

## TP 6

# La persistance des données - API JPA

## JPA (JTA) / EJB / WEB

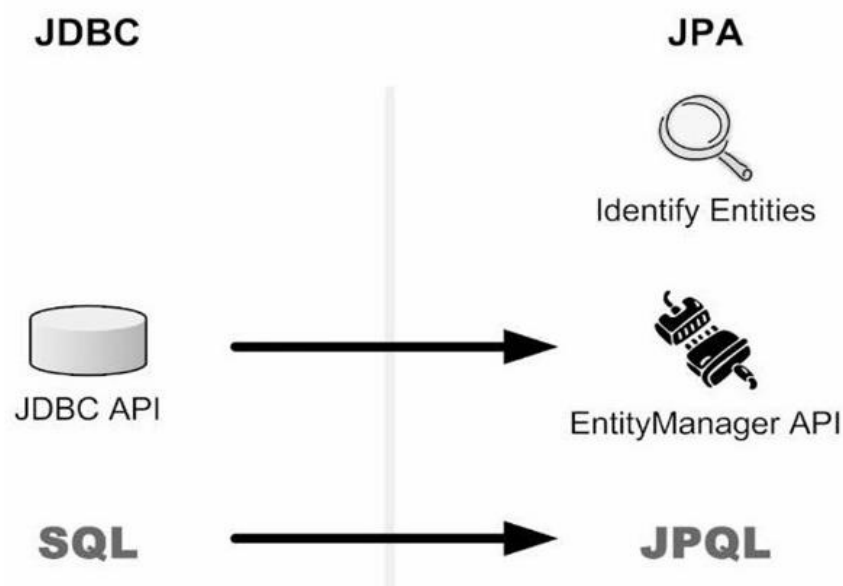
### Les objectifs de ce TP sont :

- Création d'une Source de Données JTA
- Accès aux données avec l'API de persistance JPA / EJB
- Utilisation des requêtes JPQL
- Exploitation par une application cliente Web (Dynamic web project)

### I. Migration vers EJB3 / JPA2

Pour migrer de JDBC DAOs vers l'utilisation de EJB 3 / JPA 2, **trois phases sont nécessaires**:

- ▶ Identifier les entités.
- ▶ Remplacer l'implémentation des classes DAO par EntityManager API pour générer les opérations sur les entités au lieu du JDBC.
- ▶ Migrer des requêtes SQL vers JPQL et the EntityManager API.



## II. Création de la base de données gestioncb

Table propriétaire

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<input type="checkbox"/> 1	<u>idProp</u>	int(11)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 2	nom	varchar(30)	latin1_swedish_ci		Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 3	cin	varchar(30)	latin1_swedish_ci		Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 4	tel	varchar(30)	latin1_swedish_ci		Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire

Table compte avec une clé primaire idCompte et une clé étrangère idProp.

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<input type="checkbox"/> 1	<u>idCompte</u>	int(11)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input checked="" type="checkbox"/> 2	<u>idProp</u>	int(11)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 3	dateCreation	date			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 4	solde	decimal(10,3)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire

Tout cocher / Tout décocher Pour la sélection : Afficher Modifier Supprimer Primaire Unique **Index**

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<input type="checkbox"/> 1	<u>idCompte</u>	int(11)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 2	<u>idProp</u>	int(11)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 3	dateCreation	date			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire
<input type="checkbox"/> 4	solde	decimal(10,3)			Non	Aucune		Modifier Supprimer Affiche les valeurs distinctes Primaire

Tout cocher / Tout décocher Pour la sélection : Afficher Modifier Supprimer Primaire Unique Index

Version imprimable **vue relationnelle** Suggérer des optimisations quant à la structure de la table Suivre la table

