

<p>Etude des serveurs d'applications EJB Cas particulier du serveur Bea/Weblogic</p>
--

Cette série de TP doit vous permettre de prendre en main l'architecture des serveurs d'applications proposés par la société BEA pour Weblogic.

Le TP est constitué de 4 séances de 4 heures de TP et d'une séance de TD. L'objectif de ces séances est de voir le fonctionnement d'un certain nombre d'aspects clés du développement sur les serveurs d'applications.

La difficulté des TP est croissante, et il est nécessaire de bien réaliser et comprendre chacune des étapes avant de faire la suite des TP.

Les tâches à réaliser dans ces TP sont les suivantes :

- Prise en main du serveur d'applications --> séance n°1
- Prise en main du service de nommage --> séance n°1
- Déploiement d'un EJB (Off the shelves) --> séance n°2
- Développement d'un client d'un EJB --> séance n°2
- Déploiement d'un EJB (from the sources) --> séance n°2
- Conception d'un site d'enchères électroniques --> séance de TD
- Développement et mise en oeuvre d'un site d'enchères électroniques --> séances n°3 et n°4

TP n°1

Mise en oeuvre d'un serveur d'application

I) Mise en place de l'environnement

1) Java

Plate-forme Java : jdk 1.3 , installée sous /opt/jdk1.3

Développer et lancer à l'aide d'un script le programme le plus simple en java. (Le classique " Hello, World ! ")

2) EJB

Plateforme EJB : weblogic 6 conforme aux spécifications EJB 2.0, installée sous /opt/weblo6

- ==> Lister les répertoires utilisés par le serveur
- ==> Définir approximativement le rôle de ces répertoires

3) Environnement utilisateur

Décompacter le fichier /home/Partage/enseignants/Info/TP-Frenot/sid/weblo6_user.tgz dans votre répertoire

- ==> Que contient cette archive ?
- ==> Modifier le script de lancement afin qu'il puisse exécuter votre serveur
- ==> Lancer votre serveur (startWebLogicTelecom.sh lire les remarques avant)
- ==> Vérifier que le serveur d'applications est bien lancé. Comment ?
- ==> Lancer la console d'administration (application console)

Trouver dans le fichier de configuration de votre serveur le mot de passe de l'utilisateur 'system'.

- ==> Tester le serveur Web avec https://tc-info-xxx :7002/. Que se passe t'il ?
- ==> Examiner les différentes rubriques de la console concernant votre serveur

4) Remarques

- Java pose un problème sur la version de RedHat que vous utilisez. Il faut ajouter la ligne "export LD_ASSUME_KERNEL=2.2.5" au script de lancement du serveur : startWebLogicTelecom.sh
- Le login pour lancer/se connecter sur le serveur : system/weblogic6

II) Exploitation du service de nommage

But : Réaliser des classes de manipulation du service JNDI qui est fourni en standard par le serveur.

Ecrire deux programmes simples s'échangeant des objets. Chacun de ces programmes est écrit sous forme d'une classe.

Le premier programme réalise le "binding" de trois objets dans le service de nommage du serveur. Ces objets sont des instances de la classe CarnetAdresse (à créer). Le client doit utiliser la méthode rebind() pour réaliser cette action.

Le second programme doit récupérer les entrées avec la fonction lookup() sur les objets déposés (il n'utilise **pas** les fonctions list() ou listBindings()). Il doit également vérifier que l'état des objets ainsi récupérés est bon en invoquant par exemple les accesseurs de la classe AddressBook.

Voici une description de la classe CarnetAdresse. Mettre cette classe dans votre package.

```
public class AddressBook implements serializable{  
    public AddressBook (String name, String address, String phone, int age);  
    String getName();  
    String getAddress();  
    String getPhone();  
    String getAge();  
}
```

Comment visualiser le contenu du service de nommage. Que doit contenir ce service une fois la manipulation réalisée ?

Remarque : Pour définir le contexte de nommage initial, il faut définir les variables :
Context.INITIAL_CONTEXT_FACTORY: weblogic.jndi.WLInitialContextFactory
Context.PROVIDER_URL : t3://localhost:7001

III) Question optionnelle

Afin d'améliorer l'utilisation du service de nommage, il est possible d'utiliser les fonctions de liste offert par Weblogic. Il est alors possible de parcourir toute l'arborescence du service de nommage et d'afficher tous les noms.

Remarque importante : L'accès au contexte initial est une méthode que vous allez réutiliser dans la suite des tp. Elle permet d'obtenir une connexion sur le service de nommage (comme une connexion sur une base de données).

Tp n°2

L'intégration de composants

L'objectif de ce TP est d'utiliser un serveur d'application pour déployer " rapidement " une application. Le déploiement d'une application consiste à déployer le bean implantant le service et de développer un client utilisant ce service.

Dans ce TP nous utiliserons un Enterprise Java Bean de gestion d'e-panier (ShoppingCart). L'EJB ShoppingCart est un composant qu'un site Web peut utiliser pour le suivi des éléments achetés par un client. C'est un bean de type statefull session car il faut pouvoir suivre l'évolution du panier d'achat du client. L'EJB ShoppingCart utilise également la classe ShoppingCartItem pour créer les objets du panier.

