**TP1 EJB : mise en place de l'environnement, premières applications**

**Table des matières**

[1. Installation logicielle](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#1._Installation_logicielle)

[1.1. Vérification de l'installation](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#1.1._V.c3.a9rification_de_l'installation)

[2. Création d'un nouveau projet](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#2._Cr.c3.a9ation_d'un_nouveau_projet)

[3. Création d'un premier ejb de type session bean stateless : HelloWorld](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#3._Cr.c3.a9ation_d'un_premier_ejb_de_type_session_bean_stateless_.3a_HelloWorld)

[4. Ecriture d'une servlet client de test de ce bean](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#4._Ecriture_d'une_servlet_client_de_test_de_ce_bean)

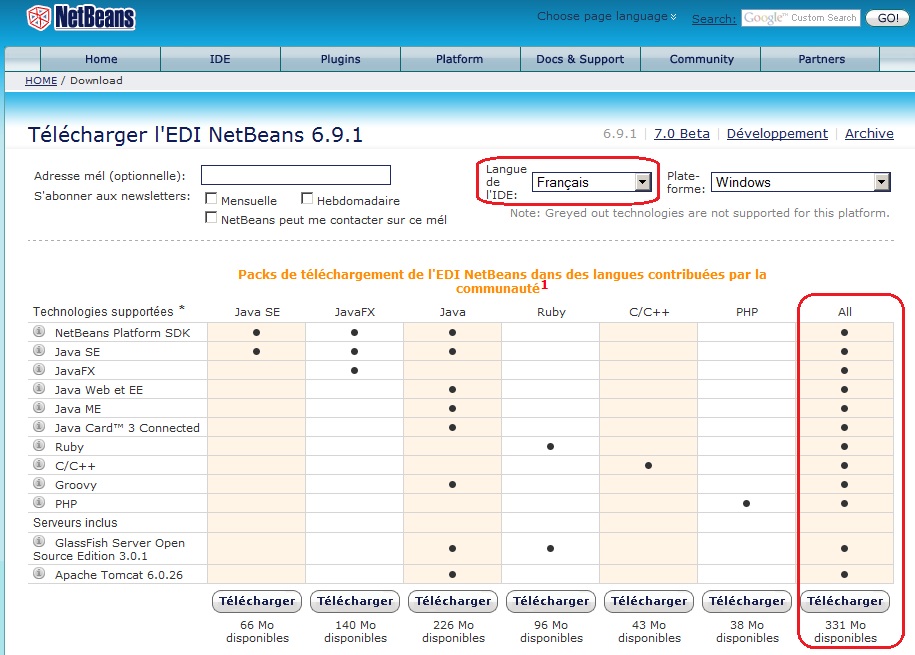
[5. Modifions la page par défaut et insérons un petit menu](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#5._Modifions_la_page_par_d.c3.a9faut_et_ins.c3.a9rons_un_petit_menu)

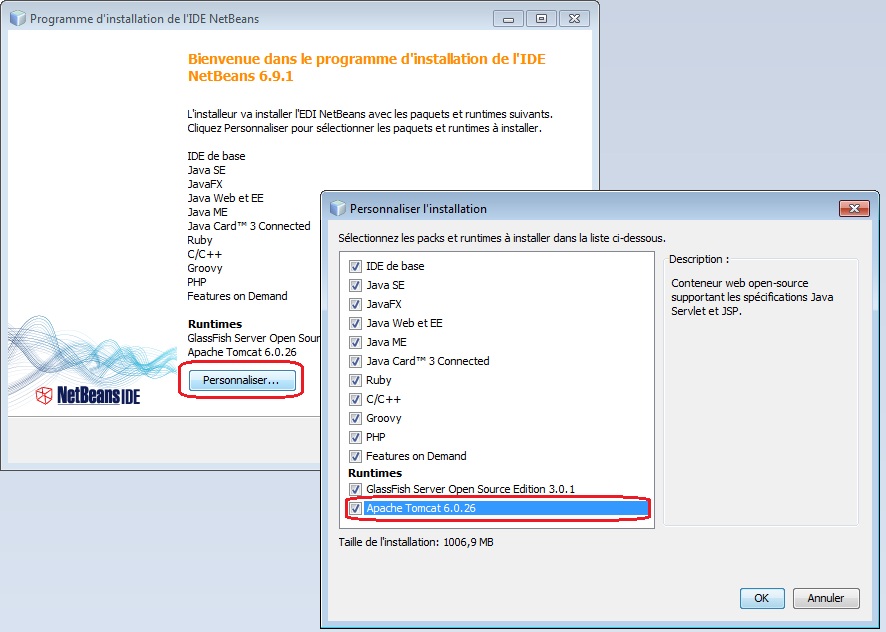
[6. Ecriture d’un second bean appelé par le bean HelloWorld](http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%c3%a9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_EJB_%3a_mise_en_place_de_l'environnement%2c_premi%c3%a8res_applications#6._Ecriture_d.e2.80.99un_second_bean_appel.c3.a9_par_le_bean_HelloWorld)

