Javabean et le web

Noël De Palma
noel.depalma@inrialpes.fr
Professeur Université de Grenoble
Remerciement F. Gayral, L. Seinturier

Javabean et le web

- Intégration coté client
  - **Applet**
    - Programme java inclus dans une page HTML
    - Technologie internet cote ≪ client ≫
- Intégration coté serveur
  - **Servlet, JSP ...**

Architecture du WEB

- Web : architecture client/serveur (requete/reponse)
- Le client, en general un navigateur, envoie a un serveur une requete d'acces a une ressource : fichier (HTML, image, class), script
- Le serveur traite la requete et renvoie une reponse au client

Standard du Web

- Langage d'ecriture des documents : **HTML (HyperText Markup Language)**
- Adresses des documents sur Internet : **URL (Unified ressource Locator)**
- Protocole de communication entre client et serveur : **HTTP (HyperText Transport Protocol)**
- Typage des documents du Web : MIME (Multipurpose Internet Mail Extensions)
    - exemples : image/gif, text/html

Exemple HTML basique

Un exemple : fichier index.html

```
<HTML>
<HEAD>
<TITLE> Exemple pour le cours </TITLE>
</HEAD>
<BODY>
C'est ici qu'on &eacute;crit le texte de la page en faisant
attention aux accents !!!
<H1> Bonjour ! </H1>
<IMG SRC=“tomcat.gif">
</BODY>
</HTML>
```

Communication Client/Serveur quand la ressource est une page HTML

- Le fichier index.html contient la ressource tomcat.gif
- Les deux fichiers (html et gif) sont presents sur le serveur
- Le navigateur envoie la premiere requete

http: //www.lipn-univ-paris13.fr/index.html

- Pour chaque ressource identifiee dans le code HTML, le navigateur envoie une nouvelle requete au serveur
- ici, une requete pour le fichier tomcat.gif

Structure d'un document HTML

Déclaration version HTML utilisée

En-tête

Corps du document

```
<! DOCTYPE ... >
<HTML>
<HEAD>
En-tête
</HEAD>
<BODY>
Corps du document
</BODY>
</HTML>
```

Envoie de requête à un serveur WEB

Message GET ou POST

GET : paramètres inclus dans l'URL

http://nom_du_serveur/cgi-bin/script.cgi?
champ1=valeur1&champ2=valeur2...

Limitation à 255 caractères, visible ...

POST : paramètres inclus dans les entêtes HTTP

Utilisation de formulaires (pour interactions)

Balises principales

`<BODY attr1="val1" ... attrn="valn">`

attributs possibles **valeurs par défaut** en cas d'absence

- `BGCOLOR` : couleur de fond
- `TEXT` : couleur du texte
- `BACKGROUND` : URL de l'image de fond d'écran
- ...

`<H1>Titre</H1> <H2> <H3> <H4> <H5> <H6>`

attribut possible `ALIGN="left|center|right"`

`<P> ... </P>` paragraphe `<BR>` passage à la ligne `<HR>` trait horizontal

`<B> ... </B>` gras `<I> ... </I>` italique

Balises principales

Liens hypertexte : Portion de texte permettant d'atteindre un document désigné par une **URL**

```
<A HREF="URL">texte du lien</A>
```

URL absolue `<A HREF="http://www.lip6.fr/index.html">LIP6</A>`

URL relative `<A HREF="sommaire.html">sommaire</A>`

```
<A HREF="../divers/plan.html">plan d'accès</A>
<A HREF="/index.html">accueil</A>
```

Balises principales

Insertion d'une image (gif ou jpeg) dans un document

```
<IMG SRC="URL (absolue ou relative) du fichier image">
```

Listes numérotées (<OL> ... </OL>) ou non (<UL> ... </UL>)

```
<UL> balise de début de liste
<LI> Rouge </LI> balises pour un élément de la liste
<LI> Vert </LI> balises pour un élément de la liste
...
</UL> balise de fin de liste
```

Attributs possibles

<UL TYPE="**disc**|circle|square"> : type de puce

<OL TYPE="**1**|I|i|A|a" : type de numérotation

START="99"> : départ de la numérotation

<LI TYPE="disc|circle|square"> : type de puce (liste UL)

<LI TYPE="1|I|i|A|a "> : type de numérotation (liste OL)

Tableaux

Tableaux à 2 dimensions (<TABLE> ... </TABLE>)

<TABLE> *balise de début de tableau*

<TR> *balise de début de ligne (row)*

<TD> cellule </TD> *balises pour une colonne (data)*

...

</TR> *balise de fin de ligne*

...

</TABLE> *balise de fin de tableau*

Attributs possibles pour <TABLE>

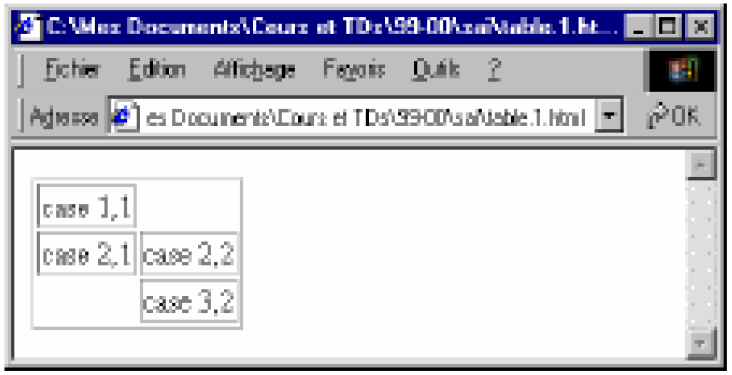
- BORDER="99" (**0**) : épaisseur en pixels des bordures du tableaux
- WIDTH="99|99%" : largeur du tableau en pixels | en % largeur fenêtre

Attributs possibles pour <TR>

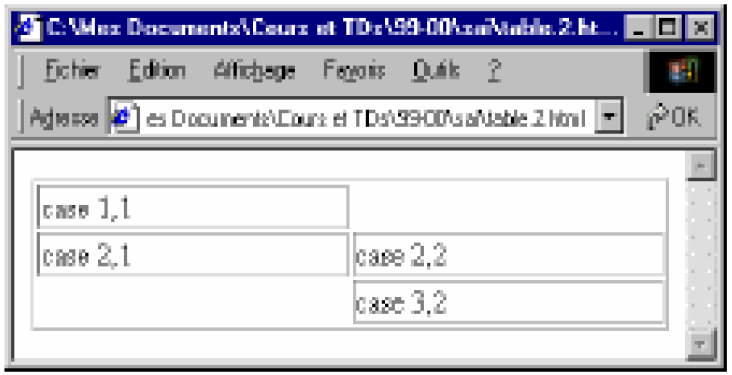
- ALIGN="**left**|right|center|justify" : alignement horiz. du texte (≠ <TABLE>)
- VALIGN="**middle**|top|bottom" : alignement vertical du texte

Tableaux

```
<TABLE BORDER="1">

<TR> <TD> case 1,1 </TD> </TR>

<TR> <TD> case 2,1 </TD>
     <TD> case 2,2 </TD> </TR>

<TR> <TD></TD>
     <TD> case 3,2 </TD> </TR>

</TABLE>
```



WIDTH="100%"

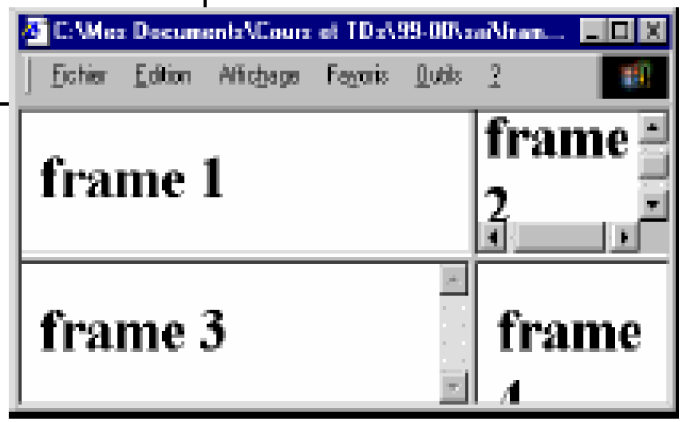


frames

Découpage de la fenêtre du navigateur en plusieurs parties.

