

## TP n°3

### Middleware à composants Java EE

#### Objectifs du TP :

- Introduction au middleware à composant
- Introduction à la plateforme Java EE
- Utilisation du serveur d'application GlassFish
- Utilisation de l'IDE NetBeans

#### Sujet : mon blog à moi

##### Présentation

Il s'agit de développer une première version d'un site Web de type blog. Dans cette première version, on se focalisera uniquement sur l'aspect fonctionnel (le site Web doit fonctionner selon les besoins et contraintes). L'aspect ergonomique et l'aspect sécuritaire ne seront donc pas considérés dans cette version.

L'application répartie sera constituée de trois couches : une couche présentation programmée principalement en JSF 2.x et Java ; une couche métier programmée en Java (API EJB) ; une couche donnée programmée en Java (API JPA + base de donnée Apache Derby intégrée à GlassFish).

**Remarque : dans ce TP, vous devrez suivre rigoureusement les étapes dans l'ordre indiqué. Tous les noms utilisés seront également rigoureusement les mêmes (options, fichiers, classes, etc.).**

##### Etapes

#### 1. Installation du serveur GlassFish

a) Rendez-vous sur le site <http://glassfish.java.net/>.

b) Avec **Netbeans 8.0** vous devez **impérativement utiliser Glassfish 4.0** (ou inférieur). Pour récupérer la bonne version, depuis la page principale du site, allez dans la partie *Download* puis la partie *Earlier Releases* puis la partie *Glassfish Archives* puis choisissez *glassfish-4.0.zip*.

Téléchargez l'archive *glassfish-4.0.zip* avec un clique droit puis item *Enregistrez le lien sous...* (cf. Remarque ci-après).

**Remarque : ne décompressez pas l'archive dans votre répertoire de connexion (répertoire distant), mais décompressez la dans le répertoire /tmp (répertoire local) de votre poste car le serveur Glassfish fait plus de 450 Mo. A la fin du TP, faites le ménage en supprimant le répertoire et le fichier zip.**

c) Ouvrez un terminal, allez dans le répertoire */tmp* puis décompressez avec la commande : *unzip glassfish-4.0.zip*. Cela crée un répertoire *glassfish4*.

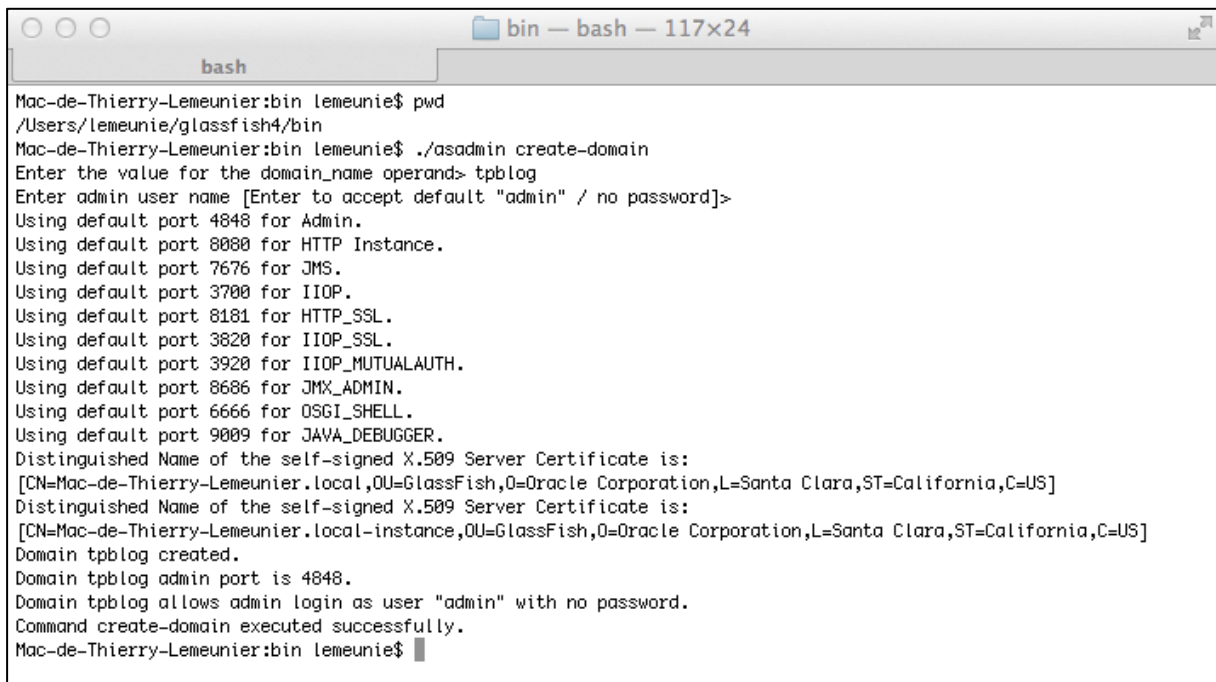
c) Lisez dans la documentation « Quick Start Guide » les paragraphes 1.1 à 1.4 (document disponible depuis la page principale du site dans la partie *Documentation*)

d) Testez que le serveur Glassfish fonctionne correctement en le démarrant et en l'arrêtant. Faire de même pour le serveur de base de données Java DB (Derby). Vérifiez également que l'on accède à la console d'administration à l'adresse <http://localhost:4848>.