**1. Installation logicielle**

Logiciels obligatoires :

* **un JDK version 1.6 ou supérieur ET Netbeans 6.9.x**
  + **Le JDK :** On a besoin d'un JDK et non pas d'un JRE.   
    Installez la version la plus récente à partir du [site d'Oracle](http://www.oracle.com/technetwork/java/javase/downloads/index.html) (février 2011 : Java SE 6 update 23)  
    QUESTION : C'est quoi JDK ? C'est quoi JRE ? Pourquoi dans le cadre de ces TP doit on prendre un JDK plutôt qu'un JRE ?
  + **Netbeans 6.9.x** : On a besoin d'une version qui contient les **serveurs glassfish** et **tomcat**.   
    Installez la version (en français si vous préférez) la plus complète sur le [site de Netbeans](http://netbeans.org/downloads/index.html) (dernière colonne à droite).   
    QUESTION : qu'est-ce qu'un IDE ?





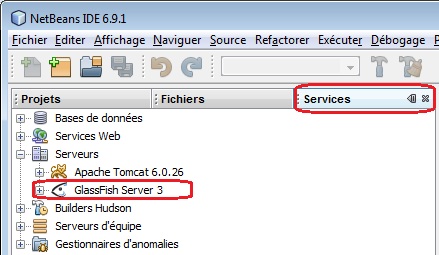
NOTE : les photos d'écrans sont toutes avec Glassfish v3 prelude et avec netbeans 6.7.1, comme nous allons utiliser glassfish v3.1 final et netbeans 6.9, il se peut qu'il y ait de petites différences.

**NE CHANGEZ AUCUN MOT DE PASSE !**   
  
Note : si jamais les liens ci-dessus ont changé, google est votre ami pour récupérer des versions à jour.

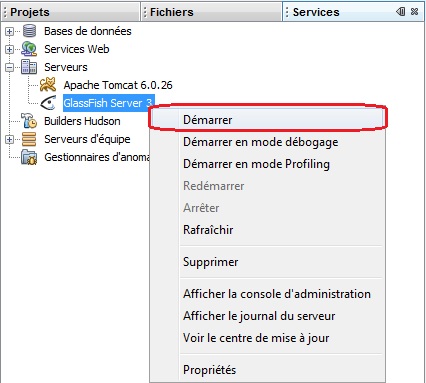
**Installez donc le JDK si vous ne l'avez pas, puis netbeans, et passons à la suite. JE CONSEILLE DE DESINSTALLER LA VERSION PRECEDENTE DE NETBEANS ET DE GLASSFISH SI VOUS AVIEZ DEJA UNE ANCIENNE VERSION !**

**1.1. Vérification de l'installation**

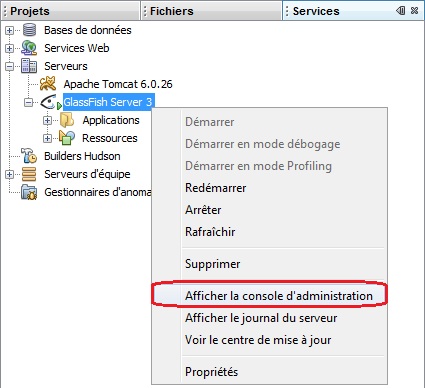
Lancez netbeans une fois l'installation terminée. Cliquez sur l'onglet "services" et ouvrez l'item serveurs. Vous devez voir que tomcat et glassfish sont installés (si vous n'avez pas installé tomcat, ce n'est pas grave) :