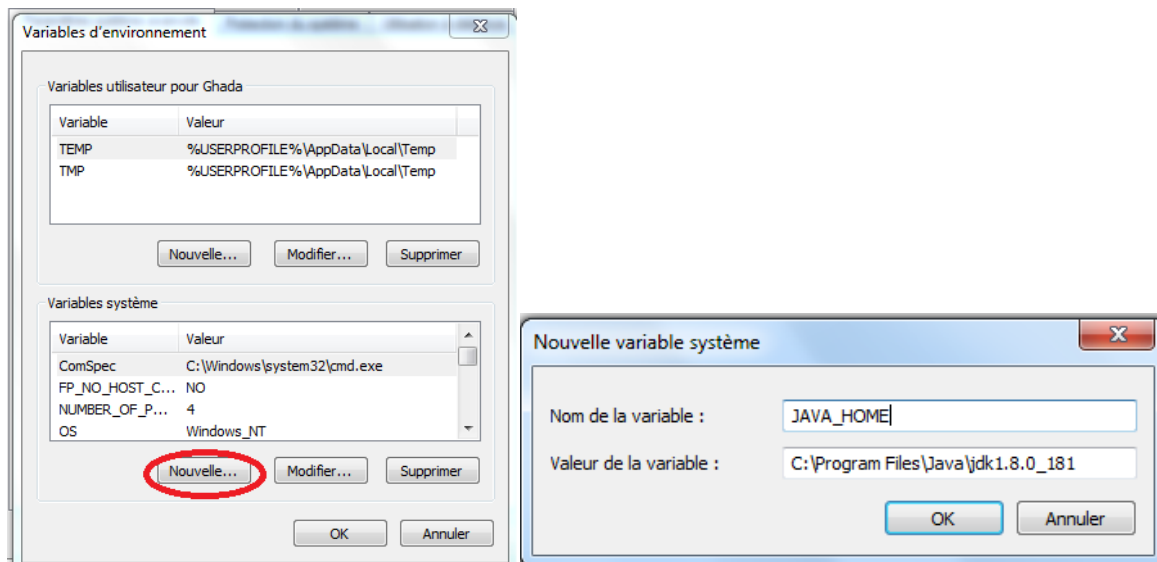
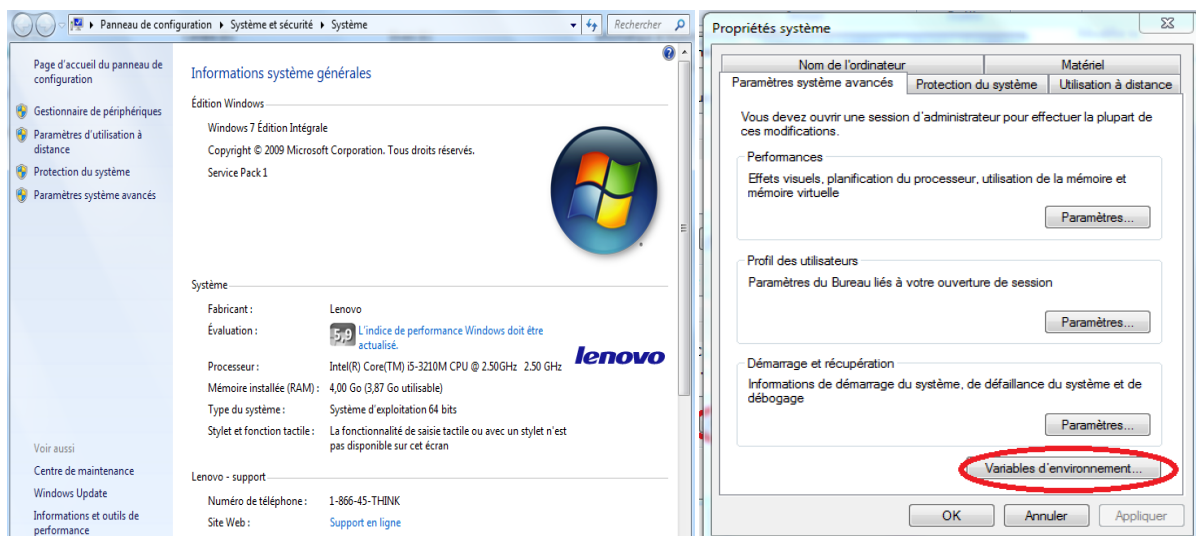
Ajouter 1 colonne(s) En fin de table En début de table Après idCompte Exécuter