**Remarque : si l'exécution du serveur Derby échoue, ce n'est pas grave à ce stade.**

## 2. Configuration du serveur GlassFish

GlassFish possède un domaine par défaut qui se nomme *domain1*. Nous allons créer un domaine propre à l'application que nous nommerons *tpblog*. Pour cela, tapez la commande *asadmin create-domain* et suivez les instructions comme dans l'exemple suivant :



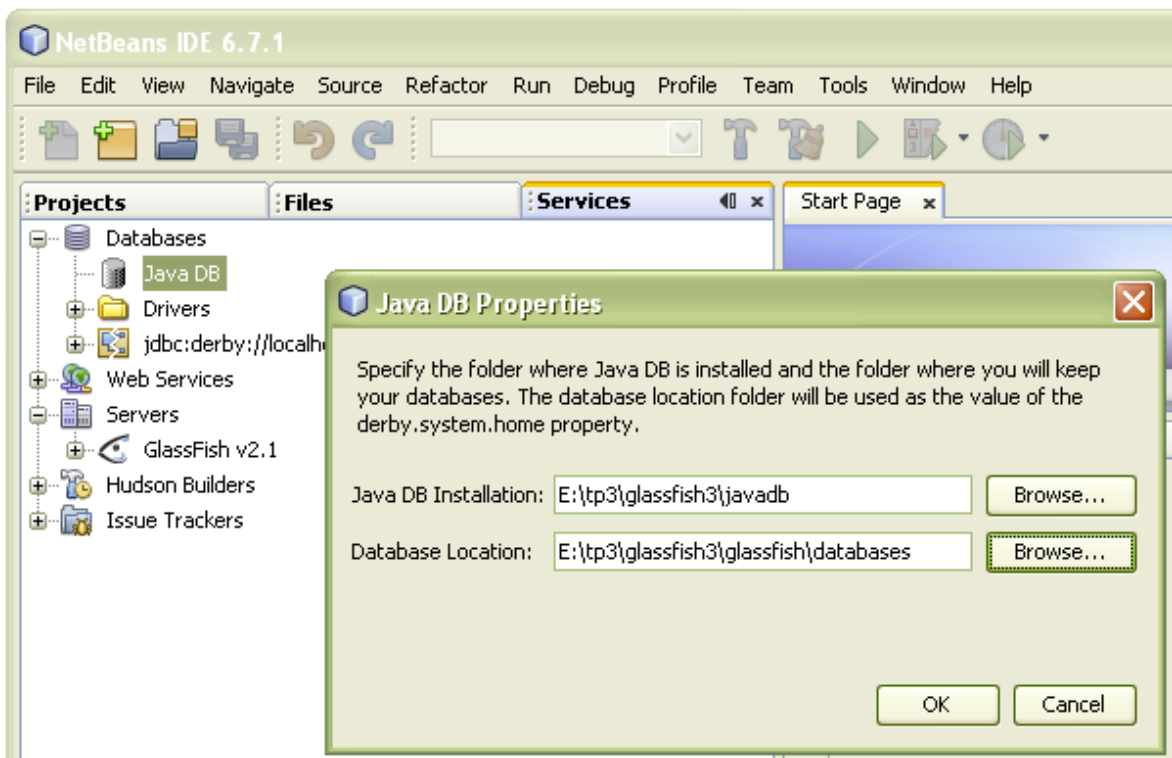
```
Mac-de-Thierry-Lemeunier:bin lemeunie$ pwd
/Users/lemeunie/glassfish4/bin
Mac-de-Thierry-Lemeunier:bin lemeunie$ ./asadmin create-domain
Enter the value for the domain_name operand> tpblog
Enter admin user name [Enter to accept default "admin" / no password]>
Using default port 4848 for Admin.
Using default port 8080 for HTTP Instance.
Using default port 7676 for JMS.
Using default port 3700 for IIOP.
Using default port 8181 for HTTP_SSL.
Using default port 3820 for IIOP_SSL.
Using default port 3920 for IIOP_MUTUALAUTH.
Using default port 8686 for JMX_ADMIN.
Using default port 6666 for OSGI_SHELL.
Using default port 9009 for JAVA_DEBUGGER.
Distinguished Name of the self-signed X.509 Server Certificate is:
[CN=Mac-de-Thierry-Lemeunier.local,OU=GlassFish,O=Oracle Corporation,L=Santa Clara,ST=California,C=US]
Distinguished Name of the self-signed X.509 Server Certificate is:
[CN=Mac-de-Thierry-Lemeunier.local-instance,OU=GlassFish,O=Oracle Corporation,L=Santa Clara,ST=California,C=US]
Domain tpblog created.
Domain tpblog admin port is 4848.
Domain tpblog allows admin login as user "admin" with no password.
Command create-domain executed successfully.
Mac-de-Thierry-Lemeunier:bin lemeunie$
```