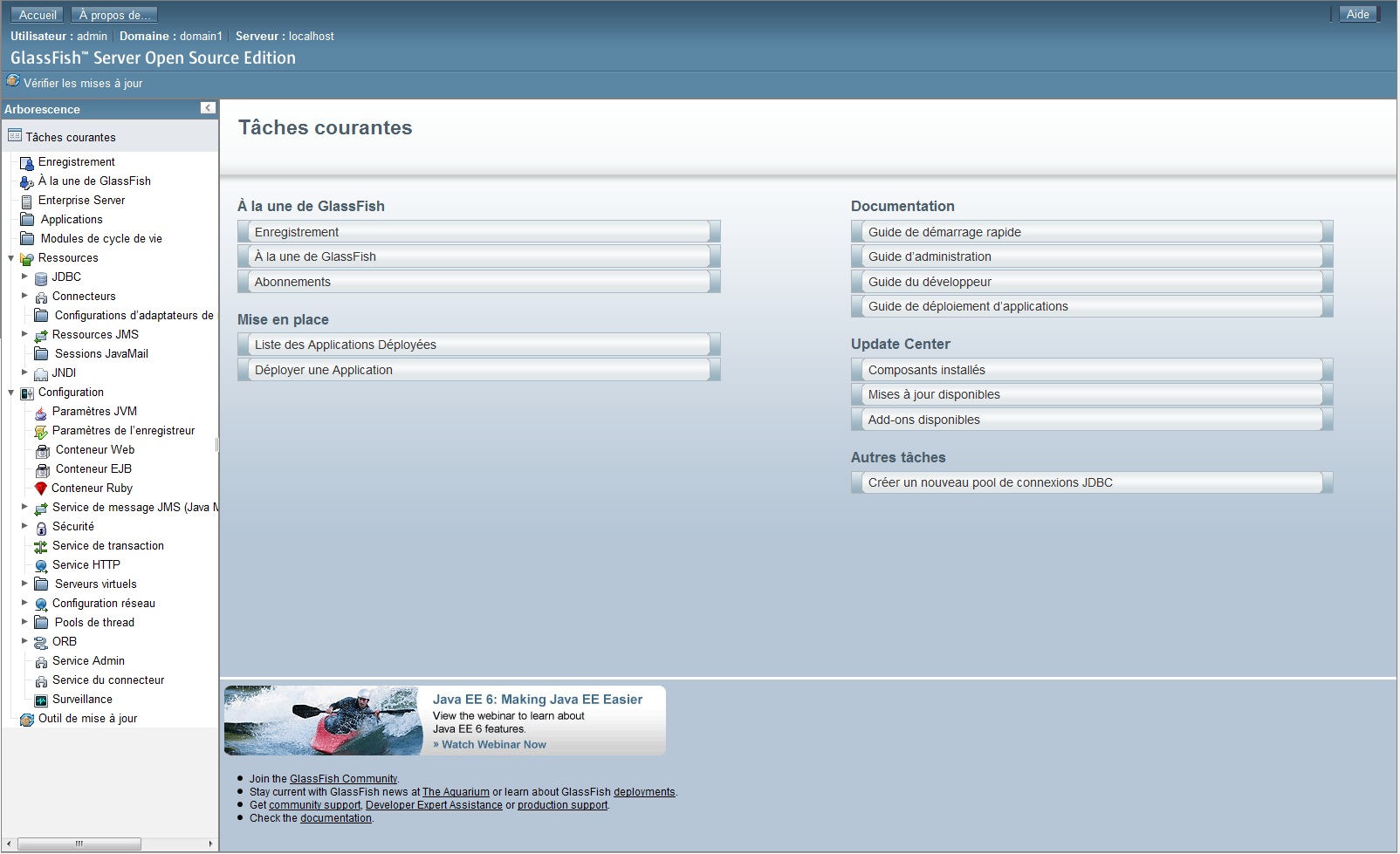
Vérifions maintenant que tout est ok... L'archive que nous avons installée comprend l'outil de développement netbeans mais aussi le serveur d'application de Sun, appellé Glassfish. Lancez netbeans, puis allez dans l'onglet "services". Nous allons lancer le serveur pour voir si tout est ok:



Puis, en demandant à voir la console d'administration, cela va ouvrir la page d'admin dans votre navigateur. En entrant le mot de passe *adminadmin*, une belle page devrait s'afficher. **J'espère que vous n'avez pas modifié le mot de passe proposé par défaut, car dans une classe de TP, les statistiques prouvent que la moitié des élèves ne se souviennent plus de leur mot de passe très rapidement!**

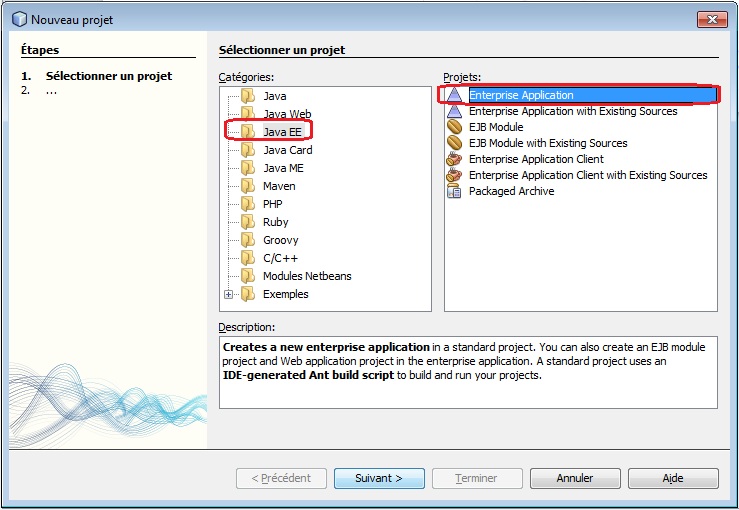


Le screenshot ci-dessous présente la console d'administration. Nous en reparlerons plus tard. Le fait qu'elle s'affiche dans votre navigateur prouve que l'installation est ok.

