

JAVASERVER PAGES

INTRODUCTION

- Comme les servlets, servent à créer du contenu Web de manière dynamique.
- Les JSP sont des documents de type texte, contenant du code HTML ainsi que des scriptlets (et/ou des expressions).
- Ces pages étant basées sur du code HTML ou XML, elles peuvent être créées et manipulées par du personnel non technique.

INTRODUCTION (servlet/jsp)

timeServlet.java

```
package jweb.servlet;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import java.io.*;

public class TimeServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
        HttpServletResponse rep)
        throws ServletException, IOException
    {
        rep.setContentType("text/html");
        PrintWriter out = rep.getWriter() ;
        out.println("<HTML><HEAD> ");
        out.println("<TITLE >Affichage date
        </TITLE></HEAD><BODY>");
        out.println("<P class=titre>"+new
        Date()+"</P>");
        out.println("</BODY></HTML>");
    }
}
```



time.jsp

```
<%@page import="java.util.*" %>
<HTML>
  <HEAD>
    <TITLE>Affichage date </TITLE>
  </HEAD>
  <BODY>
    <P class=titre>
      <%= new Date()%>
    </P>
  </BODY></HTML>
```

INTRODUCTION (jsp)

```
<html>
<head>
<title>Enregistrement des coordonnées</title>
</head>
<body bgcolor="orange" text="green">
<h2>Enregistrement des coordonnées effectué</h2>
<hr width="75%">
<p><b>Bonjour
  <%= request.getParameter("titre") %>
  <%= request.getParameter("prenom") %>
  <%= request.getParameter("nom") %>
  <%=
    int âge = Integer.parseInt(request.getParameter("age"));
    String message = "Vous êtes un";
    if (âge>0 && âge<12) message += " enfant.";
    if (âge>=12 && âge<18) message += " adolescent.";
    if (âge>=18 && âge<60) message += " adulte.";
    if (âge>=60) message += "e personne du troisième âge.";
  <%=
  <p><%= message %></b>
</body>
</html>
```

Expressions

Scriptlets

INTRODUCTION

- Les pages JSP s'exécutent sous la forme de servlets.
Donc disposent du même cycle de vie.
- Elles disposent du même support pour la gestion des sessions.
- Elles peuvent également charger des JavaBeans et appeler leurs méthodes
- Accéder à des sources de données se trouvant sur des serveurs distants
- Effectuer des calculs complexes.

INTRODUCTION

- La démarche de la plateforme J2EE, les servlets serviront plus à traiter les requêtes des clients alors que les pages JSP serviront plus à la présentation.
- Dans ce Chapitre, nous allons exploiter toutes les possibilités des pages JSP.

LES ÉLÉMENTS JSP

- Nous ne pouvons pas écrire du code Java n'importe où dans une page HTML.
- Nous avons besoin d'un moyen pour indiquer au serveur où s'arrête le code HTML et où commence le code Java.
- Pour cela la spécification JSP définit des balises pour délimiter le code Java.
- Ces balises permettent de définir trois catégories d'éléments :
 1. les directives ;
 2. les scripts ;
 3. les actions.

LES DIRECTIVES

- Les directives sont des éléments fournissant au conteneur des informations relatives à la page.

Ils existent :

1. page :

<%@ page attributs %>

ou en format XML

<jsp:directive.page attributs />

2. include :

<%@ include file = ".." %>

ou en format XML

<jsp:directive.include file = ".." />

Directive	Attribut	Description
page	import	Ex: <code><%@ page import = "java.io.*, java.util.*" %></code>
	session	True ou false indique que la page fait partie d'une session. La valeur par défaut est true .
	isThreadSafe	Indique si la page peut être employée pour des accès simultanés. La valeur par défaut est true .
	info	La valeur de cet attribut peut être une chaîne quelconque décrivant le contenu de la page, ou sa fonction, son auteur, etc.
	errorPage	Indique l'URL de la page qui doit être renvoyée au client en cas d'erreur.
	isErrorPage	Cet attribut indique si la page est une page d'erreur. Sa valeur par défaut est false .
	contentType	Définit le type de contenu de la page. Par défaut text/html
	pageEncoding	Le jeu de caractères utilisés pour la page. La valeur par défaut est ISO-8859-1
include	file	Le nom du fichier à inclure à l'emplacement de la balise. Il peut s'agir d'une page HTML ou JSP, ou du fragment de page.

LES DIRECTIVES

- Une page JSP peut contenir plusieurs directives page.
- la directive **include** est employée pour inclure une autre page, ou un fragment de page, dans la page JSP.
- Il peut s'agir d'un **en-tête** ou d'un **pied de page**.
- Cette directive est utile chaque fois qu'un contenu standard doit être **réutilisé** dans plusieurs pages.
- L'inclusion a lieu **avant** la traduction de la page en code Java.

LES SCRIPTS

- Les éléments de script permettent de placer du code java dans les pages JSP. Il en existe trois formes :

1. les déclarations :

<%! déclaration %> ou en format XML

<jsp:declaration>déclaration</jsp:declaration>

2. les scriptlets :

<% fragment de code %> ou en format XML

<jsp:scriptlet>fragment de code</jsp:scriptlet>

3. les expressions :

<%= expression %> ou en format XML

<jsp:expression>expression</jsp:expression>

LES DÉCLARATIONS

- Une déclaration doit être employée pour déclarer
- pour initialiser un attribut
- pour déclarer une méthode
- Par exemple,

```
<%! Vector v = new Vector( ) ; %>  
ou  
<jsp:declaration>Vector v = new Vector( ) ;</jsp:declaration>
```

- Nous pouvons également définir des méthodes qui seront utilisées ensuite dans l'ensemble de la page

```
<jsp:declaration>  
    public int nombreMots(String chaîne) {  
        return new StringTokenizer(chaîne).countTokens();  
    }  
</jsp:declaration>
```

- Les attributs ou les méthodes ainsi déclarées peuvent être appelées par n'importe quel code (scriptlets) présent dans toute la page

LES SCRIPTLETS

- Les scriptlets contiennent des instructions Java.
- Les scriptlets peuvent contenir n'importe quel code Java valide.

- Par exemple,

```
<%  
    for (int i=0; i<10; i++) {  
%>  
Bonjour !  
%>  
    }  
%>
```

- Tout ce qui se trouve entre les délimiteurs de scriptlets (**<% et %>**) est du code Java.
- Ce qui se trouve à l'extérieur est le contenu renvoyé au client.
- Scriptlet valide

```
<%  
    Vector v = new Vector( );  
    for (int i=0; i<10; i++)  
        v[i] = i;  
%>
```

LES SCRIPTLETS

- la différence entre les **scriptlets** et les **déclarations**
 1. **Les scriptlets** ne peuvent être employées pour définir des méthodes. Seules **les déclarations** permettent cela.
 2. Les variables déclarées dans une **déclaration** sont des attributs, accessible dans toutes **les scriptlets** de la page.
 3. Les variables déclarées dans une **scriptlet** sont des **variables locales** et ne sont donc visibles qu'à l'intérieur de la scriptlet dans laquelle elles sont définies.