Exemple de fichier de *frameset* - `index.html`

```
<HTML>
<FRAMESET ROWS="50%,50%" COLS="70%,*">
    <FRAME SRC="frame1.html" NORESIZE>
    <FRAME SRC="frame2.html">
    <FRAME SRC="frame3.html" SCROLLING="yes">
    <FRAME SRC="frame4.html" SCROLLING="no">
</FRAMESET>
</HTML>
```



Attributs `ROWS/COLS`

- `*` : taille restante
- $\sum > 100\%$ | taille fenêtre en pixels ⇨ règle de 3
- $\sum < 100\%$ | taille fenêtre ⇨ dernière taille ignorée

Feuilles de style CSS (Cascading Style Sheets)

Permettent de factoriser des éléments de présentation pour 1 ou +sieurs pages

Définition des styles dans un fichier .css

```
balise1 { prop1:val1; ... ; propn:valn; }
balise2 { prop1:val1; ... ; propn:valn; }
...
ex : H1 { font-style: italic; color: blue; }
```

Liaison entre le fichier .html et le fichier .css par la balise <LINK>

```
<HTML>
<HEAD>
<LINK REL="stylesheet" HREF="f.css">
</HEAD>

<BODY> <H1>Titre en ... </H1>
</BODY></HTML>
```

Formulaires HTML

Ecriture d'un formulaire

```
<form action="url" method="methode">
 ...
</form>
```

url : identife le programme utilisé pour traiter le formulaire

methode : méthode à utiliser pour transmettre l'information au serveur

- GET : données ajoutées à l'URL
- POST : données envoyées dans le corps du message

Éléments de formulaires

Balise INPUT (exclusivement entre <form> et </form>)

```
<input type="type" name="nom" size="size"
  maxlength="max" checked="checked" value="val"/>
```

Différents types possibles

TEXT	Champ de saisie de texte (défaut)
PASSWORD	Champ de saisie de texte caché
SUBMIT	Bouton de soumission du formulaire
CHECKBOX	Bouton à cocher
RADIO	Bouton à cocher de type radio. (même nom => même groupe)
HIDDEN	Champ invisible
RESET	Effacement du formulaire
FILE	Choix d'un fichier à envoyer

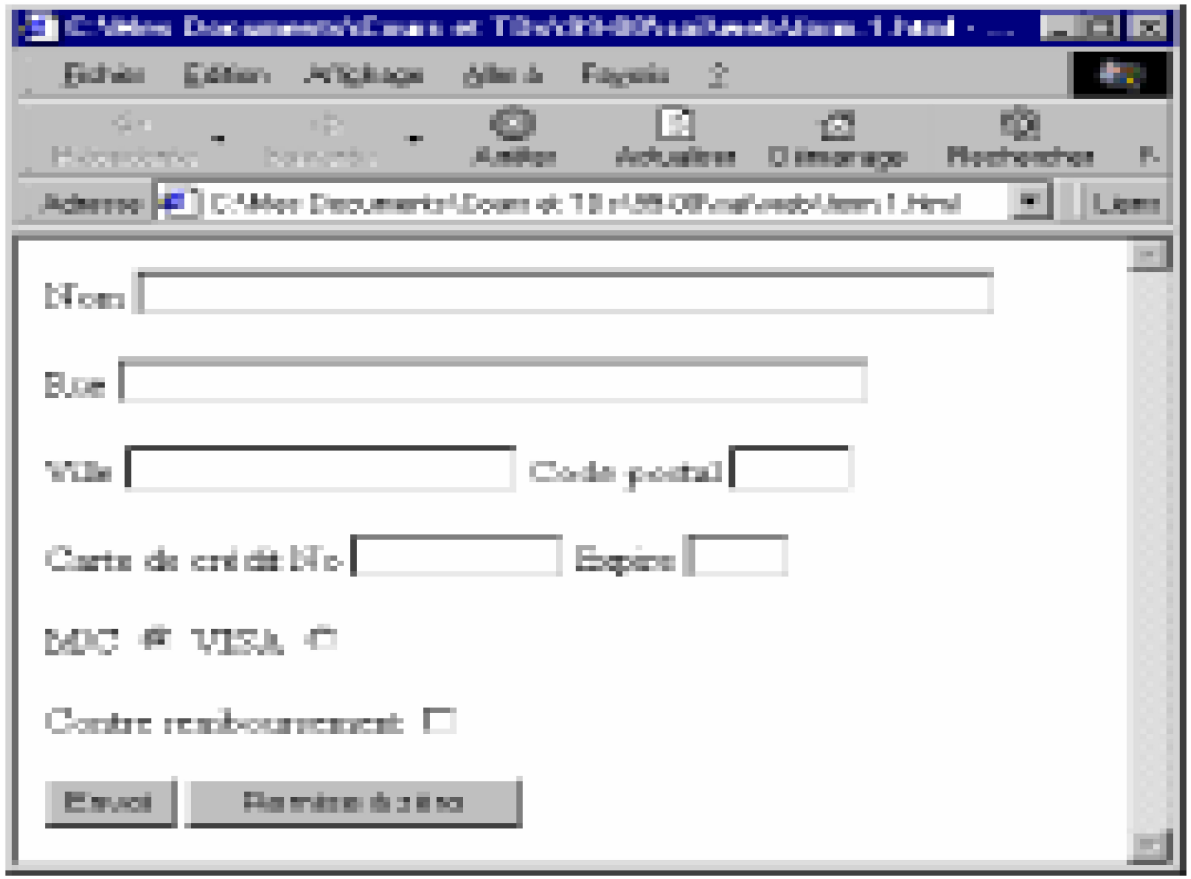
Lorsque l'utilisateur click sur submit, le navigateur envoie une requete get ou post avec les paramètres

Exemple

```
<HTML> <BODY>
<FORM ACTION="http://monserveur.com/prog.php" METHOD=POST>
    Nom <INPUT NAME="client" SIZE=46> <P>
    Rue <INPUT NAME="rue" SIZE=40> <P>
    Ville <INPUT NAME="ville" SIZE=20>
    Code postal <INPUT NAME="cp" SIZE=5> <P>
    Carte de crédit No <INPUT NAME="carte" SIZE=10>
    Expire <INPUT NAME="expire" TYPE=TEXT SIZE=4> <P>
    M/C <INPUT NAME="cc" TYPE=RADIO VALUE="mc" CHECKED>
    VISA <INPUT NAME="cc" TYPE=RADIO VALUE="vis"> <P>
    Contre remboursement <INPUT NAME="cr" TYPE=CHECKBOX> <P>
    <INPUT TYPE=SUBMIT VALUE="Envoi">
    <INPUT TYPE=RESET VALUE="Remise à zéro"> <P>
</FORM> </BODY> </HTML>
```