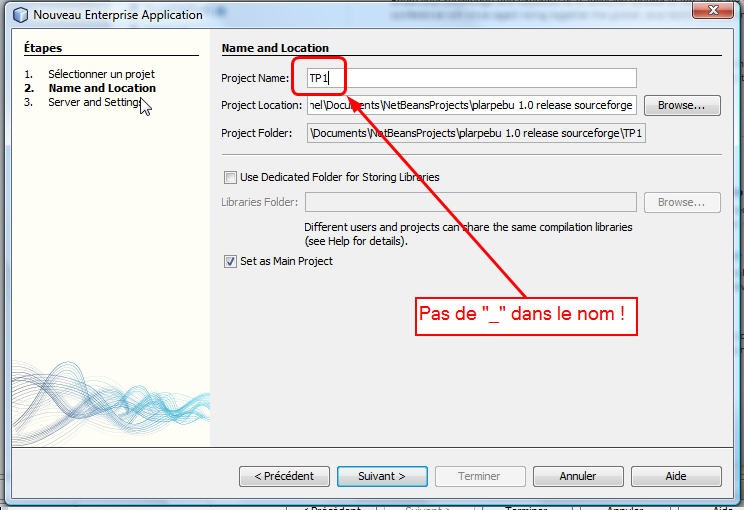


**2. Création d'un nouveau projet**

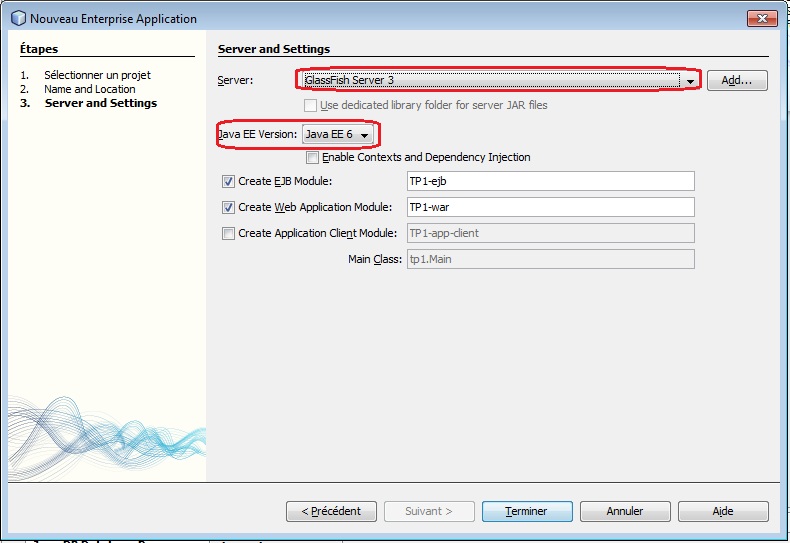
Nous allons créer maintenant un projet pour le TP d'aujourd'hui. Allez dans le menu file/new project, puis dans la fenêtre suivante, sélectionnez JavaEE  comme catégorie de projet et « enterprise application » comme projet:



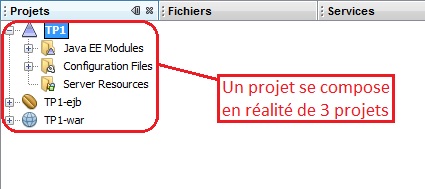
 Cliquez sur « next », puis donnez un nom à votre projet, cliquez sur next ou suivant...



Indiquez maitenant que nous allons utiliser le serveur d'application de Sun (glassfish v3 ou v3.1 et le mode Java EE6)



Une fois terminé, vous verrez dans l'onglet project « trois sous projets » :



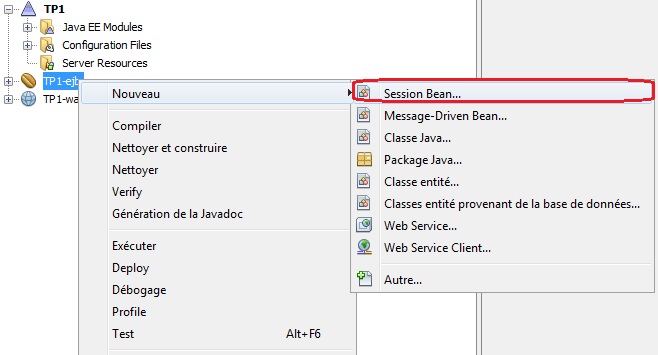
1. Le premier, "TP1", correspond au projet « global » qui contient en fait les archives des deux autres (il correspond au fichier .ear du cours),
2. le second, "TP1-ejb", correspond au .jar qui va contenir tous vos ejbs
3. le troisième, "TP1-war", correspond au .war qui va contenir vos servlets, jsps et pages html.

Le fichier .ear est en fait une archive qui contient les deux autres fichiers : le .jar et le .war, il permet de déployer l'ensemble de la web application d'un seul coup.