LES EXPRESSIONS

- Les expressions sont utilisées pour renvoyer directement au client
 1. la valeur d'une variable
 2. la valeur retour d'une méthode.
- Exemple

```
Le nombre d'éléments dans cette phrase est  
<jsp:expression >  
    nombreMots("Le nombre d'éléments de cette phrase est n")  
</jsp:expression >
```

```
Le nombre d'éléments dans cette phrase est <%= nombreMots("Le nombre d'éléments dans cette phrase est n") %>
```

- les déclarations et les scriptlets contiennent des lignes de code Java et doivent donc être systématiquement terminées par des points-virgules. En revanche, les expressions ne doivent pas en comporter.

LES COMMENTAIRES

- Il est possible d'utiliser des commentaires HTML dans les pages JSP.
- Ces commentaires apparaissent dans la page renvoyée au client

```
<!-- Ce commentaire sera transmis au client. -->
```

- Il existe également des commentaires JSP :

```
<%-- Ce commentaire Ne sera PAS transmis au navigateur client. --%>
```


Example

The hello.jsp Page

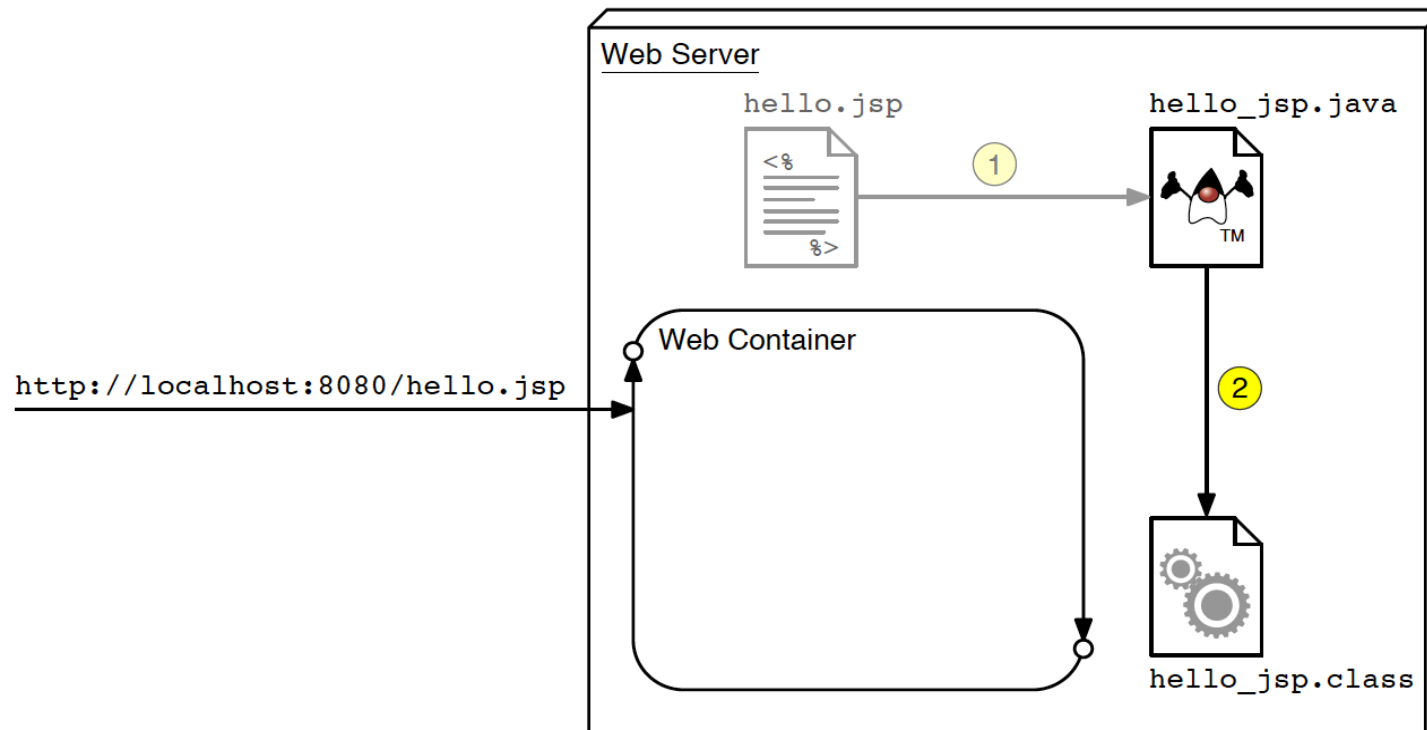
```
<%! private static final String DEFAULT_NAME = "World"; %>

<html>

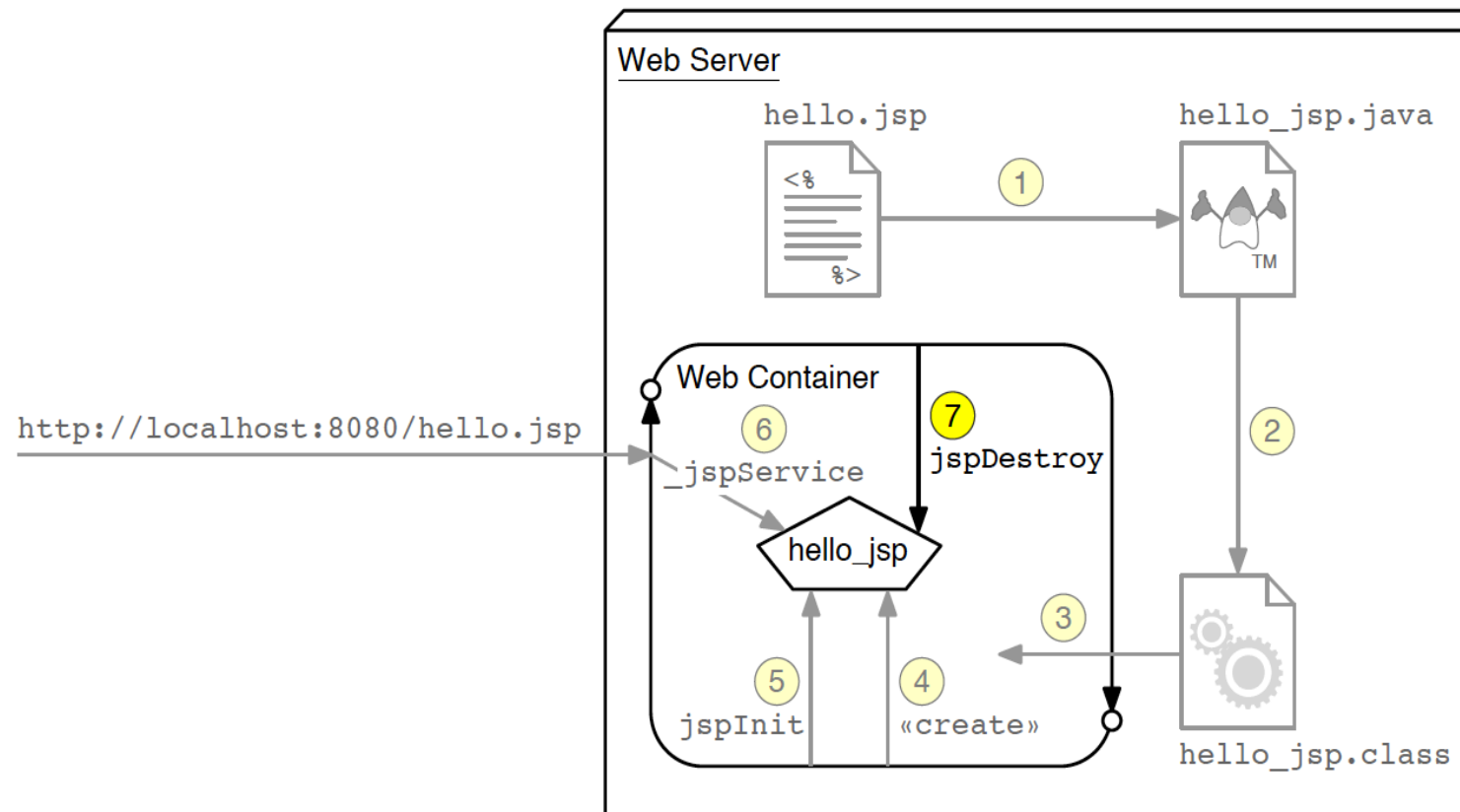
<head>
<title>Hello JavaServer Page</title>
</head>

<!-- Determine the specified name (or use default) --%>
<%
)
String name = request.getParameter("name");
:   if ( (name == null) || (name.length() == 0) ) {
;       name = DEFAULT_NAME;
:   }
; %>
;
<body bgcolor='white'>
;
) <b>Hello, <%= name %></b>
)
```