JAVA

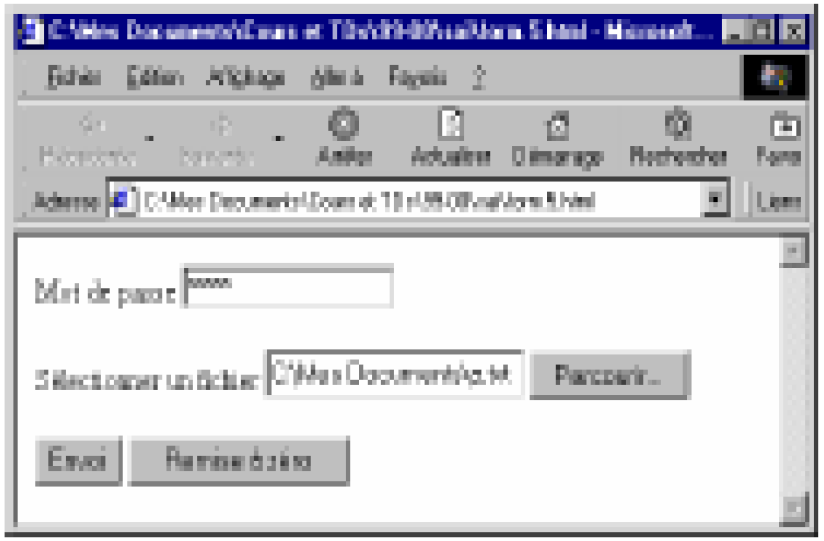
Exemple


Nom
Rue
Ville
Code postal
Carte de crédit No
Expire
MDC
VISA
Contre remboursement

Exemple

```
<FORM ACTION="http://monserveur.com/prog.jsp" METHOD=POST
ENCTYPE="multipart/form-data">
Mot de passe <INPUT TYPE=PASSWORD NAME="passe" SIZE=16> <P>
Sélectionner un fichier <INPUT TYPE=FILE NAME="fichier"> <P>
<INPUT TYPE=SUBMIT VALUE="Envoi">
<INPUT TYPE=RESET VALUE="Remise à zéro"> <P>
</FORM>
```

Exemples



PASSWORD	les caractères saisis sont remplacés par des *
FILE	provoque l'affichage - d'un champ de saisie du nom du fichier - d'un bouton `Parcourir` pour sélectionner le fichier via une fenêtre de parcours du disque

Eléments de formulaires

Elements SELECT

```
<select name="nom" size="" multiple="">
    <option>Première option</option>
    <option selected="selected">Deuxième option</option>
    ...
</select>
```

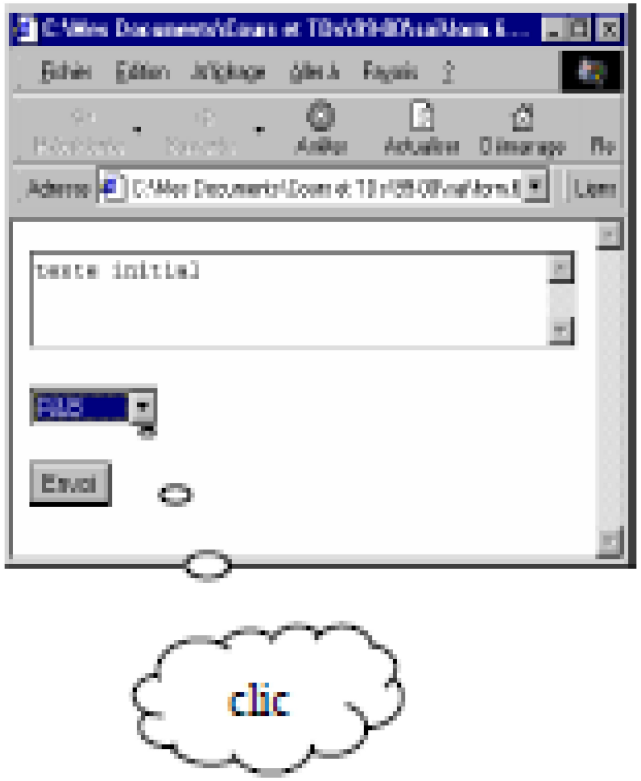
Element TEXTAREA

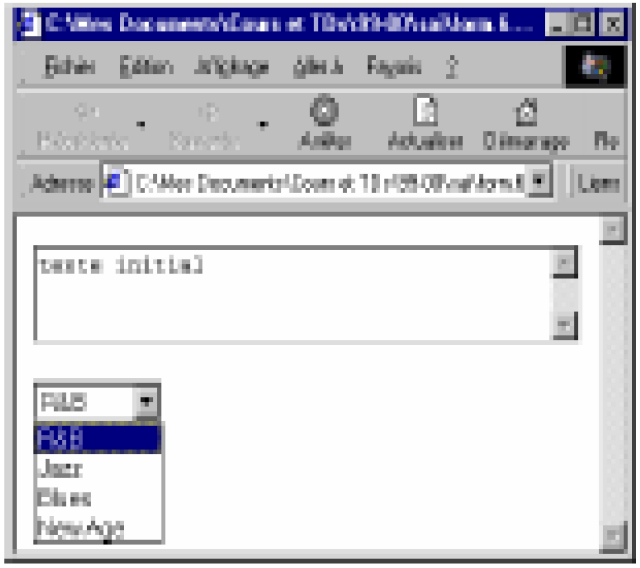
```
<textarea name="nom" rows="nbLignes" cols="nbColonnes">
 texte par defaut
</textarea>
```

Exemple

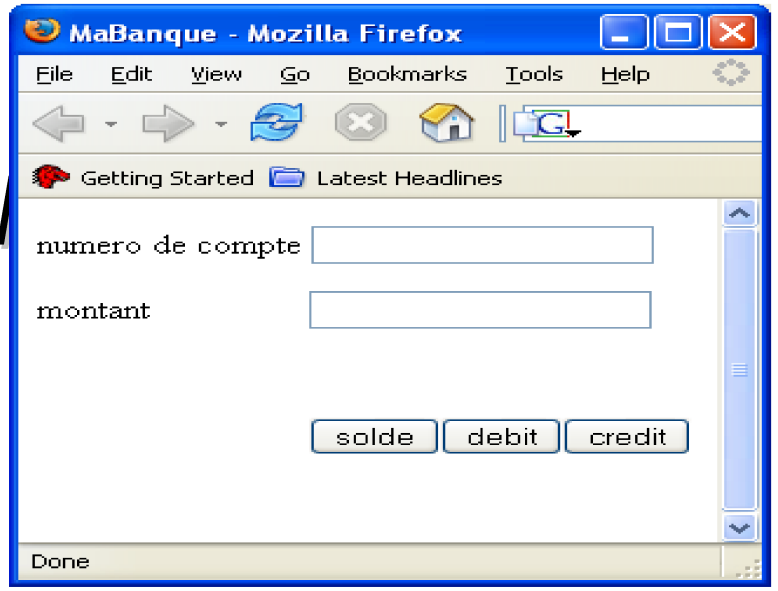
```
<HTML> <BODY>
<FORM ACTION="http://monserveur.com/prog.jsp"
  METHOD=POST>
<TEXTAREA NAME="zone" ROWS=3 COLS=40>texte
  initial</TEXTAREA> <P>
<SELECT NAME="musicTypes">
      <OPTION> R&B
      <OPTION> Jazz
      <OPTION> Blues
      <OPTION> New Age
</SELECT> <P>
<INPUT TYPE=SUBMIT VALUE="Envoi">
</FORM> </BODY> </HTML>
```

