

P.Mathieu

LP DA2I Lille http://www.iut-a.univ-lille.fr prenom.nom@univ-lille.fr

2 décembre 2018

- Les JSP Beans
- Bean et Formulaires





Ensemble d'actions XML, raccourcis d'écriture

```
jsp :useBean Associe un bean à une JSP
```

```
jsp :setProperty Affecte une propriété d'un bean
```

```
jsp :getProperty Lit une propriété d'un bean
```

```
jsp :include Appelle la page au moment de l'exécution
```

```
<jsp:include page='fichierImporté' />
```

jsp :forward Renvoie la requête à une autre page

```
<jsp:forward page='urlRedirection' />
```

jsp :param permet de passer des paramètres à include ou forward

jsp :plugin Permet de gérer des balises spécifiques au browser

Principe



- Gestion automatique des objets "métier"
- Accès en notation XML, sans avoir à connaitre Java
- Banalise les accès aux objets des différents contextes (page, request, session, application)
- Raccourci d'écriture : la balise useBean s'occupe de rechercher l'objet concerné, de l'instancier s'il n'existe pas et de le ranger au bon endroit pour sa persistance.

Principe



- Constructeur vide, méthodes "public", accesseurs.
- Placé obligatoirement dans un package
- Exemple :

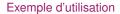
```
package test;
public class Personne
{private String nom;
    public Personne() {}
    public String getNom() {return nom;}
    public void setNom(String s) {nom=s;}
}
```



Paramètres du tag useBean

```
<jsp:useBean id="t" class="test.Personne" scope="sessi
<%= t %>
```

- id: nom de l'instance du Bean dans la Servlet.
- class: nom de la classe définissant le Bean.
- scope: précise la durée de vie du Bean (page (par défaut), request, session, application)
- beanName : nom de fichier pour les beans sérialisés.
- type : type à associer au fichier. permet de "caster" dans le type spécifié





```
// Définition du Bean
package test;

public class Personne
{private String nom;
    public Personne() {}
    public String getNom() {return nom; }
    public void setNom(String s) {nom=s; }
}
```

useBean s'occupe de tout! il fait les déclarations, teste si l'objet existe, le crée le cas échéant, fait les cast nécessaires ... etc

La notation "XML"



- Lire une propriété d'un Bean
 <jsp:getProperty name="t" property="nom" />

Attention: Les méthodes getProperty et setProperty ne fonctionnent qu'avec des String

Yet Another Compteur



```
// Compteur.java
package test;
public class Compteur
{
    private int val=0;
    public String getVal()
    { return "" + val; }
    public void incr() {val++;}
}
```

```
<!-- compteur.jsp -->
<h+m1>
<head>
   <title>Les compteurs à l'aide de Beans</title>
</head>
<body>
<%@ page language="java" errorPage="erreur.jsp" %>
<jsp:useBean id="local" class="test.Compteur"
                         scope="session" />
<jsp:useBean id="global" class="test.Compteur"
                         scope="application" />
<% local.incr(); global.incr(); %>
Vous avez accédé <%= local.getVal() %> fois à cette
page sur les <%= global.getVal() %> accès effectués.
</hody>
</html>
```



Les Beans, en résumé

Quand la balise BEAN est utilisée, c'est la balise "class" qui permet de retrouver la classe, tandis qu'avec une instanciation d'objet "à la main" il faut mettre un import.

```
package tools;
public class MonObjet
   public String toString() {return "all is ok with my object";}
<%@ page contentType="text/html; charset=UTF-8" import="tools.*" %>
<h+m1>
<body>
<h1>Test de mon objet sans balise BEAN</h1>
<% MonObjet m1 = new MonObjet(); %>
<%=m1%>
<h1>Test de mon objet avec balise BEAN</h1>
<isp:useBean id="m2" scope="page" class="tools.MonObjet" />
<%=m2%>
</body>
</html>
```

- Les JSP Beans
- Bean et Formulaires

Principe

- Les formulaires WEB sont très souvent utilisés pour interagir avec l'utilisateur et collecter des données
- JSP fournit grâce aux Beans un dispositif de gestion très pratique

```
<HTML>
<BODY>

FORM METHOD=POST ACTION="Traitement.jsp">

Votre nom ? <INPUT TYPE=TEXT NAME=username SIZE=20><BR>

Votre e-mail ? <INPUT TYPE=TEXT NAME=email SIZE=20><BR>

Votre age ? <INPUT TYPE=TEXT NAME=age SIZE=4>

<P><INPUT TYPE=SUBMIT>

</FORM>

</BODY>

</HTML>
```

Bean associé

- Pour traiter ce formulaire on crée simplement un Bean dont les attributs correspondent exactement aux champs du formulaire
- Setters pour chacun des champs

```
package tools;
public class Personne
{
    String username;
    String email;
    int age;

    public void setUsername( String value ) { username = value; }
    public void setEmail( String value ) { email = value; }
    public void setAge(int value) { age = value; }

    public String getUsername() { return username; }
    public String getEmail() { return email; }
    public int getAge() { return age; }
}
```

Traitement.jsp

```
<jsp:useBean id="p" class="tools.Personne" scope="session"/>
<jsp:setProperty name="p" property="*"/>
<HTML>
<BODY>
<A HREF="NextPage.jsp">Continuer</A>
</BODY>
</HTML>
```

- Tout ce que l'on a à faire c'est utiliser property="*"
- Le tag useBean va automatiquement rechercher une instance de Personne dans la session
- S'il en trouve une, il la met à jour
- S'il n'en trouve pas, il en crée une, l'initialise et la range dans la session
- Le tag setProperty va collecter automatiquement toutes les données en entrée et les palcer dans le Bean!

Les autres pages

```
<jsp:useBean id="p" class="tools.Personne" scope="session"/
<HTML>
<BODY>
Vous êtes<BR>
Nom: <%= p.getUsername() %><BR>
Email: <%= p.getEmail() %><BR>
Age: <%= p.getAge() %><BR>
</BODY>
</HTML>
```