La première étape consiste à déployer le bean sur le serveur d'application afin que le client puisse l'exploiter.

Le bean est défini dans le jar /home/Partage/enseignants/TP-Frenot/sid/tp_sujet_weblo.tgz

Déploiement d'un bean

Le bean est packagé dans un jar ShoppingCartEJB.jar situé dans le répertoire exercices. Son descriptif est le suivant :

Type	Statefull
Interface Remote	com.bea.EJBC.examples.ShoppingCart
Interface Home	com.bea.EJBC.examples.ShoppingCartHome
class d'implantation de l'EJB	com.bea.EJBC.examples.ShoppingCartBean

- Décompacter l'archive, et vérifiez que les fichiers s'y trouvent bien. Quels sont les autres fichiers contenus dans l'archive, à quoi servent t'ils ?

Supprimer les fichiers une fois l'opération terminée

- Pour déployer un jar il existe plusieurs manières :
 - 1) Déposer l'application dans un repertoire spécifique du serveur. Celui ci met alors à disposition toutes les applications de ce repertoire
 - 2) Utiliser la console web d'administration (section EJB)

Vous utiliserez le mécanisme de déploiement de la console. Vous vérifierez que le bean est bien déployé en vérifiant que le stub de manipulation de l'usine se trouve bien dans le service de nommage.

Ecriture d'un client

Cette étape consiste à écrire un client utilisant le bean " fraîchement " déployé. Pour cela le client doit :

- 1) Créer trois instances de la classe ShoppingCartItem.

- 2) Créer une instance de l'EJB ShoppingCart et placer les éléments créés dans l'EJB à l'aide de la méthode addItem() de l'interface remote de l'EJB
- 3) Après avoir placé les Eléments dans le panier vérifier leur nombre en utilisant la méthode getItemCount()
- 4) Extraire le second élément du bean en utilisant la méthode getItem(). Afficher son descriptif en utilisant ses accesseurs.
- 5) Retirer un des éléments en utilisant la méthode removeItem()
- 6) Interroger le bean afin de lui demander le nombre d'articles restant.

Méthodes de l'interface remote de ShoppingCartBean

```
void addItem(ShoppingCartItem i) ;  
ShoppingCartItem getItem(int i) ;  
int getItemCount() ;  
void removeItem(ShoppingCartItem i) ;
```

L'interface home du bean possède les méthodes de création suivantes :

```
ShoppingCart create() ;
```

Un élément de la ShoppingCart possède les méthodes suivantes :

```
ShoppingCartItem(double prix, String desc, String numSerie) ;  
String getDesc() ;  
double getPrice() ;  
String getSerialNum() ;
```

3) Packaging d'un Bean

Pour déployer un bean il faut packager celui-ci afin que le container d'EJB soit capable de l'instancier facilement en mémoire.

Le principe est :

- 1) Définition des fichiers de description du bean
- 2) Création d'un jar contenant tout le bean
- 3) Déploiement du bean

Le bean à packager possède les caractéristiques suivantes :

Type : Stateful Session

Interface Remote : lego.LegoBlockSession

Interface Home : lego.LegoBlockSessionHome

Classe de l'EJB : lego.LegoBlockSessionBean

1) Compiler les fichiers

2) Définir les fichiers de description

Dans cette première étape il faut créer les deux fichiers du répertoire META-INF.

Quels sont ces fichiers ?

Que contiennent ces fichiers ?

3) Créer un fichier jar contenant tous les fichiers créés

3) Compiler mes fichiers avec l'outil ejbc

```
« 'java -classpath /gore/java/weblo6/wlserver6.0/lib/weblogic.jar weblogic.ejbc  
./legoBlock.jar ./legoBlockDep.jar »
```

* Remarque : si la compilation pose un problème, il faut essayer de compiler avec l'option -source de javac.

4) Le bean est enfin prêt à être déployé et utilisé sur le serveur d'applications.

Vérifier que le déploiement s'est bien passé ?

TP n°3 & 4

Conception et implantation d'un système de gestion de ventes en ligne

L'objectif du TP est de concevoir, implanter et déployer un service de gestion de vente en ligne. Les éléments entrant en jeu dans un mécanisme de vente sont :

- Le gestionnaire de vente
- L'élément en vente
- Le vendeur
- Les acheteurs

L'objectif est de réaliser le service sous la forme d'un EJB.

I) Conception du système

Définir le ou les EJB qui interviennent dans le système. (Il y a plusieurs architectures possibles).

Définir le mode de fonctionnement du système. (interaction avec d'autres services : nommage, base de données).

II) Développement, Déploiement et implantation du service

Concevoir et implanter les classes réalisant le service. Il est possible de réaliser plusieurs versions du système : client java, client jsp/html.

III) Travail à rendre

Schéma des classes et interaction entre les composants. Service pré-packagé sous forme de bean, testable sur n'importe quel serveur d'applications