JSP Page Compilation



Cycle de vie jsp



EXEMPLE SUR LES DIRECTIVES ET LES SCRIPTS

- Nous allons mettre en œuvre une application Web qui permet de gérer une petite messagerie
- Voici, ci-dessous la page d'accueil d'un tel site :

The screenshot shows a Mozilla Firefox browser window with the following details:

- Browser title: Messages - Mozilla Firefox
- Address bar: http://localhost:8080/message/
- Page title: Messages
- Page content: A yellow background with the word "Messages" in large black font. Below it is a table with two columns: "Sujets" and "Message".

Sujets	Message
bienvenue	bienvenue dans le site d apprentissage
DotNet	cour sur C# general
java	cour sur les JSP

samedi 14 novembre 2009

BASE DE DONNÉES

The first screenshot shows the MySQL Table Editor for the 'personne' table. The table has four columns: 'idPersonne' (INTEGER, UNSIGNED, NOT NULL, AUTO INCREMENT), 'nom' (VARCHAR(15), NOT NULL), 'prénom' (VARCHAR(15), NOT NULL), and 'motDePasse' (VARCHAR(15), NOT NULL).

Column Name	Datatype	NOT NULL	AUTO INCR	Flags	Default Value	Comment
idPersonne	INTEGER	✓	✓	UNSigned, ZEROFILL		
nom	VARCHAR(15)	✓		BINARY		
prénom	VARCHAR(15)	✓		BINARY		
motDePasse	VARCHAR(15)	✓		BINARY		

The second screenshot shows the MySQL Table Editor for the 'message' table. The table has four columns: 'idMessage' (INTEGER, UNSIGNED, NOT NULL, AUTO INCREMENT), 'idPersonne' (INTEGER, UNSIGNED, NOT NULL), 'sujet' (VARCHAR(25), NOT NULL), and 'texte' (TEXT, NOT NULL). The 'idPersonne' column has a default value of 0.

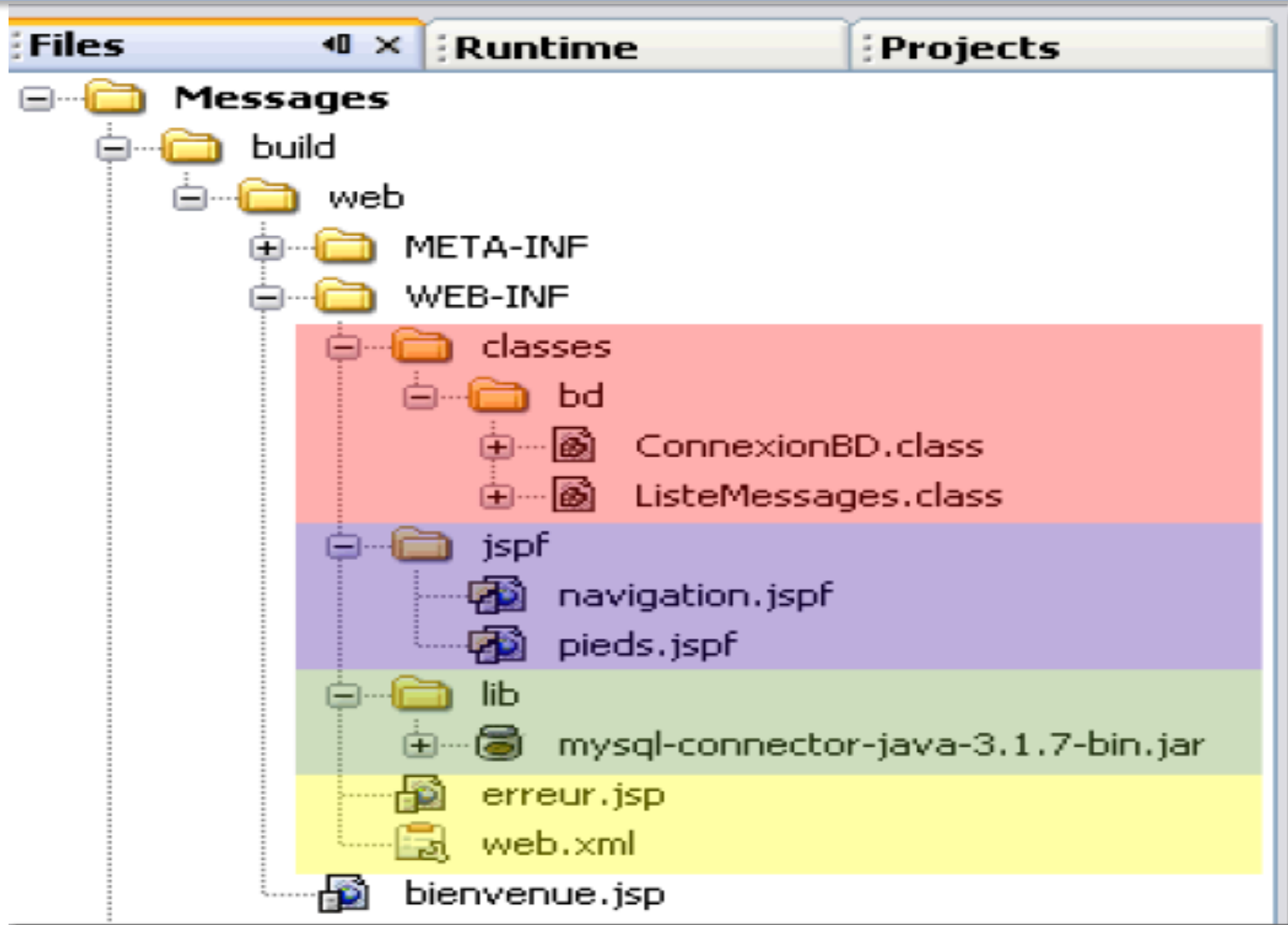
Column Name	Datatype	NOT NULL	AUTO INCR	Flags	Default Value	Comment
idMessage	INTEGER	✓	✓	UNSigned, ZEROFILL		
idPersonne	INTEGER	✓		UNSigned, ZEROFILL	0	
sujet	VARCHAR(25)	✓		BINARY		
texte	TEXT	✓				