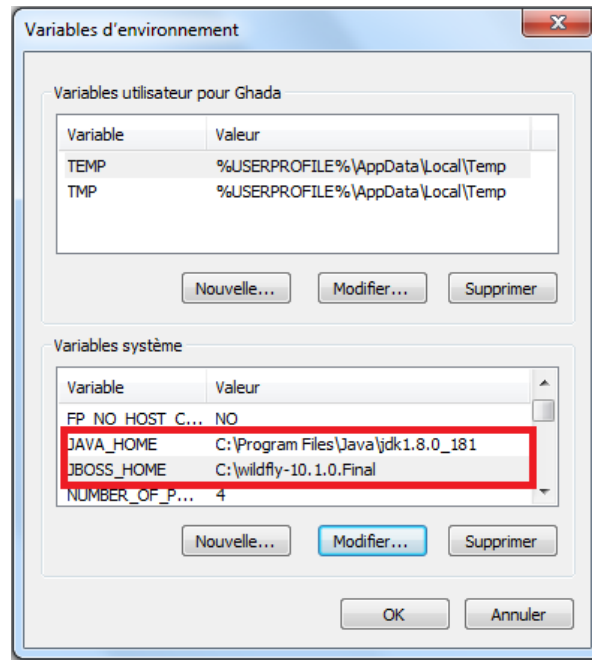
Relations

Colonne	Relation interne	Contrainte de clé étrangère (INNODB)
idCompte		
idProp		`gestioncb`.`proprietaire`.`idProp` ON DELETE RESTRICT ON UPDATE RESTRICT
dateCreation		Aucun index n'est défini !
solde		Aucun index n'est défini !

### III. Création d'une Source de Données JTA

1. Aller Panneau de configuration/Système et sécurité / Système / Paramètres système avancés :





## 2. Création d'un user management Wildfly

Allez dans wildfly/bin, exécuter le fichier : add-user.bat ; ajouter l'utilisateur adminGhada

```
C:\Windows\system32\cmd.exe
C:\wildfly-10.1.0.Final\bin>add-user.bat

What type of user do you wish to add?
a) Management User <mgmt-users.properties>
b) Application User <application-users.properties>
(a): a

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : adminGhada
Password recommendations are listed below. To modify these restrictions edit the
add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admin, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
Re-enter Password :
What groups do you want this user to belong to? <Please enter a comma separated list, or leave blank for none>[ ]:
About to add user 'adminGhada' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'adminGhada' to file 'C:\wildfly-10.1.0.Final\standalone\configuration\mgmt-users.properties'
Added user 'adminGhada' to file 'C:\wildfly-10.1.0.Final\domain\configuration\mgmt-users.properties'
Added user 'adminGhada' with groups to file 'C:\wildfly-10.1.0.Final\standalone\configuration\mgmt-groups.properties'
Added user 'adminGhada' with groups to file 'C:\wildfly-10.1.0.Final\domain\configuration\mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? yes
To represent the user add the following to the server-identities definition <secret value="Z2hhZGFAMjAxOQ==" />
Appuyez sur une touche pour continuer...
```

## Ghada Feki

### Java EE

#### 3. Ajout du driver mysql

Télécharger une version du fichier driver mysql : com.mysql.jdbc\_5.1.5.jar

Aller dans : ..\wildfly-10.1.0.Final\modules\system\layers\base\com

Créer l'arborescence suivante : com / mysql / main

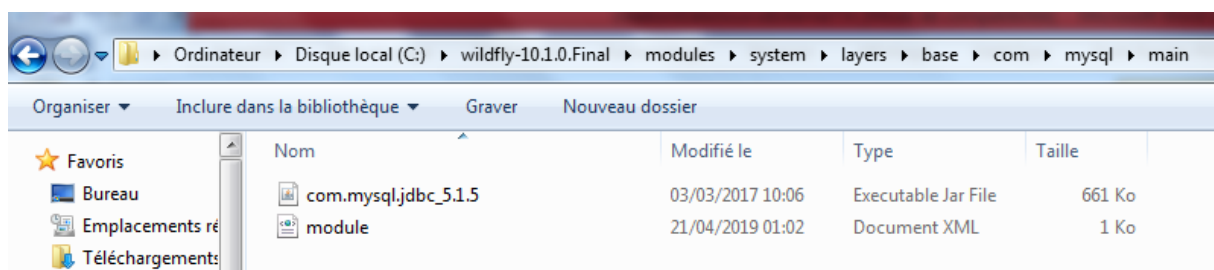
Placer dans le dossier /main les deux fichiers suivants:

- Le fichier driver de MYSQL : com.mysql.jdbc\_5.1.5.jar
- Le fichier module.xml suivant :

```
<module xmlns="urn:jboss:module:1.3" name="com.mysql">
<resources>
<resource-root path="com.mysql.jdbc_5.1.5.jar"/>
</resources>
<dependencies>
<module name="javax.api"/>
<module name="javax.transaction.api"/>
</dependencies>
</module>
```