Cela a pour effet de créer un domaine intitulé *tpblog* qui écoute sur le port 8080. Le port d'administration est le 4848. Testez que ce nouveau domaine fonctionne en démarrant le serveur pour le domaine *tpblog* et en vérifiant l'accès à la console d'administration.

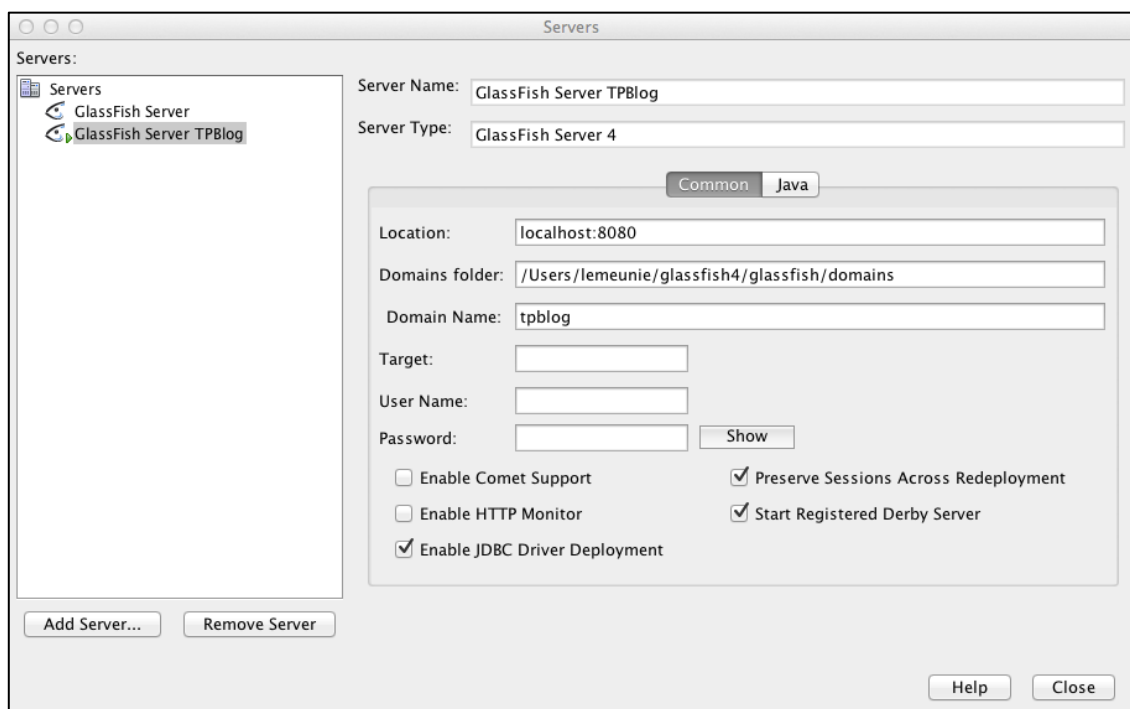
Avant de continuer, arrêtez la base de données et arrêtez le serveur d'application s'ils sont en cours d'exécution.

## 3. Configuration de l'EDI NetBeans

a) Lancez NetBeans. Dans l'onglet *Services* modifiez les propriétés de Java DB pour qu'ils utilisent la base Derby intégrée au serveur Glassfish (sélectionnez *Java DB*, cliquez à droite, item *Properties*). Par exemple :



b) Dans l'onglet *Services*, ajoutez un serveur d'application (item *Servers* de l'arborescence des services, cliquez à droite puis item *Add Server...*). Une fois ajouté, vous devrez obtenir les propriétés suivantes :



c) L'application nécessite deux ressources liées à la couche donnée : un pool de connexion JDBC à la base de donnée Derby et un nom JNDI pour publier cette source de données.

Assurez-vous d'avoir lancé le serveur d'application Glassfish sur le domaine *tpblog* (dans l'onglet *Services*, sélectionnez le serveur Glassfish que vous venez de configurer, puis

cliquez droit et item *Start*) et d'avoir également lancé le serveur de base de données Derby (normalement lancement automatique avec le serveur Glassfish).