The screenshot shows the MySQL Query Browser interface. The query executed is `SELECT * FROM `message` m`. The result set contains three rows of data.

idMessage	idPersonne	sujet	texte
1	1	Bienvenue	Bienvenue sur le site de gestion de messages, notamment pour fabriquer...
2	1	Java	Apprentissage du Langage Java, étude sur les JSP
3	1	C++	Cours et TP en toute simplicité

The Schemata panel on the right shows the database structure, including the 'gestion' and 'messagerie' databases, with 'message' and 'personne' tables listed under 'messagerie'.

ARCHITECTURE DE L'APPLICATION WEB



ARCHITECTURE DE L'APPLICATION WEB

Dossiers		Type de fichier
webapp		<p><i>ressources Web publiques (pages accessibles directement)</i></p> <p>Pages statiques : <code><*.html></code> Pages dynamiques JSP : <code><*.jsp></code> Applets : <code><*.class></code></p> <p>ici <code>bienvenue.jsp</code></p>
WEB-INF		<p><i>ressources Web privée (pages non accessibles directement)</i></p> <p>Fichier de configuration de l'application WEB : <code><web.xml></code> Pages d'erreur : <code><*.jsp></code> Autres pages JSP : <code><*.jsp></code></p> <p>ici <code>web.xml</code> et <code>erreur.jsp</code></p>
	classes	<p>Emplacement des classes utilitaires et des JavaBeans (compilées): <code><*.class></code>.</p> <p>ici <code>bd.ConnexionBD.class</code> et <code>bd.ListeMessages.class</code></p>
	jspf	<p>Fragment de pages : <code><*.jspx></code></p> <p>ici <code>navigation.jspf</code> et <code>pieds.jspf</code></p>
	lib	<p>Bibliothèques <code><*.jar></code> non standards comme les drivers JBDC.</p> <p>ici <code>mysql-connector_java_3.1.7-bin.jar</code></p>
	tlds	Bibliothèques de balises

ARCHITECTURE DE L'APPLICATION WEB

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">

  <display-name>Liste des messages personnels</display-name>
  <welcome-file-list>
    <welcome-file>bienvenue.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```


*.JSPF

pieds.jspf

```
<%@page import="java.util.Date, java.text.DateFormat" %>

<%!
    DateFormat formatDate = DateFormat.getDateInstance(DateFormat.FULL);
%>

    <br><hr>
    <h4 align="right"><%= formatDate.format(new Date()) %></h4>
</font>
</body>
</html>
```

Résultat

lundi 7 novembre 2005

Terminé

*.JSPF

navigation.jspf

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
<head><title>Messages</title></head>
<body bgcolor="#FFFF66">
  <font face="Arial">
    <h2 align="center">Messages</h2>
    <hr>
    <table bgcolor="1" cellpadding="3" cellspacing="2" width="90%" align="center">
      <tr bgcolor="#FF9900">
        <th align="left"><a href="bienvenue.jsp">Sujets</th>
        <th align="right">
          <a href="#">Identification</a>
          <a href="nouvelutilisateur.jsp">Inscription</a>
        </th>
      </tr>
    </table>
  </font>
</body>
</html>
```

Messages

[Sujets](#)

[Identification](#) [Inscription](#)

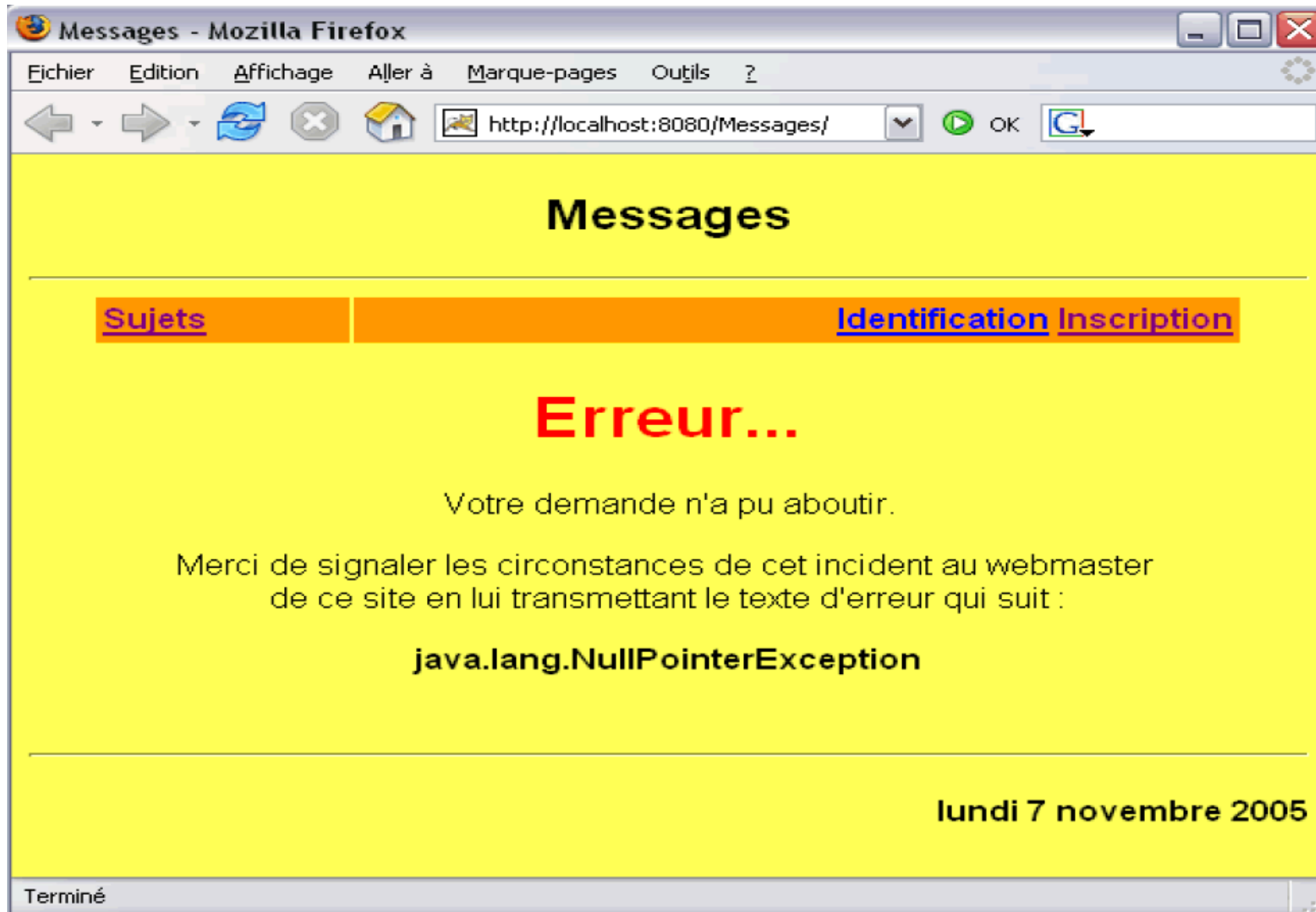
BIENVENUE.JSP

