Tutoriel 4: Utilisation des JavaBeans

Etape 0: Création d'un nouveau projet

Si on n'a pas un projet web dynamique, on le crée selon les mêmes étapes vues dans le premier tutoriel, puis on ajoute ce projet au serveur Tomcat.

Etape1: Création d'un bean simple

1) Nous allons créer une classe nomé Personne, dans le package dev. web par exemple.

```
package dev.web;
/**
* Ceci est un bean
 * @author Mohamed
 * /
public class Personne {
      private String nom;
      private String prenom;
      public String getNom() {
            return nom;
      }
      public void setNom(String name) {
            this.nom = name;
      }
      public String getPrenom() {
            return prenom;
      }
      public void setPrenom(String prenom) {
            this.prenom = prenom;
      }
}
```

Etape2: Changement du buid path

Afin de rendre vos objets accessibles à votre application, il faut que les classes compilées à partir de vos fichiers sources soient placées dans un dossier "classes", lui-même placé sous le répertoire /WEB-INF.

1) Bouton droit sur le projet, puis Build Path, puis Configure Build Path...



2) Sélectionner alors l'onglet source, puis regarder en bas le champ *Default output folder*, puis sélectionner le dossier classes qui se trouve dans WEB-INF.

RQ: Il faut d'abord crée le dossier classes, ce dernier devient invisible dans Project Explorer.

۲	Properties for premierProjet	- 🗆 🗙
type filter text	Java Build Path	<> → <> → →
Deployment Assembly A Java Build Path Java Build Path Java Code Style Java Compiler Javado Location Javado Location JavaScript JSP Fragment Project References Refactoring History Run/Debug Settings Server Service Policies Targeted Runtimes Targeted Runtimes Jask Repository Task Tags Validation Web Content Settings Web Page Editor Web Project Settings WikiTeat Jacobet	Source Projects M Libraries Order and Export Source folders on build path:	Add Folder Link Source Edit Remove
	Allow output folders for source folders Default output folder: premierProjet/build/classes	Browse Apply
?	0	Cancel

Etape3: Manipulation du Bean

1) Créer une nouvelle Servlet appelé Servlet5 dans le package dev. web.

2) Remplacer la méthode *diGet()* par le code suivant:

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

```
/* <u>Création</u> du bean */
Personne unBean = new Personne();
```

```
/* Initialisation de ses propriétés */
unBean.setNom( "Ali" );
unBean.setPrenom( "Ahmed" );
/* Stockage du message et du bean dans l'objet request */
request.setAttribute( "attrBean", unBean );
/* Transmission de la paire d'objets request/response à notre JSP */
this.getServletContext().getRequestDispatcher( "/WEB-INF/jsp5.jsp"
).forward( request, response );
}
```

3) Créer dans le répertoire WEB-INF une autre JSP appelée jsp5, qui va réaliser l'affichage des propriétés du bean. Cette JSP contient le code suivant:

```
<%@ page pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
   <head>
       <title>Test</title>
   </head>
   <body>
        Ceci est une page générée depuis une JSP.
        Récupération du bean :
           < %
dev.web.Personne notreBean = (dev.web.Personne )
request.getAttribute("attrBean");
           out.println( notreBean.getPrenom() );
           out.println( notreBean.getNom() );
            8>
       </body>
</html>
```

Voici le résultat attendu:



Ceci est une page générée depuis une JSP.

Récupération du bean : Ahmed Ali