Exemple





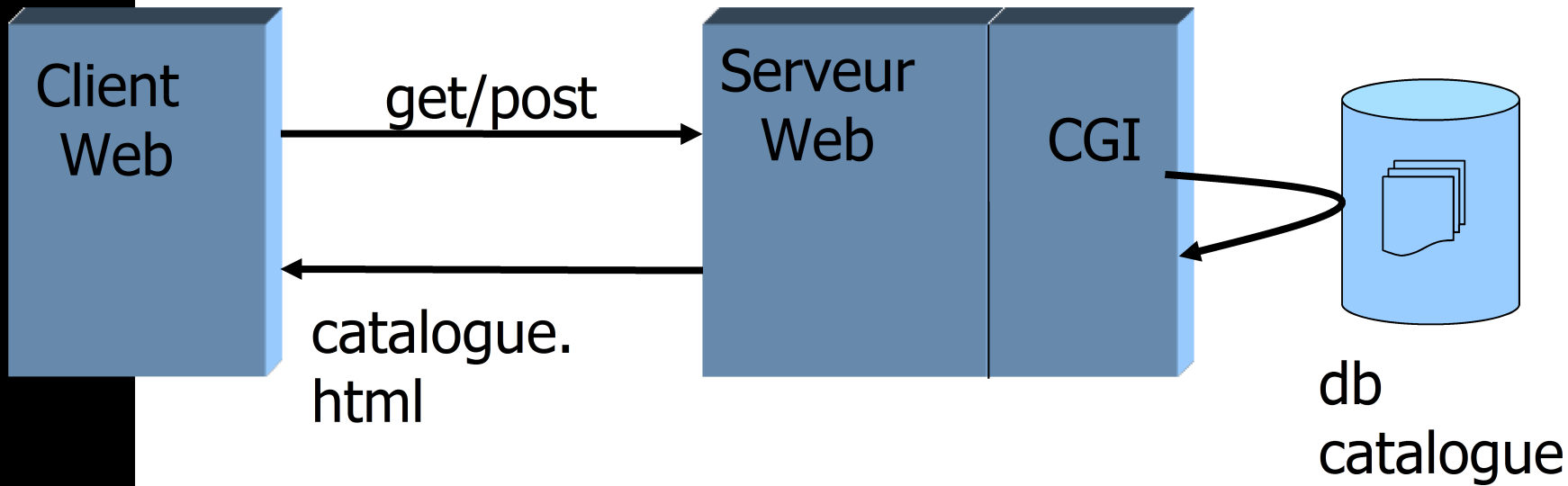
Exempl




```
<html>
    <head><title>MaBanque</title></head>
    <body>
        <form method="post"  action="/servlet/BanqueAccount">
            <p>numero de compte<input type="text" name= "num"></p>
            <p>montant<input type="text" name= "val"></p>
            <p><input type="submit" name="op1" value="solde">
                <input type="submit" name="op2" value="debit">
                <input type="submit" name="op3" value="credit"></p>
        </form>
     </body>
</html>
```

Page WEB dynamique

- Page web html généré dynamiquement
- Éventuellement à partir d'informations provenant d'une BD



Orientation vers des solutions plus flexibles

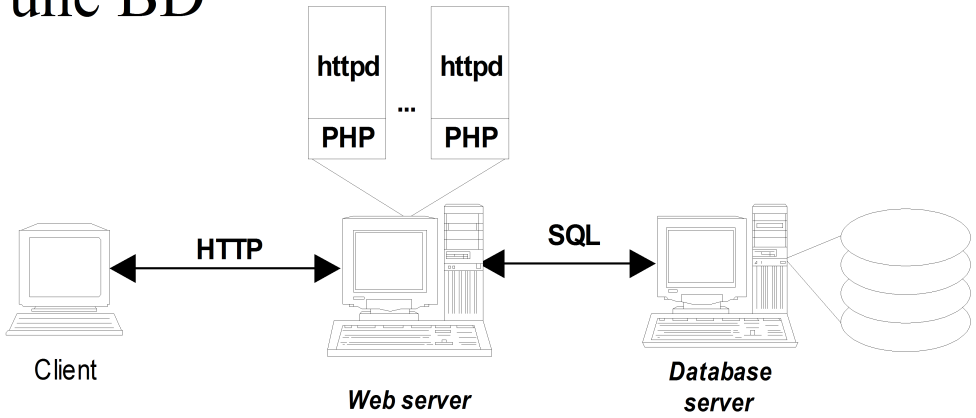
- PHP
  - Balises spécifiques au sein d'une page HTML
  - Langage de script exécuté coté serveur
- Servlet
  - Programme coté serveur écrit en Java
- JSP
  - Balises spécifiques au sein d'une page HTML
  - Bonne intégration à Java

A ne pas confondre avec

- Applets
  - Programme Java
  - Référencé depuis une page HTML
  - Stocké sur le serveur Web
  - Chargé et exécuté par le client (browser) web
- JavaScript
  - Langage de script
  - Balises spécifiques au sein d'une page HTML
  - Chargé et exécuté par le client (browser) web

Introduction au langage de script : PHP

- Langage de script
  - Générer des pages dynamiques
  - Intégré dans des pages web
  - Interprété (PHP3) ou compilé (PHP4)
  - Dérive de C et Perl
  - Facilite les accès à une BD



Intégration d'un script PHP dans une page

- Fichier .php
  - Html + script

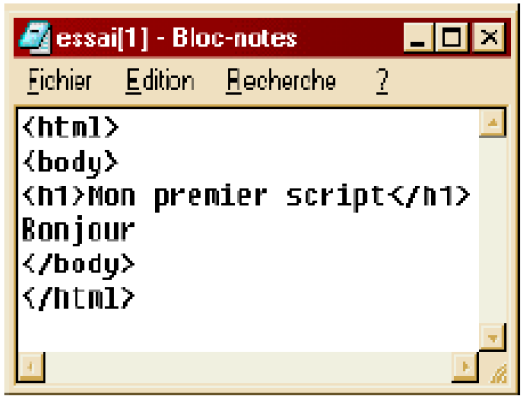
```
 n premier script</h1>
<?php echo "Bonjour\n";?>
>
>
```

```
<?php
echo "<html>\n<body>\n";
echo "<h1>mon premier
                    script</h1>\n";
echo "Bonjour\n";
echo "</body>\n</html>\n";
?>
```

Résultat

Résultat affiché par le navigateur :

Code source (côté client) de la page essai.ph3 résultant du script



Eléments du langage (1/2)