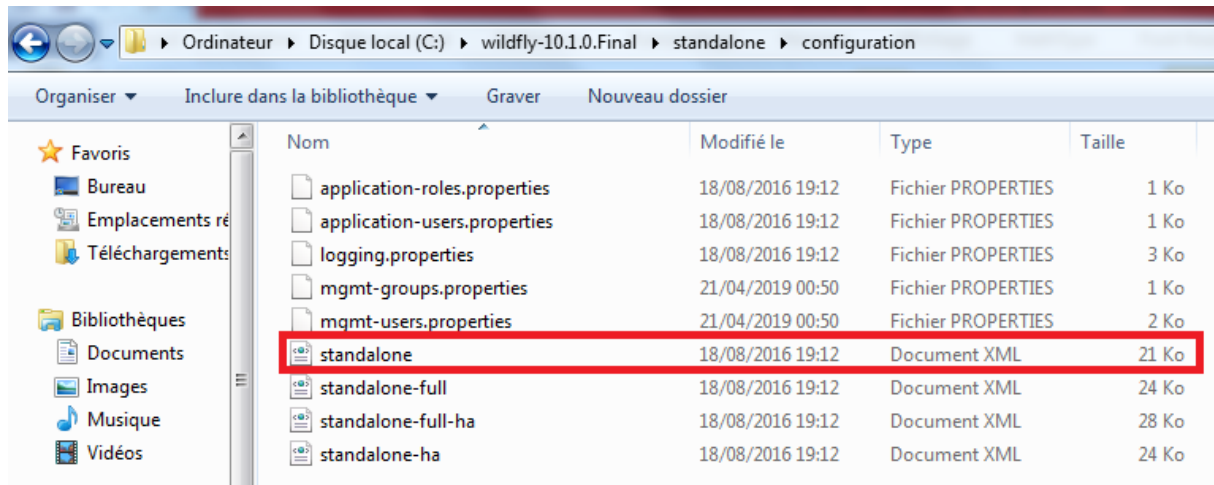
A la fin voici la structure de votre répertoire :

C:\wildfly-10.1.0.Final\modules\system\layers\base\com\mysql\main :



#### 4. Ajout du driver MYSQL dans Wildfly :

Dans le répertoire : ...\\wildfly-10.1.0.Final\\standalone\\configuration, existe le fichier xml standalone.xml.



Editer ce fichier et chercher la balise « drivers »: <drivers>

```
<driver name="h2" module="com.h2database.h2"><xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class></driver>
```

---ajouter ici le driver de mysql</drivers>

Vous remarquez l'absence du driver de mysql , ajouter la balise xml suivante :

```
<driver name="mysql" module="com.mysql"><driver-class>com.mysql.jdbc.Driver</driver-class></driver>
```

Vous devez avoir maintenant:

```
<drivers>
<driver name="h2" module="com.h2database.h2"><xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class></driver>
<driver name="mysql" module="com.mysql"><driver-class>com.mysql.jdbc.Driver</driver-class></driver>
```