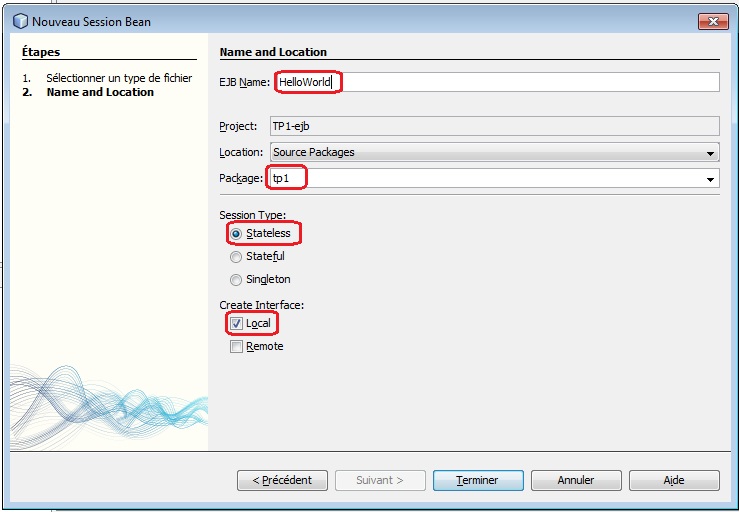
*Nous verrons par la suite qu'on peut aussi mettre des EJBs dans une simple application web (nouveauté Java EE 6).*

**3. Création d'un premier ejb de type session bean stateless : HelloWorld**

Allez sur le sous projet TP1-ejb (celui avec l'icone en forme de haricot), qui correspond à la partie "ejb" du projet) et faites bouton droit/new/session bean :

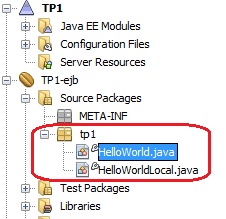


Puis dans la fenêtre suivante, indiquez le nom de votre Ejb, son type, et un nom de package (obligatoire) :



Une fois terminé, deux fichiers sont ajoutés dans le sous-projet : ***HelloWord.java*** et ***HelloWorldLocal.java***, qui correspondent à la classe du bean et à son interface « locale » (un bean avec un interface locale ne peut être utilisé que par un client tournant dans la même JVM : une jsp, une servlet ou un autre ejb).

NOTE : si jamais on ne coche pas d'interface, ce qui n'est possible qu'avec un serveur Java EE6, cela suppose que le bean dispose par défaut d'une interface locale. Il est toujours nécessaire de créer explicitement une interface remote dans le cas d'une application distribuée.



Ajoutons une méthode dans ce bean, la méthode ***getMessage()*** qui renvoie la chaîne de caractères «Hello World» :

[view plaincopy to clipboardprint?](http://miageprojet2.unice.fr/index.php?title=Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2F%2F_EJB_2009/TP1_EJB_:_mise_en_place_de_l'environnement%2C_premi%C3%A8res_applications)

1. **public** String getMessage() {
2. System.out.println("Hello World qui va s'afficher dans la console du serveur, pour trace");
3. **return** "Hello World";
4. }

public String getMessage() {

System.out.println("Hello World qui va s'afficher dans la console du serveur, pour trace");

return "Hello World";

}

Dans l'éditeur, une petite lampe jaune doit apparaître sur la gauche de l'en-tête de la méthode. Lorsqu'on clique dessus, une ou des actions sont proposées.  
Choisissez *"Expose method in local business interface HelloWorldLocal"* pour indiquer que cette méthode sera accessible au travers de l'interface locale du bean (ce n'est pas une méthode interne du bean).



Cette action ajoute automatiquement la déclaration de la méthode ***getMessage()*** dans l'interface HelloWorldLocal.java.  
(Si la petite lampe jaune n'apparaît pas, on peut obtenir le même résultat en déclarant manuellement la méthode ***getMessage()*** dans l'interface.)

[view plaincopy to clipboardprint?](http://miageprojet2.unice.fr/index.php?title=Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2F%2F_EJB_2009/TP1_EJB_:_mise_en_place_de_l'environnement%2C_premi%C3%A8res_applications)

1. @Local
2. **public** **interface** HelloWorldLocal {
3. java.lang.String getMessage();
4. }

@Local

public interface HelloWorldLocal {

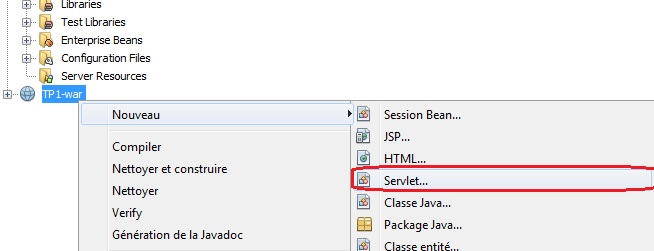
java.lang.String getMessage();