```
<%@ page errorPage = "/WEB-INF/erreur.jsp" import="bd.*" %>
<%@ include file = "/WEB-INF/jspf/navigation.jspf" %>
<font face="Arial">
<p><table border="1" cellpadding="3" cellspacing="2" width="90%" align="center">

  <tr bgcolor="#FF6600">
    <th>Sujet</th>
    <th>Message</th>
  </tr>
  <%
    ListeMessages listeMessages = new ListeMessages(1);
    int ligne = 0;
    while (listeMessages.suivant()) {
  <%
  <tr bgcolor="<%= ligne++ % 2 == 0 ? "#FFFF66" : "#FFCC00" %>">
    <td><b><%= listeMessages.sujet() %></b></td>
    <td><%= listeMessages.texte() %></td>
  </tr>
  <%
    }
    listeMessages.arrêt();
  <%
  </table></p>
</font>

<%@ include file = "/WEB-INF/jspf/pieds.jspf" %>
```

erreur.JSP



erreur.JSP

```
erreur.jsp

<%@page isErrorPage="true"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

<%@include file="/WEB-INF/jspf/navigation.jspf"%>

<center>
<h1><font color="red">Erreur...</font></h1>
<p>Votre demande n'a pu aboutir.</p>
<p>Merci de signaler les circonstances de cet incident au webmaster
<br>de ce site en lui transmettant le texte d'erreur qui suit :</p>
<p><b><%= exception %></b></p>
</center>

<%@ include file = "/WEB-INF/jspf/pieds.jspf" %>
```

- Une page peut rediriger les erreurs provenant d'une exception vers une autre page (erreur.jsp). Il faut :
- déclarer dans la page **erreur.jsp** qu'elle traite les erreurs

```
<%@page isErrorPage="true"%>
```

- déclarer que la page ne le fait pas et préciser le nom de la page d'erreur :

```
<%@ page errorPage = "/WEB-INF/erreur.jsp" %>
```

ERREURS ET EXCEPTIONS

- Sinon le client peut recevoir au travers de son navigateur

Etat HTTP 500 -

type Rapport d'exception

message

description Le serveur a rencontré une erreur interne () qui l'a empêché de satisfaire la requête.

exception

```
javax.servlet.ServletException
  Formulaire.init (Formulaire.java:23)
  javax.servlet.GenericServlet.init (GenericServlet.java:211)
  org.apache.catalina.valves.ErrorReportValve.invoke (ErrorReportValve.java:118)
  org.apache.coyote.tomcat5.CoyoteAdapter.service (CoyoteAdapter.java:160)
  org.apache.coyote.http11.Http11Processor.process (Http11Processor.java:799)
  org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.processConnection (Http11Protocol.java:705)
  org.apache.tomcat.util.net.TcpWorkerThread.runIt (PoolTcpEndpoint.java:577)
  org.apache.tomcat.util.threads.ThreadPool$ControlRunnable.run (ThreadPool.java:683)
  java.lang.Thread.run (Thread.java:595)
```

note La trace complète de la cause mère de cette erreur est disponible dans les fichiers journaux de Apache Tomcat/5.0.28.

LES ACTIONS

- Egalemeht appelées actions standards.
- Les actions standards sont définies par la spécification JSP.
- Il est possible de définir de nouvelles actions et les utiliser dans nos pages JSP.

LES ACTIONS

- La spécification JSP 2.0 définit les actions standards suivantes :

1. `<jsp:useBean>`
2. `<jsp:setProperty>`
3. `<jsp:getProperty>`
4. `<jsp:param>`
5. `<jsp:include>`
6. `<jsp:forward>`
7. `<jsp:plugin>`, `<jsp:params>`, `<jsp:fallback>`
8. `<jsp:attribute>`
9. `<jsp:body>`
10. `<jsp:invoke>` et `<jsp:dobody>`

LES JAVABEANS

- Dans les pages JSP, il est toujours très difficile de lire ce mélange à la fois de code HTML et de code Java.
- De préférable, utiliser une écriture plus proche du HTML en utilisant la syntaxe du XML tout en faisant référence, malgré tout, à des classes Java.
- dans la page Web **.jsp**, nous retrouvons quand même un peu de code Java pour pouvoir utiliser ces classes.
- Les JavaBeans permettent de composer une structure particulière sur ces classes respectant un canevas standard, sans code Java dans la page JSP.

L'ACTION <JSP:USEBEAN>

- Cet élément permet de rendre un `JavaBean` accessible dans la page.
- Un `JavaBean` est simplement une classe Java respectant un certain nombre de conventions.
- Les deux plus importantes sont :
 1. La classe d'un `JavaBean` doit posséder un constructeur sans arguments.
 2. La classe d'un `JavaBean` doit posséder un ensemble de propriétés.