5. Création d'une source de données externe

Démarrez votre console d'administration de wildfly

```
ca. C:\Windows\system32\cmd.exe - standalone.bat
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Ghada>cd..
C:\Users>cd..
C:\>cd C:\wildfly-10.1.0.Final\bin
C:\wildfly-10.1.0.Final\bin>standalone.bat
Calling "C:\wildfly-10.1.0.Final\bin\standalone.conf.bat"
Setting JAVA property to "C:\Program Files\Java\jdk1.8.0_181\bin\java"
=====
JBoss Bootstrap Environment

JBOSS_HOME: "C:\wildfly-10.1.0.Final"

JAVA: "C:\Program Files\Java\jdk1.8.0_181\bin\java"

JAVA_OPTS: "-Dprogram.name=standalone.bat -Xms64M -Xmx512M -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256m -Djava.net.preferIPv4Stack=true -Djboss.modules.system.pkgs=org.jboss.byteman"
=====
01:27:38,466 INFO [org.jboss.modules] <main> JBoss Modules version 1.5.2.Final
01:27:43,198 INFO [org.jboss.msc] <main> JBoss MSC version 1.2.6.Final
01:27:44,212 INFO [org.jboss.as] <MSC service thread 1-7> WFLYSRV0049: WildFly Full 10.1.0.Final <WildFly Core 2.2.0.Final> starting
01:28:17,678 INFO [org.jboss.as.server] <Controller Boot Thread> WFLYSRV0039: Creating http management service using socket-binding <management-http>
01:28:18,327 INFO [org.xnio] <MSC service thread 1-5> XNIO version 3.4.0.Final
01:28:18,360 INFO [org.xnio.nio] <MSC service thread 1-5> XNIO NIO Implementation Version 3.4.0.Final
01:28:18,566 INFO [org.jboss.as.clustering.infinispan] <ServerService Thread Pool -- 46> WFLYCLINF0001: Activating Infinispan subsystem.
```

```
ca. C:\Windows\system32\cmd.exe - standalone.bat
ce thread 1-5> ISPN000128: Infinispan version: Infinispan 'Chakra' 8.2.4.Final
01:28:56,589 INFO [org.infinispan.configuration.cache.EvictionConfigurationBuilder] <ServerService Thread Pool -- 75> ISPN000152: Passivation configured without an eviction policy being selected. Only manually evicted entities will be passivated.
01:28:56,612 INFO [org.infinispan.configuration.cache.EvictionConfigurationBuilder] <ServerService Thread Pool -- 75> ISPN000152: Passivation configured without an eviction policy being selected. Only manually evicted entities will be passivated.
01:28:56,632 INFO [org.infinispan.configuration.cache.EvictionConfigurationBuilder] <ServerService Thread Pool -- 74> ISPN000152: Passivation configured without an eviction policy being selected. Only manually evicted entities will be passivated.
01:28:56,429 INFO [org.infinispan.configuration.cache.EvictionConfigurationBuilder] <ServerService Thread Pool -- 72> ISPN000152: Passivation configured without an eviction policy being selected. Only manually evicted entities will be passivated.
01:28:56,691 INFO [org.infinispan.configuration.cache.EvictionConfigurationBuilder] <ServerService Thread Pool -- 74> ISPN000152: Passivation configured without an eviction policy being selected. Only manually evicted entities will be passivated.
01:28:56,717 INFO [org.infinispan.configuration.cache.EvictionConfigurationBuilder] <ServerService Thread Pool -- 72> ISPN000152: Passivation configured without an eviction policy being selected. Only manually evicted entities will be passivated.
01:29:00,462 INFO [org.wildfly.extension.undertow] <MSC service thread 1-2> WFLYUT0006: Undertow HTTPS listener https listening on 127.0.0.1:8443
01:29:02,164 INFO [org.jboss.ws.common.management] <MSC service thread 1-2> JBWS022052: Starting JBossWS 5.1.5.Final <Apache CXF 3.1.6>
01:29:03,172 INFO [org.jboss.as] <Controller Boot Thread> WFLYSRV0060: Http management interface listening on http://127.0.0.1:9990/management
01:29:03,175 INFO [org.jboss.as] <Controller Boot Thread> WFLYSRV0051: Admin console listening on http://127.0.0.1:9990
01:29:03,180 INFO [org.jboss.as] <Controller Boot Thread> WFLYSRV0025: WildFly Full 10.1.0.Final <WildFly Core 2.2.0.Final> started in 89640ms - Started 332 of 578 services (393 services are lazy, passive or on-demand)
```

http://localhost:9990/console/

WildFly

Messages: 0 adminGhada

Home

Deployments


Configuration

Runtime

Access Control

Patching

WildFly

 **Deployments**  
Add and manage deployments


Deploy an Application

Start

Deploy an application to the server

1. Use the 'Add Deployment' wizard to deploy the application

2. Enable the deployment

 **Configuration**  
Configure subsystem settings

Create a Datasource

Start

Define a datasource to be used by deployed applications. The proper JDBC driver must be deployed and registered.


1. Select the Datasources subsystem

2. Add a Non-XA or XA datasource

3. Use the 'Create Datasource' wizard to configure the datasource settings

Create a JMS Queue

Start

 **Runtime**  
Monitor server status


Monitor the Server

Start

View runtime information such as server status, JVM status, and server log files.

1. Select the server

2. View log files or JVM usage

 **Access Control**  
Manage user and group permissions for management operations

Assign User Roles

Start

Assign roles to users or groups to determine access to system resources.