}

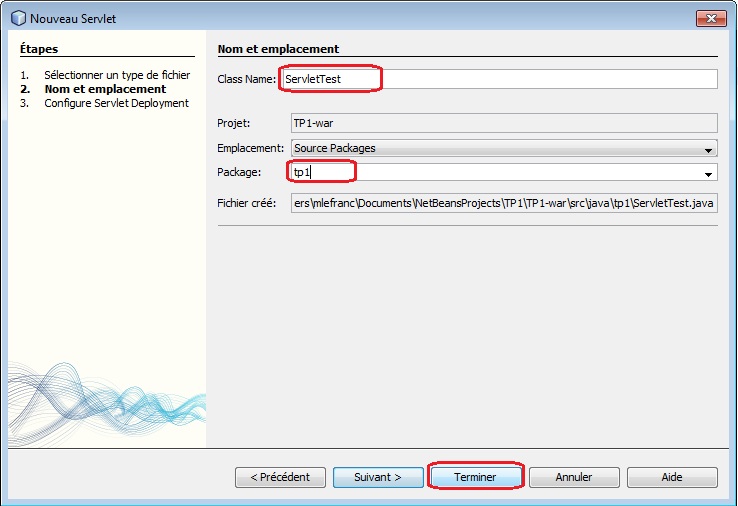
**Rappel** : les lignes commençant par **@** sont des **"attributs de code"** ou encore des **"annotations de code"**. Ce sont en fait *des méta-données*… Elles serviront au compilateur à **générer du code contextuel**. Ceci permet de n'avoir dans le source .java que du POJO (Plain Old Java Object, une classe java "normale" en fin de compte !)

**4. Ecriture d'une servlet client de test de ce bean**

Sur le projet finissant par -war, faites clic-droit/new/Servlet :



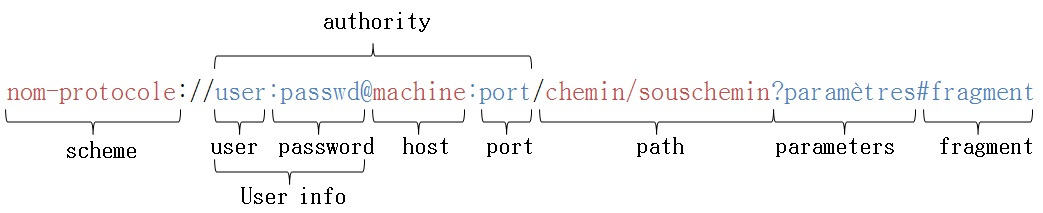
N'oubliez pas de donner un nom de package !



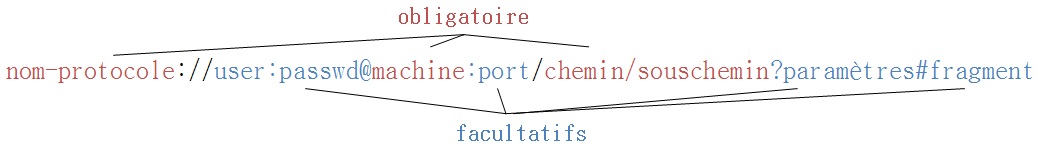
Vous verrez que votre servlet est apparue dans les sources du projet. Dans un projet Java EE 6, les caractéristiques des servlets ne se trouvent plus dans un descripteur web.xml, mais directement dans le code source, sous la forme d'annotations de code (lignes commençant par @). Ouvrez donc le code source de votre servlet et observez les annotations de code. Le second paramètre de l'annotation ***WebServlet*** indique le chemin relatif de l'URL auquel la servlet va répondre (c.f., Rappel URL ci-dessous). Vous pouvez modifier directement le code pour modifier l'URL, ou même utiliser des *expressions régulières*, par exemple : /Servlet\* etc.

Rappel : URLs :

différentes parties :



obligatoire/facultatif :



exemple :



[view plaincopy to clipboardprint?](http://miageprojet2.unice.fr/index.php?title=Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2F%2F_EJB_2009/TP1_EJB_:_mise_en_place_de_l'environnement%2C_premi%C3%A8res_applications)

1. @WebServlet(name="ServletTest", urlPatterns={"/ServletTest"})
2. **public** **class** ServletTest **extends** HttpServlet {

@WebServlet(name="ServletTest", urlPatterns={"/ServletTest"})

public class ServletTest extends HttpServlet {

Dans les sources de cette servlet, on va indiquer que l'on va « parler » au bean HelloWorld. Pour cela, cliquez avec le bouton droit dans le code de la servlet (n'importe où) et selectionnez « insert code » :

Un menu apparait, choisissez "call enterprise bean" :