```
private type unePropriété ;  
public type getUnePropriété( ) { return unePropriété; }  
public boolean isUnePropriété( ) { return unePropriétéBooléenne; }  
public void setNom(type unePropriété) { this.unePropriété = unePropriété; ...(reste du code)... }
```

L'ACTION <JSP:USEBEAN>

- L'action <jsp:useBean> prend les paramètres suivants :
- 1. **id** : Le **nom utilisé** pour accéder au bean dans le reste de la page. Il doit être **unique**. Il s'agit en fait du **nom de l'objet** référençant l'instance de la classe du bean donné par le paramètre **class**.
- 2. **scope** : La **portée** du bean. Les valeurs possibles sont **page, request, session et application**. La valeur par défaut est **page**.
- 3. **class** : Le nom de la **classe bean**.
- 4. **type** : Le type de la variable **référençant le bean**.

L'ACTION <JSP:SETPROPERTY>

- Cette action permet de modifier la valeur d'une propriété d'un JavaBean.
- Elle prend les attributs suivant :
 1. name - l'**id du bean**.
 2. property - le **nom de la propriété** à modifier, La valeur peut nommer explicitement une propriété du bean. La valeur peut également être (*).
 3. value - contient la nouvelle **valeur** à affecter à la propriété.
 4. param - le nom du paramètre de la requête contenant la valeur à affecter à la propriété.
- Si les attributs **param** et **value** ne sont pas présents l'action <jsp:setProperty> tente d'utiliser le **paramètre** de la **requête** portant le **même nom** que la **propriété**.

EXEMPLE : <JSP:SETPROPERTY>

```
public class Personne {  
    private String nom;  
    private String prénom;  
    private String motDePasse;  
  
    public Personne() { }  
  
    public String getNom() { return this.nom; }  
    public void setNom(String nom) { this.nom = nom; }  
  
    public String getPrénom() { return this.prénom; }  
    public void setPrénom(String prénom) { this.prénom = prénom; }  
  
    public String getMotDePasse() { return this.motDePasse; }  
    public void setMotDePasse(String motDePasse) { this.motDePasse = motDePasse; }  
}
```

```
<jsp:useBean id = "utilisateur" class = "Personne" />  
<jsp:setProperty name = "utilisateur" property = "nom" value = "REMY" />  
<jsp:setProperty name = "utilisateur" property = "prénom" value = "<%= request.getParameter("prénom") %>" />
```

```
<jsp:useBean id = "utilisateur" class = "Personne" />  
...  
<jsp:setProperty name = "utilisateur" property = "prénom" param = "prénom" />
```

L'ACTION <JSP:GETPROPERTY>

- Cette action permet de lire la valeur d'une propriété d'un JavaBean.
- Elle possède les attributs suivant :
 1. **name** - l'id du bean
 2. **property** - le **nom** de la propriété à lire.
- Les attributs **name** et **property** sont toujours **requis**.
- La **valeur** de la **propriété** est incluse dans la **réponse**

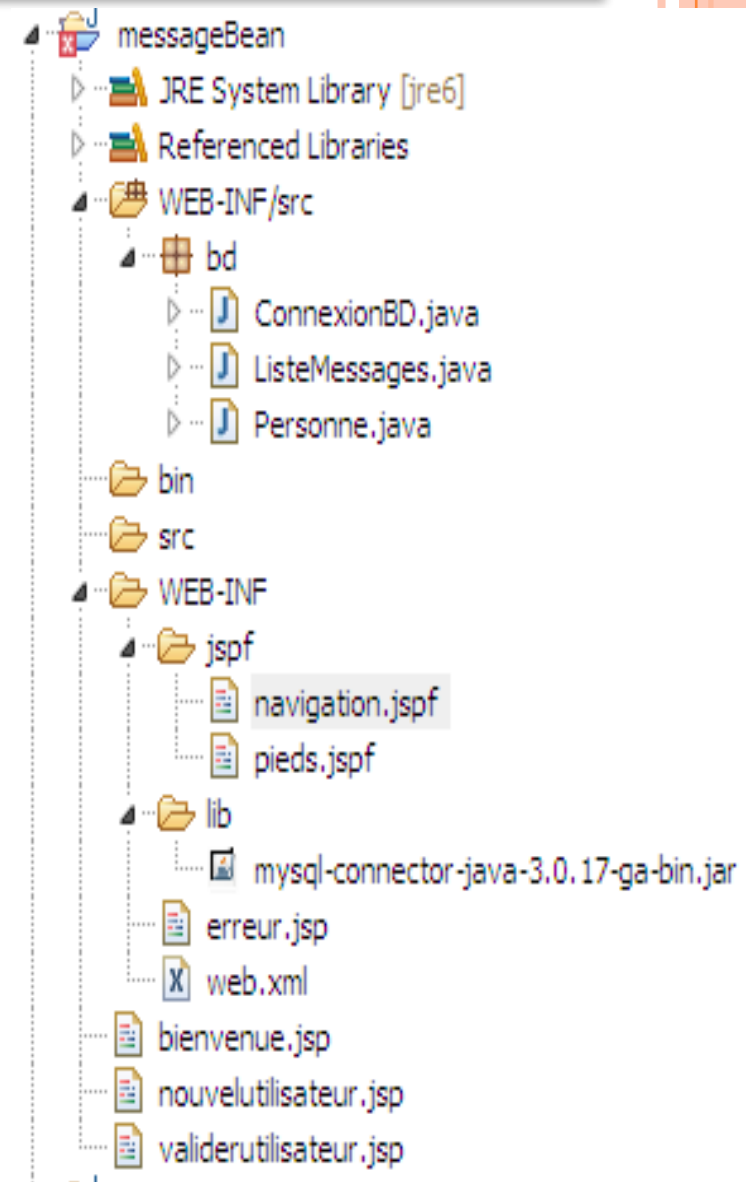
L'ACTION <JSP:GETPROPERTY>

```
L'utilisateur a pour nom  
<jsp:getProperty name = "utilisateur" property = "nom" />  
et pour prénom  
<jsp:getProperty name= "utilisateur" property = "prénom" />
```

- Lorsque la page JSP est traduite en code Java, cette action est remplacée par un appel aux méthodes `getNom()` et `getPrénom()`.

EXEMPLE

- Le javaBean **Personne** qui **récupère** les informations saisies par l'utilisateur afin de les **enregistrer** dans la **base de données** avec une mise en majuscule adaptée
- La page JSP <nouvelutilisateur.jsp> qui s'occupe du formulaire de **saisie**.
- La page JSP <validerutilisateur.jsp> qui **récupère** les informations issues du **formulaire**, se met ensuite en relation avec le **JavaBean** **Personne** et **affiche le résultat** suivant le comportement du JavaBean.



EXEMPLE

nouvelutilisateur.jsp

```
<%@ page errorPage = "/WEB-INF/erreur.jsp"%>
<%@ include file = "/WEB-INF/jspf/navigation.jspf" %>