1. Add a new user or group

2. Assign one or more roles to that user or group

10.1.0.Final Tools Settings

WildFly

Messages: 0 adminGhada

Home

Deployments

Configuration

Runtime

Access Control

Patching

Configuration

Subsystems

Interfaces

Socket Binding

Paths

System Properties

Subsystem (28)

JCA

Datasources

Resource Adapters

Mail

Transactions

Type

Non-XA

XA

Datasource

ExampleDS

Add

Non-XA Datasources

Manage Non-XA datasources, which are used for applications which do not use transactions, or applications which use transactions with a single database.



Create Datasource



Choose Datasource

- ☐ Custom
- ☐ H2 Datasource
- ☐ PostgreSQL Datasource
- ☒ MySQL Datasource
- ☐ Oracle Datasource
- ☐ Microsoft SQLServer Datasource
- ☐ IBM DB2 Datasource
- ☐ Sybase Datasource

Cancel

« Back

Next »

Create Datasource



Step 1/3: Datasource Attributes

[Need Help?](#)

Name \*:

gestioncbJTA

JNDI Name \*:

java:/datasources/gestioncbJTA

Required fields are marked with an asterisk (\*).

Cancel

« Back

Next »

Create Datasource

Step 2/3: JDBC Driver

Select one of the installed JDBC driver. Don't see your driver? Please make sure it's deployed as a module and properly registered.

Specify Driver

Detected Driver

Need Help?

Name\*:

mysql

Module Name\*:

com.mysql

Driver Class:

com.mysql.jdbc.Driver

Major Version:

0

Minor Version:

0

Required fields are marked with an asterisk (\*).

Cancel

<< Back

Next >>

Create Datasource



Step 3/3: Connection Settings

[Need Help?](#)

Connection URL\*:

jdbc:mysql:///localhost/gestioncb

Username:

root

Password:

Security Domain:

Required fields are marked with an asterisk (\*).

Cancel

« Back

Next »

## Create Datasource



### Summary

Please verify your settings. After the datasource is created you can test the connection by selecting the datasource in the configuration or runtime section and press 'Test Connection'.

Name: gestioncbJTA

JNDI Name: java:/datasources/gestioncbJTA

Connection URL: jdbc:mysql://localhost/gestioncb

Username: root

Password:

Cancel

<< Back

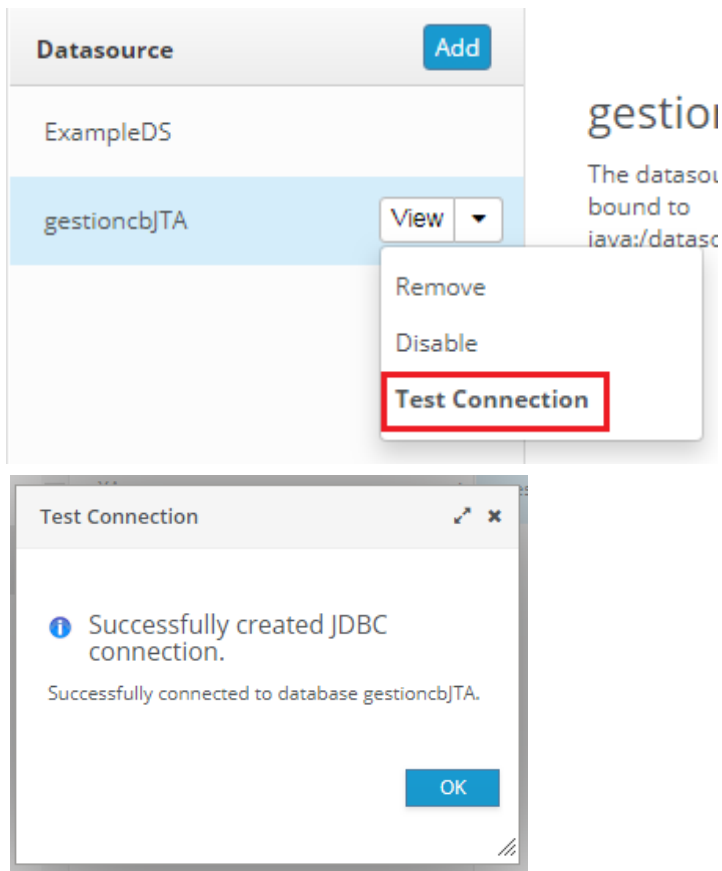
Finish

**WildFly**

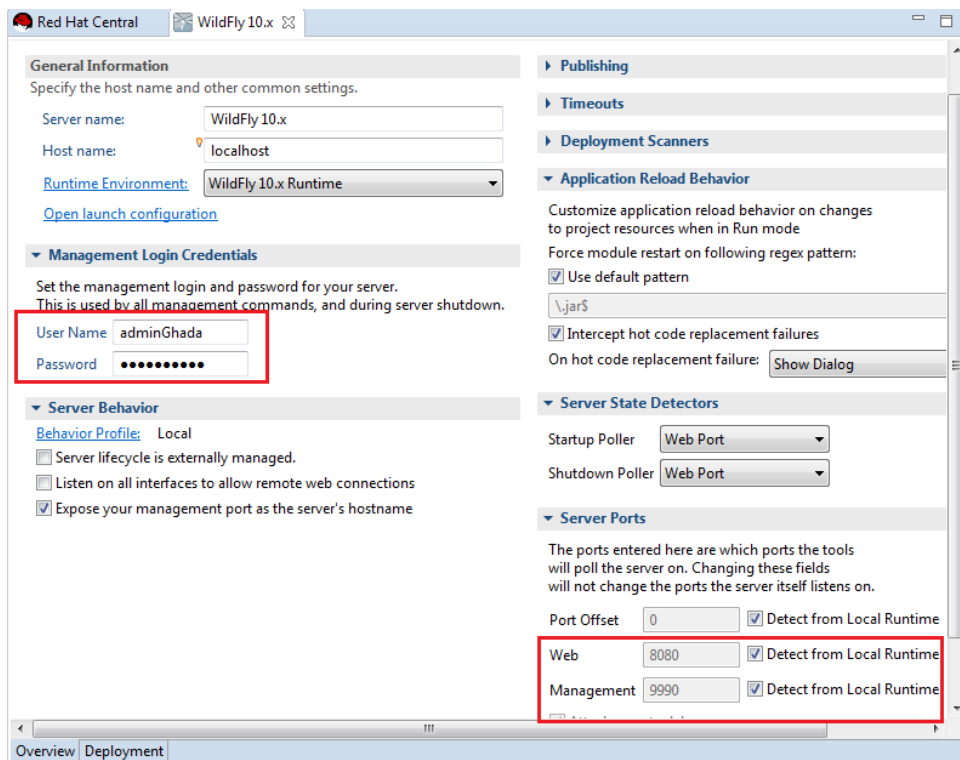
Home Deployments **Configuration** Runtime Access Control Patching

Configuration	Subsystem (28)	Type	Datasource	Add
Subsystems >	Q	Non-XA	ExampleDS	
Interfaces	JCA	XA	gestioncbJTA	
Socket Binding	Datasources >			
Paths	Resource Adapters >			
System Properties	Mail >			
	Transactions			

## Ghada Feki Java EE

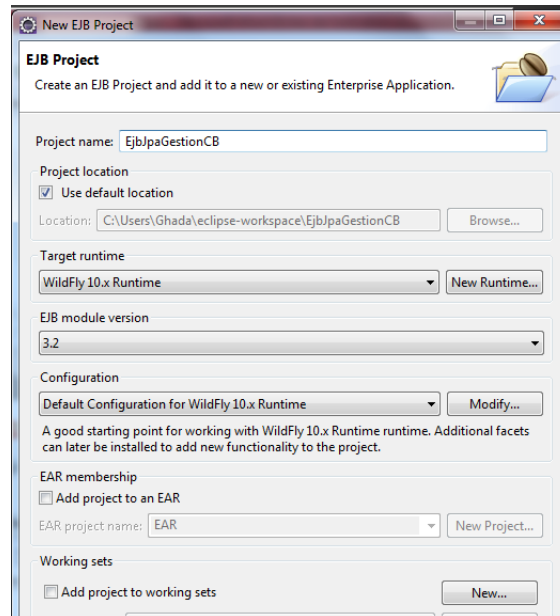


Sur Eclipse, vérifier la configuration du serveur Wildfly.