Cela va insérer dans la Servlet deux lignes de code qui vont correspondre à la fois à la déclaration et à l'initialisation d'une variable référençant un objet du type de l'interface du Bean que l'on souhaite appeler. Le @EJB est l'annotation de code qui va procéder à "l'injection de code" lors de la compilation ou du déploiement. La variable ***helloWorld*** sera initialisée par le code injecté et ne vaudra pas "***null***" comme cela pourrait sembler :

[view plaincopy to clipboardprint?](http://miageprojet2.unice.fr/index.php?title=Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2F%2F_EJB_2009/TP1_EJB_:_mise_en_place_de_l'environnement%2C_premi%C3%A8res_applications)

1. **public** **class** ServletTest **extends** HttpServlet {
2. @EJB
3. **private** HelloWorldLocal helloWorld;

public class ServletTest extends HttpServlet {

@EJB

private HelloWorldLocal helloWorld;

Notons aussi que quelques imports ont étés rajoutés, ce sont ceux dont le code généré aura besoin. La variable ***helloWorld*** référençant un objet du type de l'interface du Bean ***HelloWorld*** va, à partir de maintenant pouvoir être utilisé directement dans la servlet, *sans que l'on ait besoin de l'initialiser*.

Ajoutons la ligne suivantes dans la méthode processRequest de la servlet, après avoir décommenté les lignes de processRequest() :

[view plaincopy to clipboardprint?](http://miageprojet2.unice.fr/index.php?title=Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2F%2F_EJB_2009/TP1_EJB_:_mise_en_place_de_l'environnement%2C_premi%C3%A8res_applications)

1. ...
2. **try** {
3. out.println("<html>");
4. out.println("<head>");
5. out.println("<title>Servlet ServletTest</title>");
6. out.println("</head>");
7. out.println("<body>");
8. out.println("<h1>Servlet ServletTest at " + request.getContextPath () + "</h1>");
9. out.println("L'ejb HelloWorld a renvoyé : <em>" + helloWorld.getMessage() + "</em>");
10. out.println("</body>");
11. out.println("</html>");
12. } **finally** {
13. ...

...

try {

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet ServletTest</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Servlet ServletTest at " + request.getContextPath () + "</h1>");

out.println("L'ejb HelloWorld a renvoyé : <em>" + helloWorld.getMessage() + "</em>");

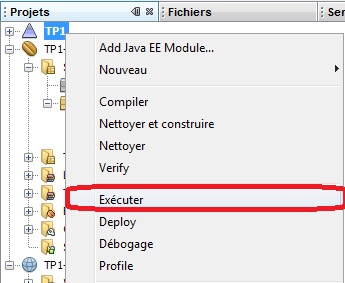
out.println("</body>");

out.println("</html>");

} finally {

...

Sauvegardons le tout, et exécutons le projet principal : **clic droit sur le projet-TRIANGLE, puis "exécuter"**



Cela va avoir pour effet de lancer le serveur d'application, (cette opération peut prendre un certain temps, des lignes vont s'afficher et au final, vous devriez avoir la ligne ci-dessous :

GÉNÉRATION TERMINÉE (durée totale  X seconde)

INFO : si on fait **plusieurs fois "exécuter"**, alors que le serveur tourne déjà, le serveur ne sera pas relancé  
INFO : si on se trompe et qu'on **"exécute" un sous-projet**, il faut "arréter" ce sous-projet avant d'exécuter le projet compet (projet-triangle)

Votre projet a été déployé. Normalement après un certain temps, le navigateur ouvre une page par défaut.  
L'URL par défaut est **le context de l'application web**,  (e.g., [**http://localhost:8080/TP1-war/**](http://localhost:8080/TP1-war/) ) . L'appel de cette URL appelle la page JSP **index.jsp**, "home page" du projet.

INFO : le fichier index.jsp se trouve dans le dossier "Web Pages" du sous-projet "TP1-war".  
INFO : dans JEE5 et antérieur, un fichier **web.xml** présent dans le war du projet permettait de configurer l'application web de différentes manières. Par exemple, il permettait de définir le nom de la "home page" en éditant le fichier web.xml  
POUR ALLER PLUS LOIN : que pouvait on configurer d'autre dans le web.xml ?

Rajoutons juste l'URL relatif de la servlet à la fin : (e.g., <http://localhost:8080/TP1-war/>**ServletTest** ). Cela invoque la servlet.