**Remarque : si l'exécution du serveur Derby échoue, il faut désactiver le gestionnaire de sécurité comme proposé par Netbeans puis relancer la base Berby (sélectionnez *Java DB*, cliquez à droite puis item *Start Server*).**

Pour créer le pool de connexion, rendez-vous dans la console d'administration dans la partie *Ressources/JDBC/JDBC Connection Pools* (dans l'onglet *Services*, sélectionnez le serveur Glassfish que vous venez de configurer, puis cliquez droit et item *View Domain Admin Console*).

Créez un nouveau pool avec les paramètres suivants :

Pool Name	tpblogPool
Ressource Type	javax.sql.DataSource
Database Driver Vendor	Derby
Datasource Classname	org.apache.derby.jdbc.ClientDataSource
User	dbuser
Password	dbpwd
Database Name	tpblogDB
Connection Attributes	;create=true

Assurez-vous que l'option *Ping* est bien « *enabled* ». Cela crée le répertoire de la base de données dans le répertoire du serveur dédié au stockage des bases gérées par Derby.

Pour créer une source de données, rendez-vous dans la console d'administration dans la partie *Ressources/JDBC/JDBC Ressources*. Créez une nouvelle source de donnée avec les paramètres suivants :

JNDI Name :	jdbc/tpblogDS
Pool Name :	tpblogPool

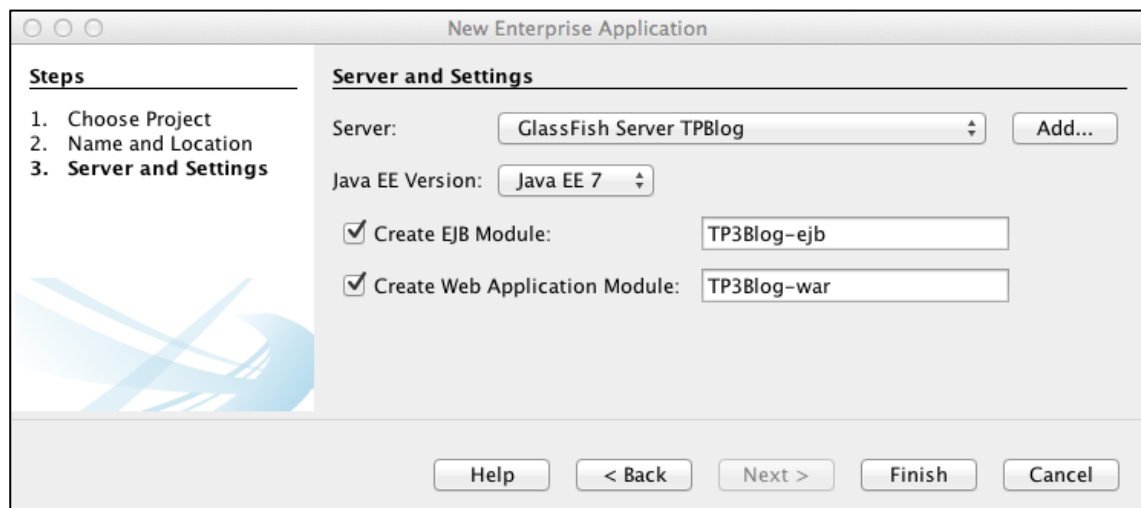
**Note 1 : Le schéma DBUSER et les tables seront créées automatiquement lors du premier déploiement. Si ce n'est pas le cas, vous pouvez utiliser le fichier *create\_tpblogDB.sql* fourni sur UMTICE.**

**Note 2 : Les tables de la base sont vidées à chaque re-déploiement de l'application (mais le schéma DBUSER n'est pas supprimé).**

*A ce stade du développement, vous devez pouvoir lancer directement le serveur d'application et la base de données à partir de NetBeans (sélectionnez le serveur, cliquez à droite, puis item *start*). **Il ne sera plus nécessaire de passer par la ligne de commande pour lancer le serveur d'application Glassfish et le serveur de base de données Derby.***