- Pas de déclaration
- Typage implicite (entier, flottants, chaînes, booléens)
- Variable : $toto
- Fonctions de test
  - isset($toto)
  - unset($toto)
  - is_integer($toto)

- Opérateurs & Structure de ctrl
  - Idem que C
  - Conditionnelles (if, switch) et boucles (for, while)

Eléments du langage (2/2)

- Tableaux
  - Clé / valeur
  - Pas forcément du même type
- A la volée
  - $tab[0] = 54;
  - $tab[1] = "pif";
  - $tab["paf"] = false;
- Direct
  - $tab = array(54,"pif");
  - $tab = array("paf" => false);
- Parcours
  - Foreach ($tab as $value) { .. }

- Les fonctions
  - Paramètre par valeur
  - Paramètre par référence

```
// par valeur
Function triple($x){
$x=$x*3;
return $x;
}

// par référence
Function triple(&$x){
$x=$x*3;
}
```

Exemple : petit formulaire (1/2)

- Formulaire HTML

```
<html>
  <head><title>helloworld</title></head>
  <body>
  <h1>Nom de la personne a rechercher</h1>
   <form action= "helloworld.php" method="post">
      <p>Nom<input type="text" name="firstname">
      <p>Prenom<input type="text" name="lastname">

      <input type="submit" value="Envoyer"></p>
   </form>
 </body>
</html>
```

- Le script PHP reçoit un tableau
  - $_GET pour la méthode GET
  - $_POST pour la méthode POST

Exemple : petit formulaire (2/2)

- Page PHP

helloworld.php

```
<html>
<body>
<h1>Informations soumises</h1>
<?php
        echo "Vous etes";
         echo $_POST['firstname'];
         echo $_POST['lastname'];
?>
</body>
</html>
```

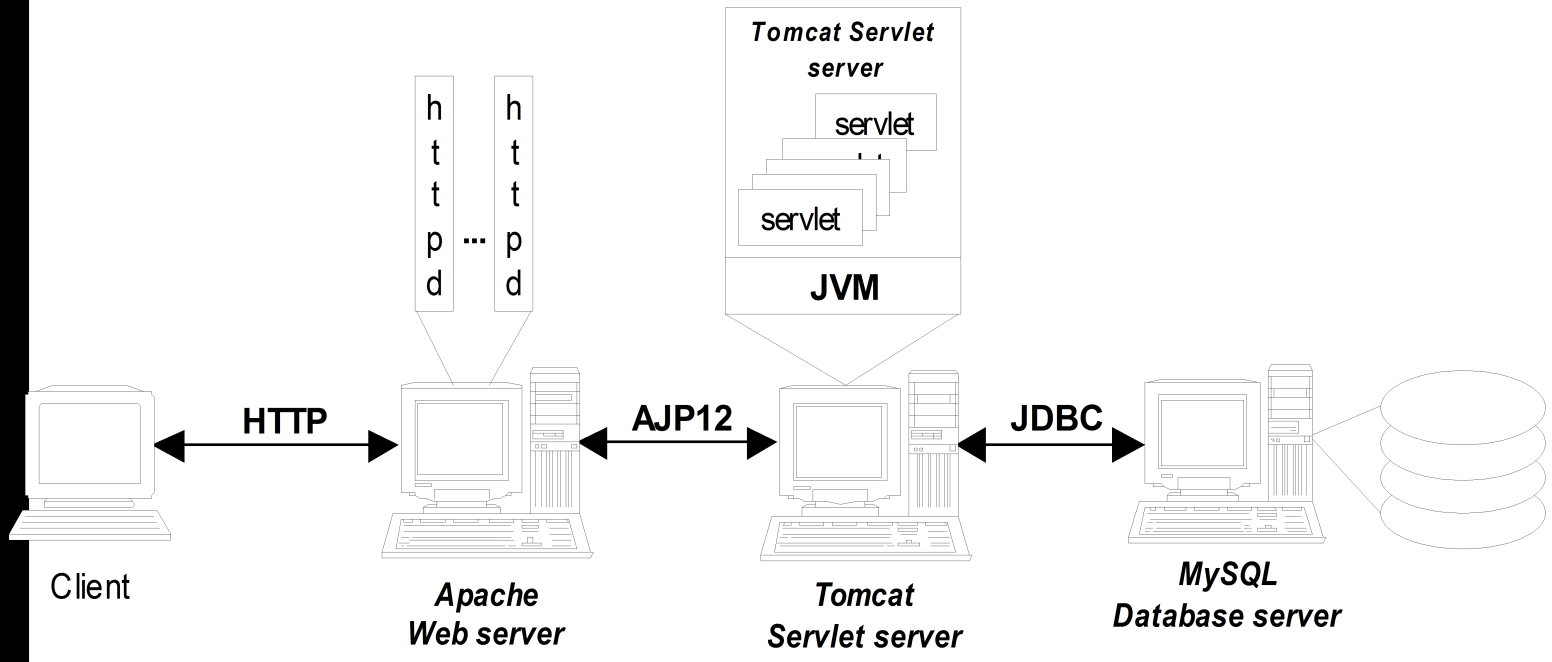
PHP
Bilan

- Facile à utiliser
  - Juste des pages Web étendues
- Pas toujours très propre
  - Script, pas du Java (typage)
  - Interface BD
  - Mélange entre aspects présentation (génération HTML), code métier et code d'accès aux données persistantes

Servlets Java

exécute dans un “Servlet Container” sur une JVM

- Serveurs JAVA : Java Web Server ou Tomcat
  - Font aussi serveur web
- Serveur Plug-in de Apache



Servlet HTTP - API

c void **init**()

cted void **doGet**(HttpServletRequest req, HttpServletResponse resp)

cted void **doPost**(HttpServletRequest req, HttpServletResponse resp)

aramètres

- HttpServletRequest : permet de manipuler la requête reçue
- HttpServletResponse : permet de générer la réponse

emarques

- Attention, ces méthodes peuvent être exécutées en concurrence
- Ces méthodes peuvent appeler des BD : JDBC

Exemple (1/2)

```
<html>
  <head><title>helloworld</title></head>
  <body>
  <h1>Nom de la personne a rechercher</h1>
   <form action= "helloworld.php" method="post">
      <p>Nom<input type="text" name="firstname">
      <p>Prenom<input type="text" name="lastname">
        <input type="submit" value="Envoyer"></p>
   </form>
 </body>
</html>
```

Exemple 2/2

```
import java.io.*; import javax.servlet.*; import javax.servlet.http.*;
public class HelloWWW extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
      IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 " +
                                "Transitional//EN\">\n" +
              "<HTML>\n" +
              "<HEAD><TITLE>Hello WWW</TITLE></HEAD>\n" +
              "<BODY>\n" +
              "<H1>Hello "+ req.getParameter("firstname")+" " +
                                 req.getParameter("lastname")+
             "</H1>\n" +
              "</BODY></HTML>");
  }
}
```

Session

- Notion de session
  - Une requête dépend du résultat des requêtes précédentes
  - Ex : caddie
- Création de session
  - HttpSession **HttpServletRequest.getSession**()
  - HttpSession **HttpServletRequest.getSession** (boolean create)

HttpSession

Object getAttribute(String name)
Enumeration getAttributeNames()
Long getCreationTime()
String getId()
int getMaxInactiveInterval()
void invalidate()
void removeAttribute(String name)
void setAttribute(String name, Object value)
void setMaxInactiveInterval(int interval)
...