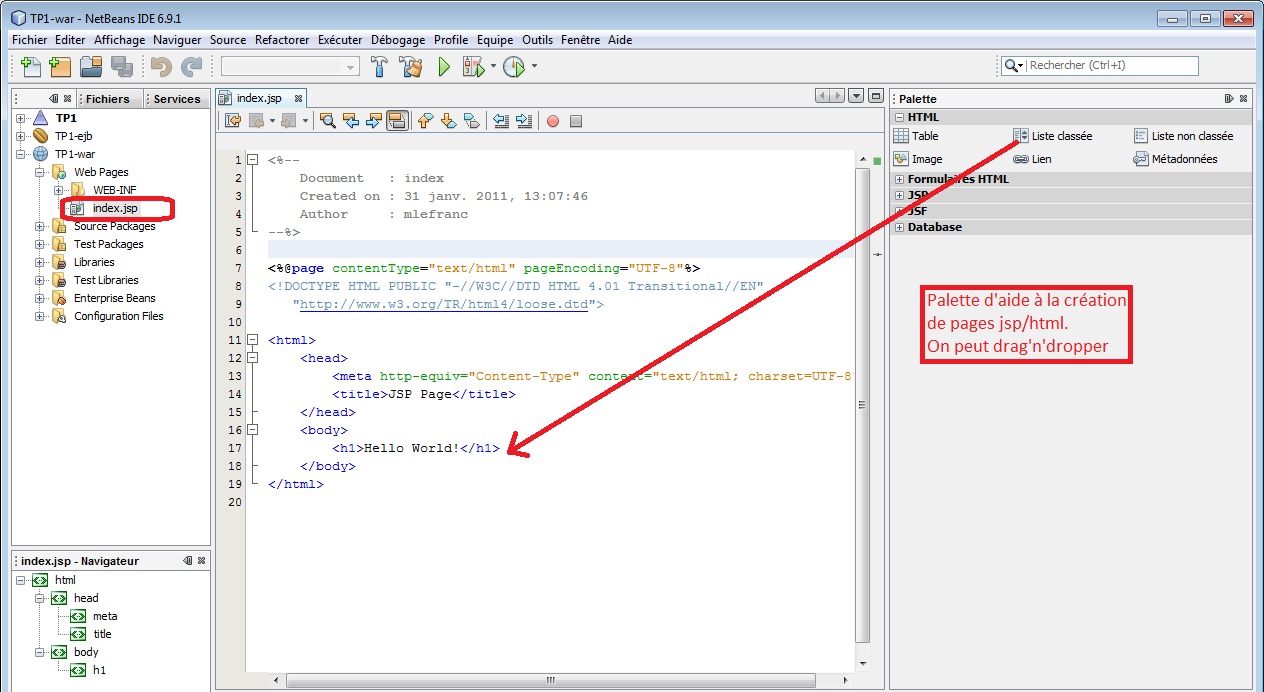


Voilà, vous avez exécuté votre première appli web utilisant des EJBs !!!!

**5. Modifions la page par défaut et insérons un petit menu**

La page par defaut est la page index.jsp du projet. Modifions là pour ajouter un menu avec une entrée qui permet de lancer la servlet facilement. Pour cela, double cliquez sur la page index.jsp dans le dossier Web Pages du sous-projet war.

Nous pouvons simplifier l'ajout de HTML en utilisant la palette disponible à partir de **Outils>palette (CTRL+MAJ+8)**



Modifiez le source comme ceci :

[view plaincopy to clipboardprint?](http://miageprojet2.unice.fr/index.php?title=Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2F%2F_EJB_2009/TP1_EJB_:_mise_en_place_de_l'environnement%2C_premi%C3%A8res_applications)

1. **<**%@page contentType="text/html" pageEncoding="UTF-8"%**>**
2. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3. "http://www.w3.org/TR/html4/loose.dtd"**>**
5. **<html>**
6. **<head>**
7. **<meta** http-equiv="Content-Type" content="text/html; charset=UTF-8"**>**
8. **<title>**JSP accueil TP1**</title>**
9. **</head>**
10. **<body>**
11. **<h1>**Menu du TP1**</h1>**
12. **<ul>**            **<li><a** href="ServletTest"**>**Appeler la Servlet de test**</a></li>**        **</ul>**
13. **</body>**
14. **</html>**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP accueil TP1</title>

</head>

<body>

<h1>Menu du TP1</h1>

<ul> <li><a href="ServletTest">Appeler la Servlet de test</a></li> </ul>

</body>

</html>

**6. Ecriture d’un second bean appelé par le bean HelloWorld**

* Procédez comme précédemment (paragraphe 3.) pour créer un second bean, semblable à ***HelloWorld***, appelez-le ***MonSecondBean***, ajoutez une méthode **getMessage()** qui renvoie « Je suis le second bean, appelé par le bean HelloWorld ».
* Modifiez le bean ***HelloWorld*** afin de lui attribuer un bean **MonSecondBean** .   
  Modifiez le bean ***HelloWorld*** afin la méthode **getMessage()** récupère la valeur de la méthode **getMessage()** de **monSecondBean**, et l'ajoute à la fin du message qu’elle renvoie.
* Une fois vos modifications terminées, sauvegardez, déployez, testez la servlet. (pour redéployer, il suffit de cliquer droit sur le projet-triangle, puis "exécuter".)

Au final, lorsqu’on exécute la servlet de test, cela doit afficher : **L'ejb HelloWorld a renvoyé : Hello World Je suis le second bean, appelé par le bean HelloWorld**