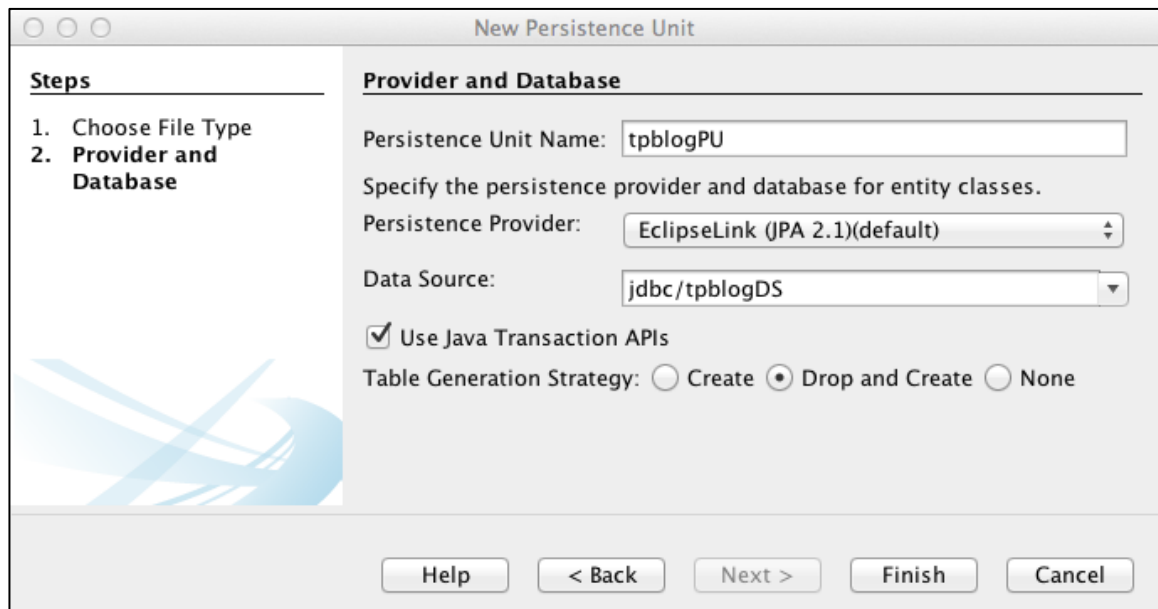
#### 4. Création des entités et des EJB sans état

- Lancez le serveur d'application dans NetBeans (et le serveur de base de données si cela n'est pas fait automatiquement) s'il n'est pas déjà lancé.
- Dans l'onglet *Projects*, créez une application d'entreprise Java EE nommé *TP3Blog* (cliquez à droite, item *New Project...* puis dans la catégorie *Java EE* choisissez *Entreprise Application*). Cette application sera constituée d'un module EJB et d'un module Web :



Si l'option *Enable Contexts and Dependency Injection* apparaît, cochez la case.

- c) Dans le module EJB, créez une unité de persistance (sélectionnez le module EJB de l'application, cliquez à droite, item *New/Persistence Unit...*) :

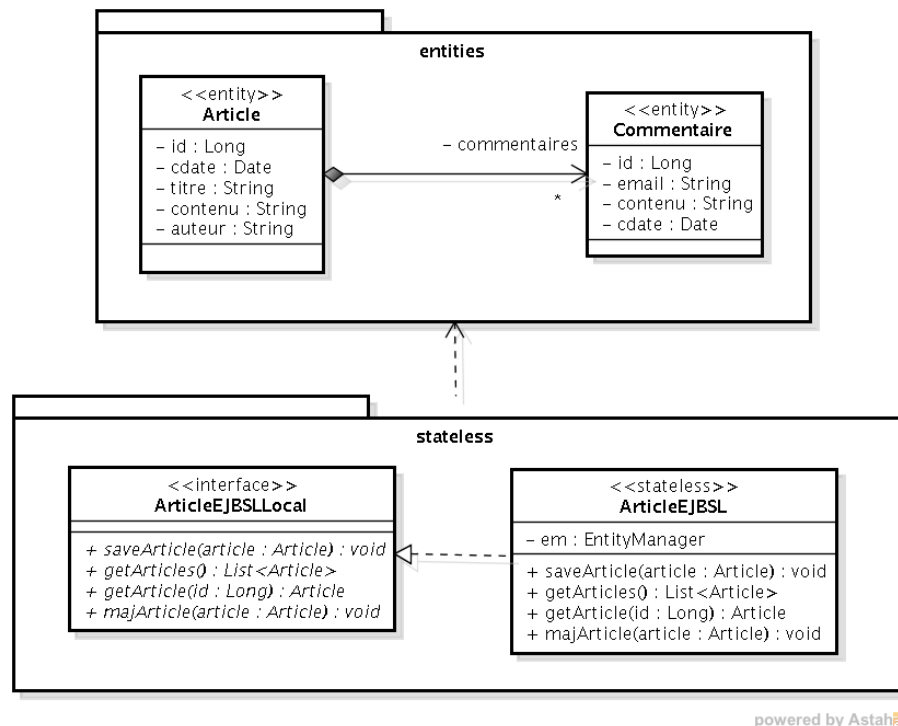
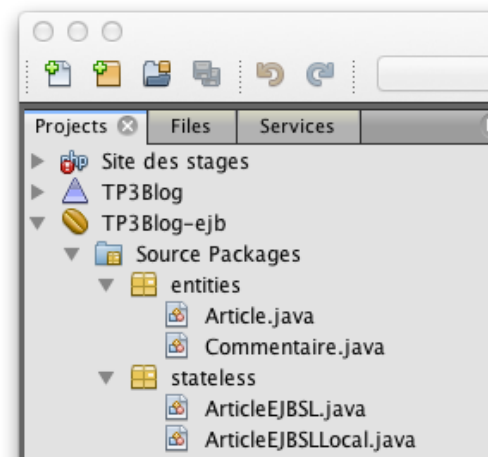