Gestion des cookies

- Création/initialisation

  Cookie(java.lang.String name, java.lang.String value)
  void **setValue**(java.lang.String newValue)
  void **setMaxAge**(int expiry)
  void **setDomain**(java.lang.String pattern)
  java.lang.String **getValue**()
  java.lang.String **getDomain**()
  int **getMaxAge**()

- A l'exécution

  Cookie[] HttpServletRequest.getCookies()

  HttpServletResponse.addCookie(javax.servlet.http.Cookie)

Packaging d'une servlet

Un répertoire par application

- Pages web (html)
- Répertoire "WEB-INF"
  - Répertoire "classes" : les classes des servlets
  - "web.xml" : descripteur des servlets

```
<web-app>
   <servlet>
      <servlet-name>myBanque</servlet-name>
      <servlet-class>BanqueAccount</servlet-class>
   </servlet>
   <servlet-mapping>
      <servlet-name>myBanque</servlet-name>
      <url-pattern>/URL_Banque</url-pattern>
   </servlet-mapping>
```

Installation dans Tomcat

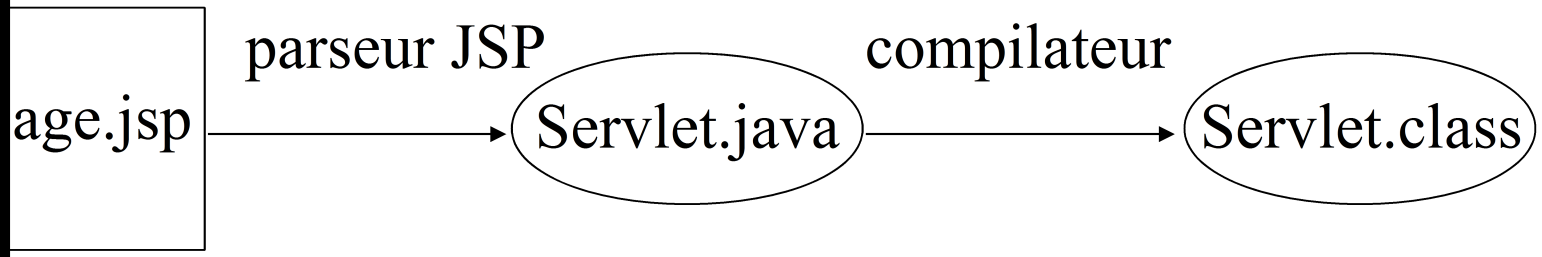
- Création d'un fichier WAR (jar)
- Copie dans $CATALINA_HOME/webapps
- Le fichier WAR est expansé

Servlet
Bilan

- Facile à programmer
  - Programmation en Java et API simple
  - Mais pas aussi simple qu'un script comme PHP
- Mieux adapté que les scripts à des traitements plus complexes
  - Interface JDBC pour accès à différentes BD SQL
  - Extension plus facile (liaison dynamique de Java) que les extensions d'un langage de script
- Le traitement des données récupérées de la BD peuvent être lourds (d'où les EJB)

JSP (Java Server Page)

- Langage de script (proche de Java)
  - Générer des pages dynamiques
  - Intégré dans des pages web
  - Compilé dynamiquement en servlet
- Interation avec des classes Java



Un petit exemple

```
<%@ page language="Java" %>
<html>
<head>
<title>First.jsp</title>
</head>
<body>
<h1>Nombres de 1 à 10</h1>

<% for(int i=1; i<=10; i++) {
        out.println(i + "<br>");
    }
%>
</body>
</html>
```

Les directives

- <%@ directive attribut1="valeur" attribut2="valeur"... %>

- 3 directives possibles :
  - page (informations relatives à la page)
    - <%@ page import="..."%>
  - include : fichiers à inclure littéralement (file)
    - <%@ include file="..."%>
  - Taglib : permet d'utiliser des librairies de tags personnalisés
    - <%@ taglib uri="..." prefix="..."%>

Les déclarations

- <%! declaration %> variables et méthodes globales à la page

- Exemple

```
<%!
String Chaine = "bonjour";
Int Numero = 10;
public void jspInit() {
    // instructions;
}
%>
```

Les scripts Java

- Du code Java : <% code Java %>
- Des évaluations d'expression : <%= expression %>
- Des variables prédéfinies

```
<%@ page language="Java" %>
<html><head><title>First.jsp</title>
</head><body>
<h1>Nombres de 1 à 10</h1>
<% for(int i=1; i<=10; i++) {  %>
  <%= i %> <br/>
<%  } %>
</body>
</html>
```

Variables prédéfinies

HttpServletRequest request
HttpServletResponse response
HttpSession session
ServletContext application
PrintWriter out
Object page
ServletConfig config
javax.servlet.jsp.PageContext pageContext
Throwable exception

Les actions

- **Des tags standards des JSP de la forme**

  <jsp:tag attribut1="valeur" attribut2="valeur"... %>

- **<jsp:forward page="page2.jsp" />**
  - Transfère le contrôle à une autre page JSP (annule l'appelante)
- **<jsp:include page="page2.jsp" />**
  - Transfère le contrôle à une autre page JSP (inclusion)
- Peuvent prendre des paramètres avec

<jsp:param name="..." value="..." />

une JSP ou une Servlet)

Les actions

- **<jsp:usebean id="nomAttribut" class="package.classe" scope="portéeAttribut>**

  <%-- code executé si l'attribut est créé --%>

  </jsp:usebean>
- Importe un attribut si il existe, le crée sinon

<jsp:usebean id="personne" class="testjsp.Personne" scope="session/>

équivaut à

```
<% testjsp.Personne personne = (testjsp.Personne)
session.getAttribute("personne");
    if (personne == null) {
        personne = new testjsp.Personne();
        session.setAttribute("personne", personne);
    } %>
```

Les actions

- **<jsp:setProperty name="nomAttr" property="nomProp"/>**
- S'utilise en complément de useBean
- La classe Java doit être un javaBean (constructeur vide, set(), get())
- Initialise un javaBean à partir des paramètres de formulaire

```
<jsp:usebean id="personne" class="testjsp.Personne"
scope="session/>
<jsp:setProperty name="personne" property="nom" />
```

équivaut à

```
<jsp:usebean id="personne" class="testjsp.Personne"
scope="session/>
<%  if (request.getParameter("nom") != null)
        personne.nom = request.getParameter("nom");
%>
```

*

Les actions

- **jsp:getProperty** permet d'afficher une propriété d'un javaBean

```
<jsp:usebean id="personne" class="testjsp.Personne"
scope="session/>
<jsp:getProperty name="personne" property="nom" />
```