<h3 align="center">Demande d'inscription</h3>

<form action="validerutilisateur.jsp" method="post">
  <p><table border="1" cellpadding="3" cellspacing="2" width="90%" align="center">
    <tr>
      <td bgcolor="#FF9900" width="100"><b>Nom</b></td>
      <td><input type="text" name="nom"></td>
    </tr>
    <tr>
      <td bgcolor="#FF9900" width="100"><b>Prénom</b></td>
      <td><input type="text" name="prénom"></td>
    </tr>
    <tr>
      <td bgcolor="#FF9900" width="100"><b>Mot de passe</b></td>
      <td><input type="password" name="motDePasse"></td>
    </tr>
  </table></p>
  <p align="center"><input type="submit" value="Nouvel utilisateur"></p>
</form>

<%@ include file = "/WEB-INF/jspf/pieds.jspf" %>
```

```
1 <%@ page errorPage = "/WEB-INF/erreur.jsp" import="bd.*" %>
2 <%@ include file = "/WEB-INF/jspf/navigation.jspf" %>
3
4 <h3 align="center">Confirmation de votre demande d'inscription</h3>
5
6 <jsp:useBean id="utilisateur" class="bd.Personne">
7     <jsp:setProperty name="utilisateur" property="*" />
8
9     <p><table border="1" cellpadding="3" cellspacing="2" width="90%" align="center">
10         <tr>
11             <td bgcolor="#FF9900" width="100"><b>Nom</b></td>
12             <td><jsp:getProperty name="utilisateur" property="nom" /></td>
13         </tr>
14         <tr>
15             <td bgcolor="#FF9900" width="100"><b>Prénom</b></td>
16             <td><jsp:getProperty name="utilisateur" property="prénom" /></td>
17         </tr>
18         <tr>
19             <td bgcolor="#FF9900" width="100"><b>Mot de passe</b></td>
20             <td><jsp:getProperty name="utilisateur" property="motDePasse" /></td>
21         </tr>
22     </table></p>
23     <h3 align="center">
24     <% if (!utilisateur.enregistrer()) { %>
25         <font color="red">ATTENTION : Utilisateur déjà enregistré</font>
26     <%
27         }
28         else {
29     %>
30         <font color="green">Nouvel utilisateur enregistré</font>
31     <%
32         }
33         utilisateur.arrêt();
34     %>
35     </h3>
36 </jsp:useBean>
37
38 <%@ include file = "/WEB-INF/jspf/pieds.jspf" %>
```

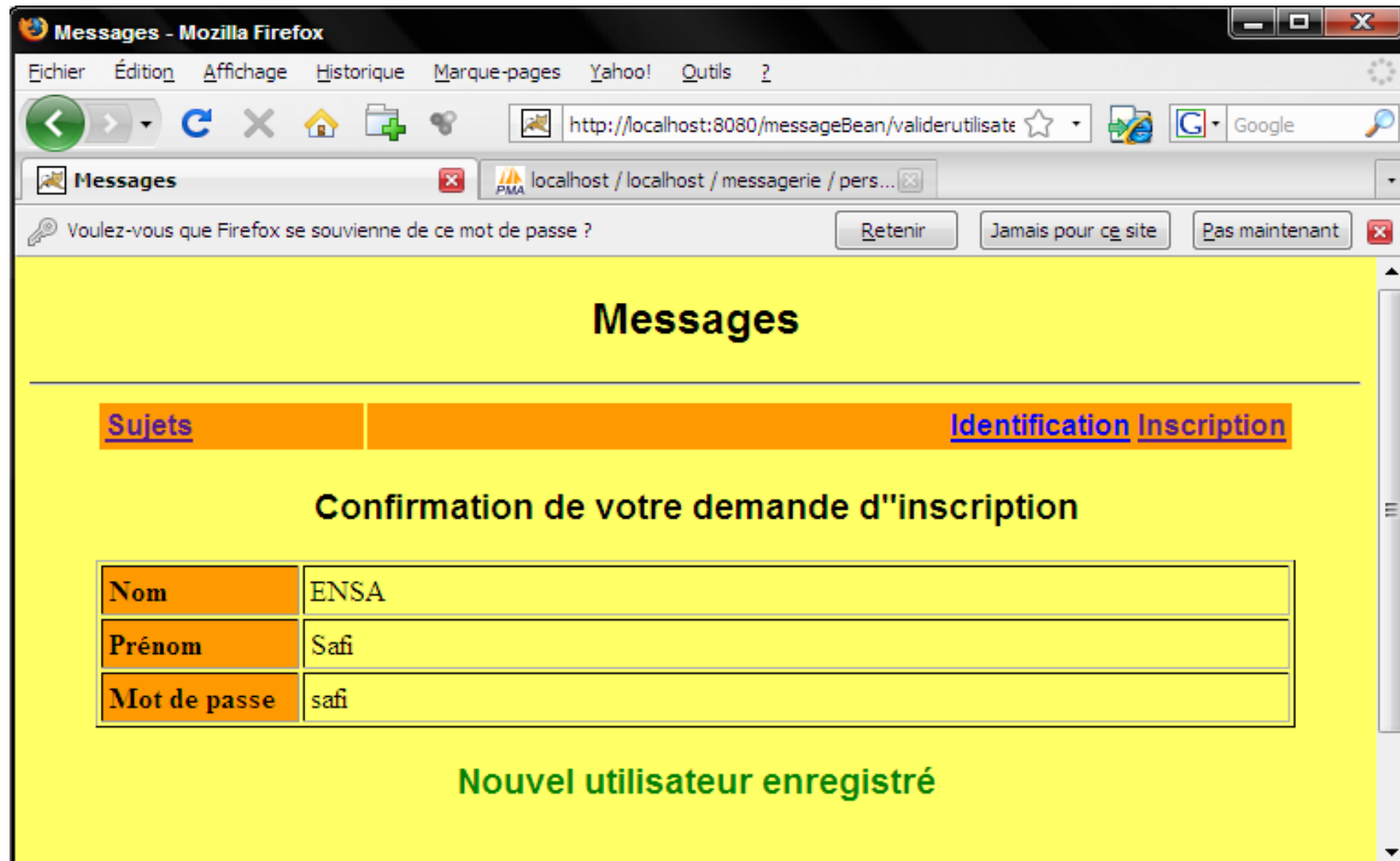
EXEMPLE

The screenshot shows a Mozilla Firefox browser window with the following elements:

- Browser title: Messages - Mozilla Firefox
- Address bar: http://localhost:8080/messageBean/fr
- Page title: Messages
- Page content: A registration form titled "Demande d'inscription" with a yellow background. The form includes three input fields: "Nom" (ensa), "Prénom" (safi), and "Mot de passe" (masked with dots). A "Nouvel utilisateur" button is located below the form.

Sujets	Identification	Inscription
Demande d'inscription		
Nom	<input type="text" value="ensa"/>	
Prénom	<input type="text" value="safi"/>	
Mot de passe	<input type="password" value="••••"/>	
<input type="button" value="Nouvel utilisateur"/>		

EXEMPLE



The screenshot shows a Mozilla Firefox browser window with the following details:

- Browser title: Messages - Mozilla Firefox
- Address bar: http://localhost:8080/messageBean/validerutilisateur
- Page title: Messages
- Page content: Confirmation de votre demande d'inscription
- Registration details table:

Sujets	Identification	Inscription
Confirmation de votre demande d'inscription		
Nom	ENSA	
Prénom	Safi	
Mot de passe	safi	

Nouvel utilisateur enregistré

EXEMPLE

The screenshot shows a Mozilla Firefox browser window with the following details:

- Browser title: Messages - Mozilla Firefox
- Address bar: http://localhost:8080/messageBean/validerutilise
- Page title: Messages
- Page content: Confirmation de votre demande d'inscription
- Warning message: ATTENTION : Utilisateur déjà enregistré

The page features a yellow background and a navigation bar with links: [Sujets](#), [Identification](#), and [Inscription](#).

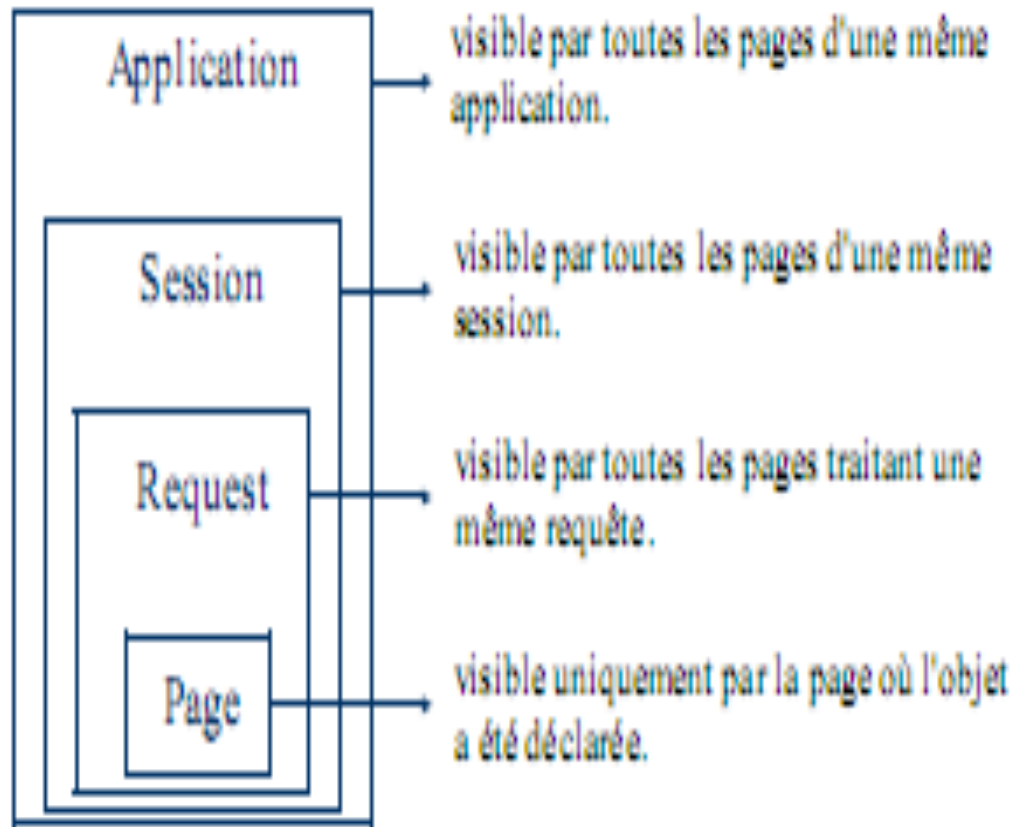
Nom	ENSA
Prénom	Safi
Mot de passe	safi

LES OBJETS IMPLICITES

- Avant de continuer sur la suite des actions, nous allons passer par la connaissance des objets implicites.
- Comme les pages JSP sont finalement **des servlets**, il est normal de retrouver les **mêmes objets**
- Ces objets, nous pouvons y **accéder sans jamais avoir à les déclarer** ou à les **initialiser**.
- Sont accessibles dans : **scriptlets** et **expressions**.
- Voici la liste des objets implicites :
 1. request
 2. response
 3. out
 4. session
 5. config
 6. exception
 7. application
 - **request** : représente l'objet `HttpServletRequest`
 - **response** : représente l'objet `HttpServletResponse`
 - **out** : représente le flux de sortie de la réponse
 - **config** : représente l'objet `ServletConfig` passé à l'initialisation
 - **application** : représente l'objet `ServletContext`
 - **exception** : représente une exception qui n'a pas été interceptée

LES DIFFÉRENTES PORTÉES

- La portée d'un objet est précisée par l'attribut scope



EXEMPLE SUR LES OBJETS IMPLICITES

Voir l'Exemple en TP.

INCLUSION DE PAGES ET TRANSMISSION DE REQUÊTES

- Les pages JSP offrent la possibilité d'**inclure** d'autres **pages** ou **servlets** dans la sortie renvoyée au client
- **Transmettre la requête** à une **autre page** ou à une **servlet**
- Grâce aux actions standards
`<jsp:include>`
Ex: `<jsp:include page=« /includePage.jsp » />`
- et `<jsp:forward>`
Ex: `<jsp:forward page=« suite.jsp » />`

L'ACTION INCLUDE

- Il peut arriver que de nombreuses pages JSP contiennent des fragments semblables, voir identiques
- Par exemple, le haut d'une page Web ou encore un des éléments de la charte graphique.
- Il est alors pratique d'**isoler** ces fragments dans des fichiers séparés et de les **inclure** dans les différentes pages.
- Cette approche facilite la maintenance du site

L'ACTION INCLUDE

- Il existe deux approches:
 1. La **directive** include,
 2. L'**action** include.
- **Directive** - `<% include file = "... " %>` : La ressource est incluse au moment de la traduction de la page en code Java, il s'agit ici d'une inclusion statique.
- **Action** - `<jsp:include page = "... " />` - Contrairement aux directives qui ne servent qu'au moment de la traduction/compilation de page, les actions sont exécutées lors du traitement d'une requête.

L'ACTION INCLUDE : Exemple

- Date.jsp

```
<p>  
  Today's date: <%= (new  
  java.util.Date()).toLocaleString()%>  
</p>
```

- Main.jsp

```
<html>  
<head>  
<title>The include Action Example</title>  
</head>  
<body>  
<center>  
<h2>The include action Example</h2>  
<jsp:include page="date.jsp" flush="true" />  
</center>  
</body>  
</html>
```

L'ACTION FORWARD

- Une requête peut être renvoyée à une autre page JSP, un servlet ou même une simple page HTML :

```
<jsp:forward page="myPage.jsp" />
```

- Tous les objets dont la portée est au moins la requête sont accessibles dans cette autre page.