**Note 3 : Pour la suite du TP, vous utiliserez les fichiers sources disponibles dans le zip accessible sur UMTICE. Les fichiers suivants sont à compléter : Article.java, ArticleEJB.java, editarticle.xhtml**

- d) Dans le module EJB créez les entités de l'application : *Article* et *Commentaire* ainsi que l'EJB sans état *ArticleEJB* et son interface *ArticleEJBLocal*.

Aidez-vous des sources disponibles.

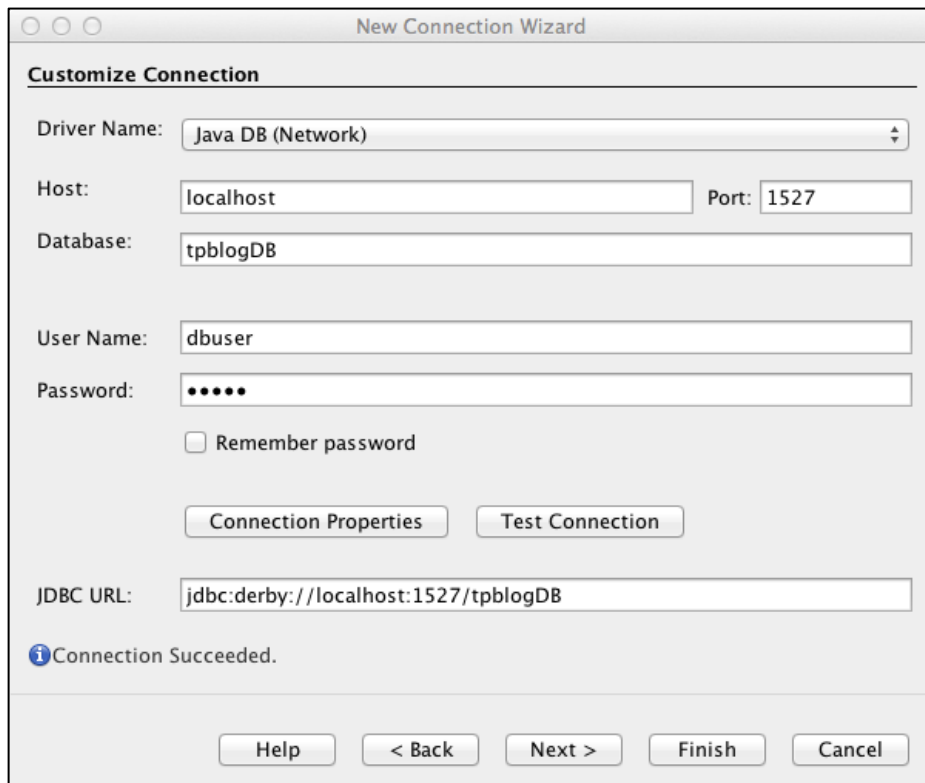
Veuillez suivre la structure des packages suivante :



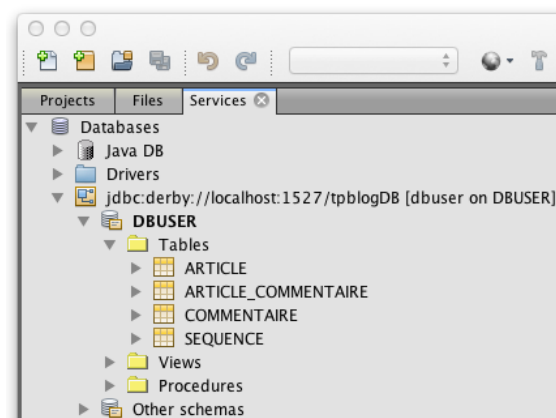
powered by Astah

*A ce stade du développement, l'application peut être déployée sur le serveur GlassFish (sélectionnez l'application TP3Blog, cliquez à droite, item Deploy).*

*Après déploiement, la base tpblogDB est accessible et visualisable directement dans NetBeans. Pour cela, créez une nouvelle connexion (onglet Services, sélectionnez DataBases, cliquez à droite puis item New Connection...):*