équivaut à

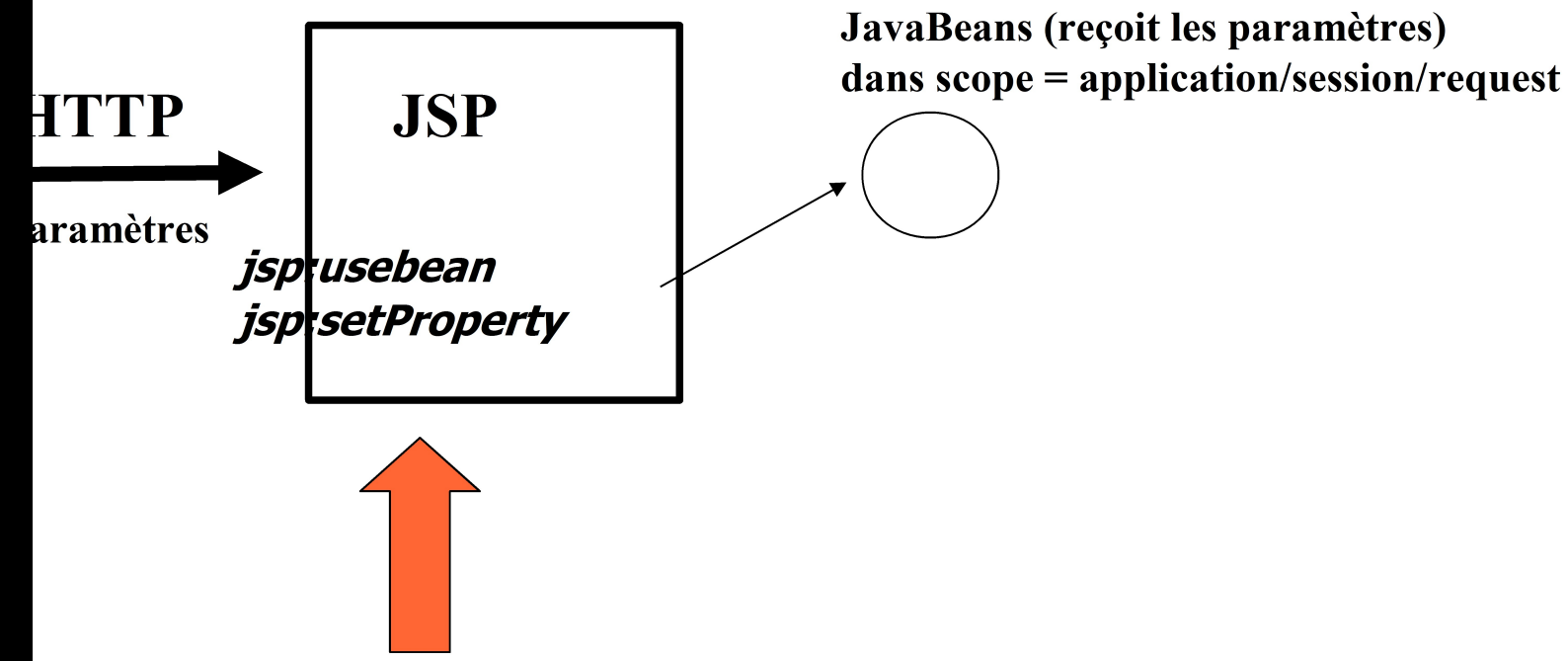
```
<jsp:usebean id="personne" class="testjsp.Personne"
scope="session/>
<%=personne.getNom() %>
```

(peu utile)

Lien HTML/Servlet/JSP

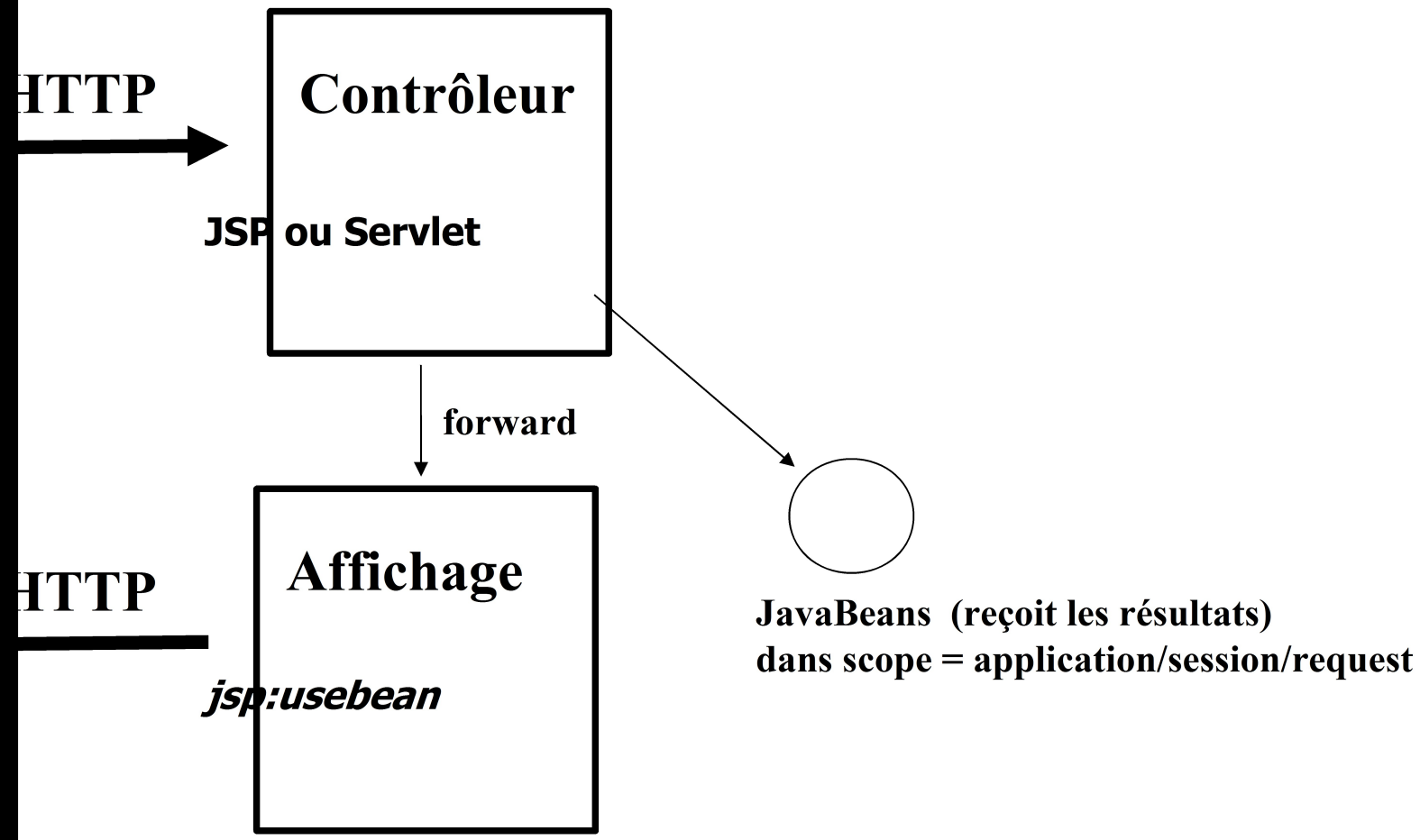
- Une page HTML peut référencer une servlet ou une page JSP
- Une page JSP peut référencer une servlet
  - jsp:include ou jsp:forward
- Une servlet peut référencer une page JSP
  - RequestDispatcher disp = request.getRequestDispatcher("page.jsp");
  - disp.forward(request, response);

