»)) et appelle le métier pour recherche
- 3. Le métier renvoie une personne trouvée
- 4. La Servlet instancie un bean de type ClientBean
- 5. La Servlet positionne le nom du client renvoyé par le métier dans le bean : bean.setNom()
- 6a. La Servlet met le bean en attribut de l'objet HttpServletRequest
- 6b. La Servlet transfert la requête vers la JSP (RequestDispatcher)
- 7. La JSP récupère le bean depuis l'objet request
- 7b. La JSP affiche le nom du client depuis le bean
- 8. La page HTML est renvoyée vers le client

Couplage JSP / Servlet

- Appel de la servlet :
  - Une requête HTTP sera émise dès que l'utilisateur aura cliqué sur le bouton « Rechercher ». La servlet récupèrera l 'identifiant saisi par l 'utilisateur dans le formulaire HTML par la méthode « doPost() », effectuera le traitement pour obtenir l 'objet métier Client voulu puis créera un objet JspBean (JSP JavaBean) qu'il initialisera par les « input » du formulaire HTML.

```
<html>
  <head><title>PageRecherche</title></head>
  <body>
   <form action="/servlet/RechercheServlet" method="POST">

      <BR><BR>Recherche d'un client :<BR>
         <BR><input type="text" name="identifiant">
         <BR><BR><input type="submit" name="Rechercher">

    </form>
  </body>
</html>
```

Couplage JSP / Servlet

Déclaration de la classe client.ClientBean

```
package client;
class ClientBean {
       private String nom;
       private int numero;

       public void setNom(String pNom) {
                nom = pNom;
      }

      public String getNom() {
                return nom;
      }
…
```

La Servlet appelle le métier et crée le bean

```
String identifiant = request.getParameter(«identifiant»);
Client client = serviceRecherche.getClient(identifiant); // Appel métier
ClientBean clientBean = new ClientBean(); // Instanciation du bean
clientBean.setNom(nomClient); // Remplissage du bean
…
```

Couplage JSP / Servlet

- Transfert de la servlet vers la JSP :
  - La Servlet positionne le bean en attribut de l'objet request

```
//objet request de type HttpServletRequest
request.setAttribute("clientBean", clientBean);
```

  - L 'objet de type `RequestDispatcher` est créé par le moteur de servlet et permet d 'appeler la page JSP.

```
// Code de la servlet pour obtenir l'objet RequestDispatcher
RequestDispatcher dispatcher =
   request.getRequestDispatcher("/jsp/Resultat.jsp");
```

  - La requête est transmise à la page JSP qui va se charger de retourner la réponse au poste client.

```
//Méthode « forward » pour appeler la page JSP
dispatcher.forward (request, response);
```

Questions

?