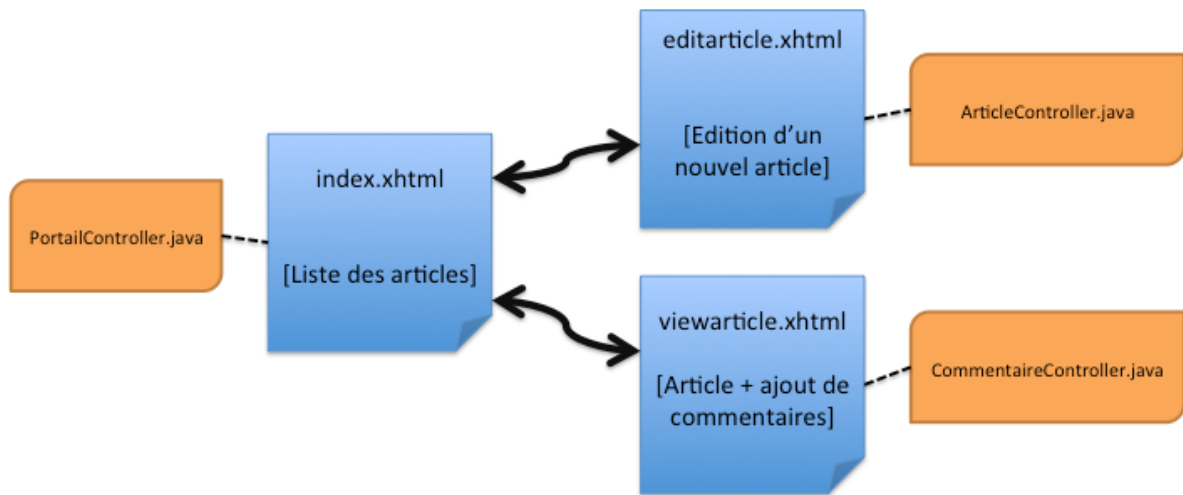


Une fois la connexion établie, vous devez pouvoir visualiser les tables de la base (cf. Note 1 si ce n'est pas le cas) :

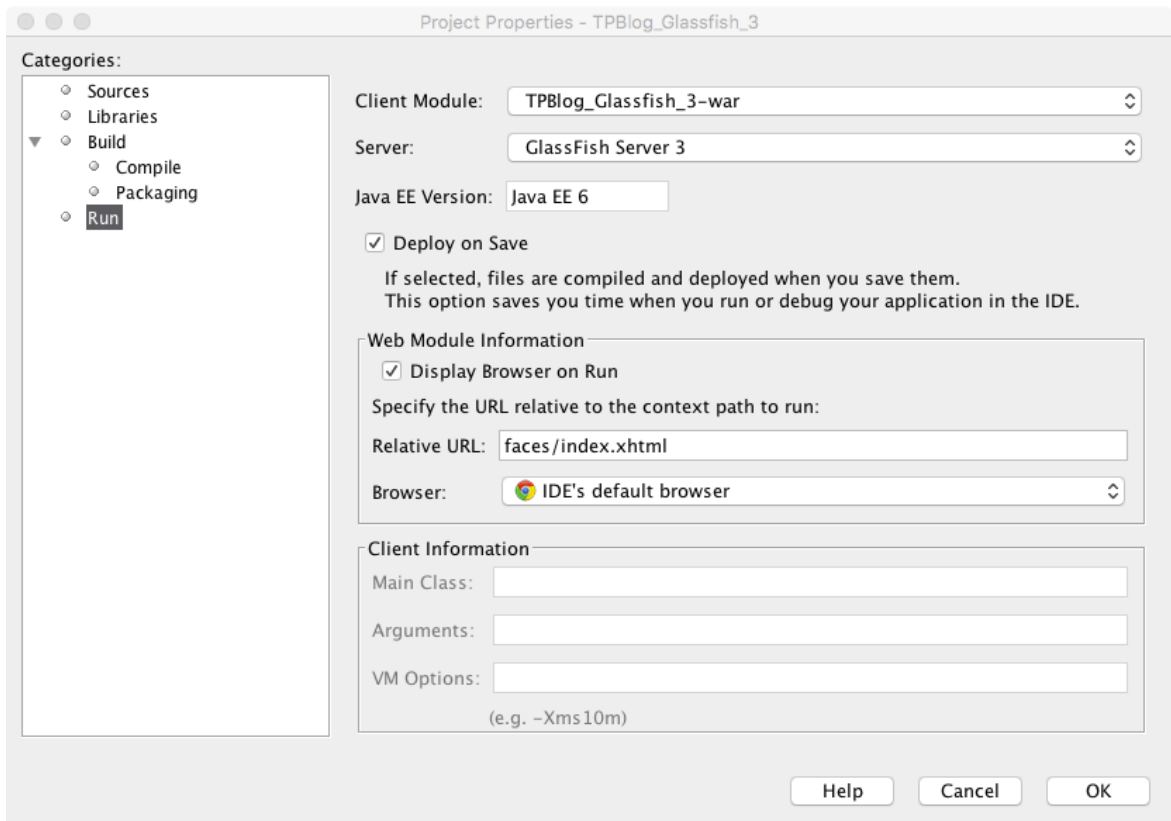


##### 5. Création des pages JSF et des *managed beans* associés

Dans le module Web de l'application (*TP3Blog-war*), à partir des fichiers sources disponibles, créez les pages *index.xhtml*, *viewarticle.xhtml* et *editarticle.xhtml* ainsi que leurs *managed bean* associés tels que le montre le schéma suivant :