JSP – à l'aller

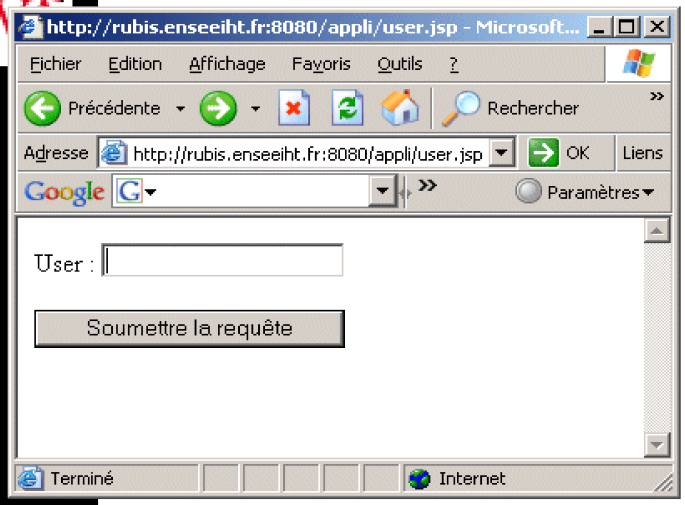


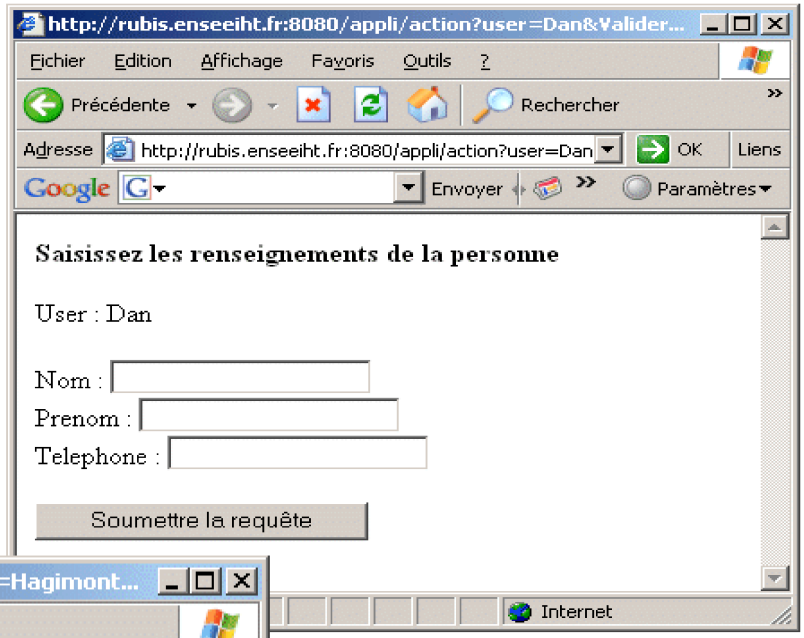
Cette JSP est un contrôleur ...

JSP – au retour



Exemple : un annuaire





Exemple : architecture

- Une page JSP pour chaque écran
  - user.jsp, personne.jsp, listeuser.jsp
- Une servlet qui aiguille les requêtes vers les pages
- Deux objets Java (beans) pour gérer les données
  - Personne.java, ListePersonne.java

Exemple : user.jsp

```
<%@ page language="java" %>
<html>
<body>
<form action="action" method="post">
User : <input type="text" name="user"/><br/><br/>
<input type="submit" name="Valider"/>
<input type="hidden" name="formulaire" value='user'/>
</form>
</body>
</html>
```

Exemple : personne.jsp

```
<%@ page language="java" %>
<html>
<body>
<jsp:useBean id="user" class="mvc.Personne" scope="session"/>
<b> Saisissez les renseignements de la personne</b> <br/><br/>
User : <%= user.user %><br/>
<form action="action" method="post">
Nom : <input type="text" name="nom"/><br/>
Prenom : <input type="text" name="prenom"/><br/>
Telephone : <input type="text" name="telephone"/><br/><br/>
<input type="submit" name="Valider"/>
<input type="hidden" name="formulaire" value="personne"/>
</form>
</body>
</html>
```

Exemple : listeuser.jsp

```
<%@ page language="java" import="java.util.*, mvc.* "%>
<html>
<body>
<jsp:useBean id="listeuser" class= "ListePersonne" scope="application"/>
<b> Liste des users enregistres </b> <br/><br/>
<table border="2">
<th>User</th><th>Nom</th><th>Prenom</th><th>Telephone</th>
```

Exemple : listeuser.jsp

```
<%
Enumeration enu = listeuser.liste.elements();
while (enu.hasMoreElements()) {
Personne personne = (Personne)enu.nextElement();
%>
<tr>
<td><%= personne.user %></td>
<td><%= personne.nom %></td>
<td><%= personne.prenom %></td>
<td><%= personne.telephone %></td>
</tr>
<%  }  %>
</table>
</body>
</html>
```

Exemple : action.java (servlet)

```
public class action extends HttpServlet {

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

    String formulaire = request.getParameter("formulaire");
    HttpSession session = request.getSession();
    ServletContext context = getServletContext();

    if (formulaire == null) {
      RequestDispatcher disp = request.getRequestDispatcher("user.jsp");
      disp.forward(request, response);
    }
```

Exemple : action.java (servlet)

```
if (formulaire.equals("user")) {
      Personne personne = new Personne();
      personne.user = request.getParameter("user");
      session.setAttribute("user", personne);
      ListePersonne listeUser =  (ListePersonne)context.getAttribute("listeuser");
      if (listeUser == null) {
        listeUser = new ListePersonne();
        context.setAttribute("listeuser", listeUser);
       }
      if (listeUser.isRegistered(personne.user)) {
        RequestDispatcher disp = request.getRequestDispatcher("listeuser.jsp");
        disp.forward(request, response);
       } else {
         RequestDispatcher disp = request.getRequestDispatcher("personne.jsp");
         disp.forward(request, response);
    } }
```

Exemple : action.java (servlet)

```
if (formulaire.equals("personne")) {
        Personne personne = (Personne)session.getAttribute("user");
        personne.nom = request.getParameter("nom");
        personne.prenom = request.getParameter("prenom");
        personne.telephone = request.getParameter("telephone");

        ListePersonne listeUser =  (ListePersonne)context.getAttribute("listeuser");
       listeUser.liste.put(personne.user, personne);

        RequestDispatcher disp = request.getRequestDispatcher("listeuser.jsp");
        disp.forward(request, response);
    }
  }
}
```

Exemple : les beans

```
public class Personne {
public String user, nom, prenom, telephone;
}

public class ListePersonne {
public Hashtable liste = new Hashtable();

public boolean isRegistered(String user) {
Enumeration enu = liste.elements();
while (enu.hasMoreElements()) {
Personne personne = (Personne)enu.nextElement();
if (personne.user.equals(user)) return true;
}
return false;
}
}
```

Bilan

- Présentation
  - Sous forme de pages HTML
  - Programmation en Java dans les pages
- Code métier
  - Sous forme de servlet
  - Échange de données avec les pages sous forme de javaBeans
- Séparation (modèle MVC) claire entre
  - Présentation (page JSP)
  - Contrôle (servlet)
  - Métier (programmes Java)
- Le traitement des données récupérées de la BD peuvent être lourds (d'où les EJB)