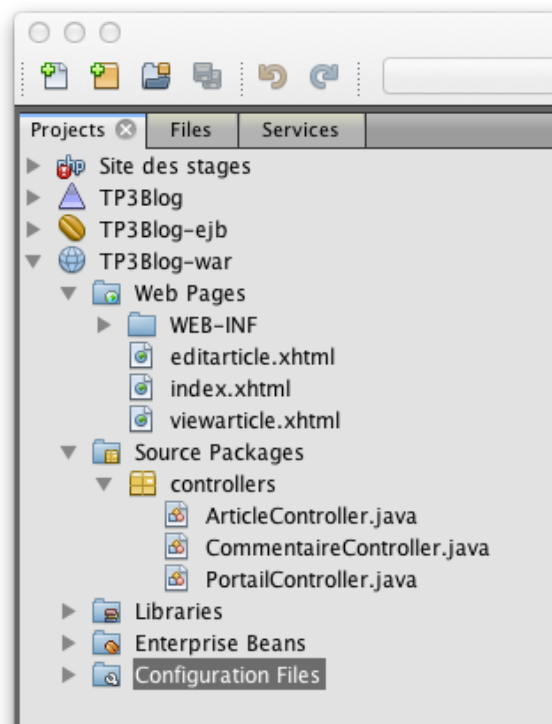


Pensez à indiquer si nécessaire la page d'accueil du site (sélectionnez le projet *TP3Blog*, cliquez à droite, item *Properties*, catégorie *Run*) :



Veuillez suivre la structure des packages suivante :





**Note 4 :** Pour tester l'application, pensez d'abord à la déployer (sélectionnez l'application **TP3Blog**, cliquez à droite, item **Deploy**) puis lancez l'exécution (sélectionnez l'application **TP3Blog**, cliquez à droite, item **Run**). A chaque déploiement, la base de données est vidée.

Tous les champs de *editarticle.xhtml* et de *viewarticle.xhtml* sont obligatoires. Un message d'erreur est affiché si le champ est vide au moment de la validation.

La page d'entrée de l'application (*index.xhtml*) lorsqu'il n'y a aucun article :



La page d'édition d'un nouvel article (*editarticle.xhtml*) accessible à partir de la page d'entrée (bouton « Rédiger un article ») :



**Edition d'un article**

Titre:

Auteur:

Date:

Contenu:

• Le champs contenu doit être rempli.

La page d'entrée lorsqu'il y a un ou plusieurs articles :



**Bienvenue sur mon blog**

Titre	Auteur	Date	Commentaires
<a href="#">Mon premier article</a>	toto	07-02-2014 13:24	0

La page *viewarticle.xhtml* d'un article (accessible en cliquant sur le lien de l'article affiché dans la page d'entrée) lorsqu'il n'y pas encore de commentaire :



**Article**

Mon premier article  
rédigé par toto  
le 07/02/14 14:24  
Super j'adore. Mon premier article de mon blog !

**Commentaires**

Il n'y a aucun commentaire.

**Ajouter un commentaire**

Date :

Email :

Contenu :

Après édition des champs du commentaire et l'enregistrement, on reste sur la même page. Pour ce traitement, on utilisera Ajax.

Après l'enregistrement d'un ou plusieurs commentaires la page est :

**Article**

Mon premier article  
rédigé par toto  
le 07/02/14 14:24  
Super j'adore. Mon premier article de mon blog !

**Commentaires**

Date	Email	Contenu
07-02-2014 13:20:13	toto@gmail.com	Super. C'est impressionnant !
07-02-2014 13:26:26	toto@gmail.com	Super. C'est impressionnant !

**Ajouter un commentaire**

Date : 07-02-2014 13:26:41  
Email :   
Contenu :

La page d'entrée indique le nombre de commentaires de chaque article :

**Bienvenue sur mon blog**

Titre	Auteur	Date	Commentaires
<a href="#">Mon premier article</a>	toto	07-02-2014 13:24	2

---

*A ce stade du développement, l'application web est terminée.*

---

## Travail à rendre

Le travail donne lieu à un compte-rendu à déposer sur UMTICE **à la fin de la séance**. Il sera rendu uniquement par fichier compressé (format ZIP exclusivement) contenant le projet NetBeans complet (répertoire du projet compressé).