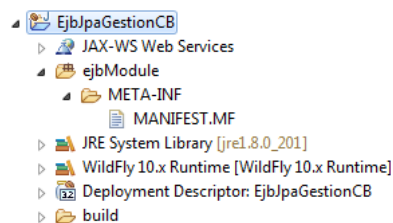


## IV. Accès aux données avec l'API de persistance JPA

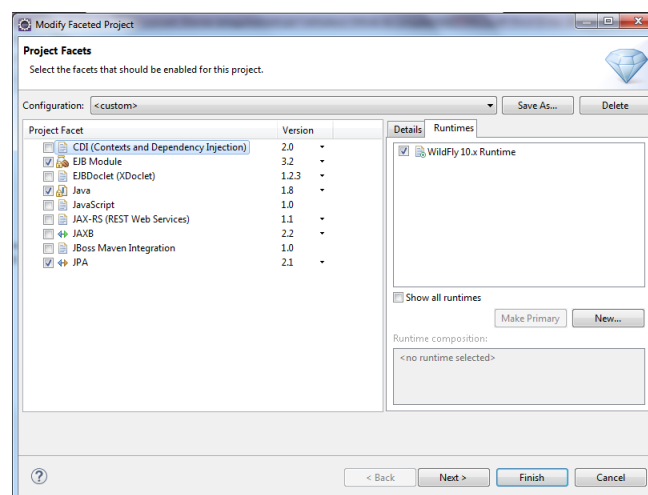
### 1. Créer un nouveau projet EJB



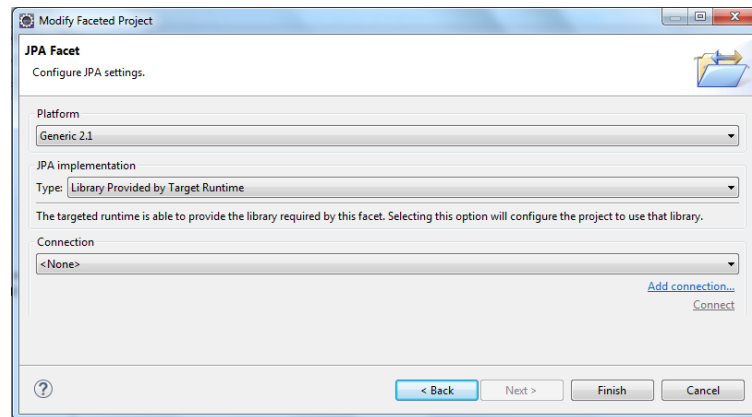
### 2. Vérifier que le fichier persistence.xml n'existe pas sous ejbModule/META-INF



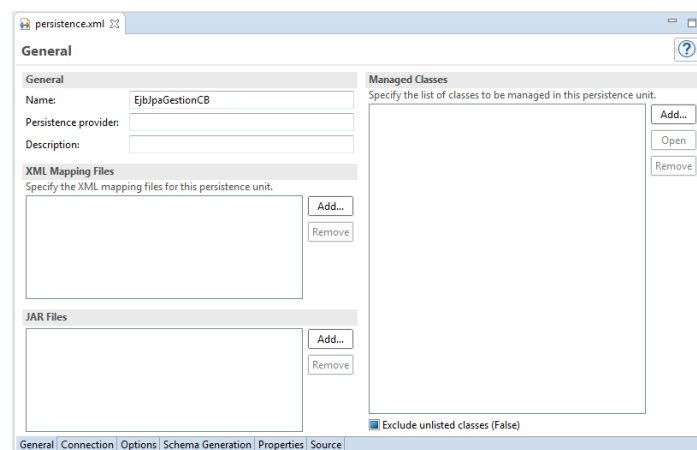
### 3. Convertir le Projet EJB déjà créé en Projet JPA : bouton droit , configurer, convert to JPA Project.



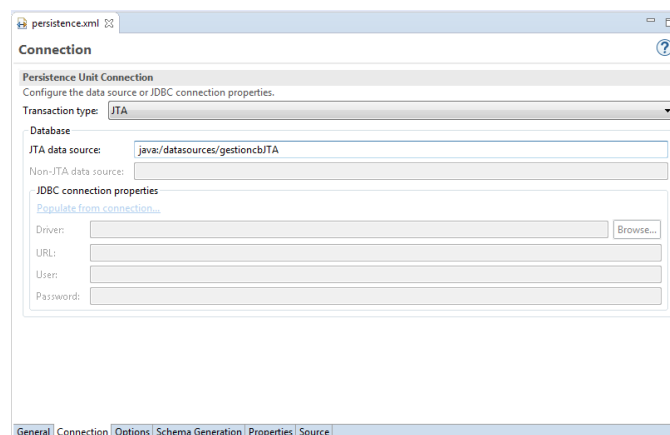
4. Configurer la facette du projet pour travailler avec une source de données externe JTA (gérée par le serveur) : Next



5. Vérifier l'ajout du fichier persistence.xml

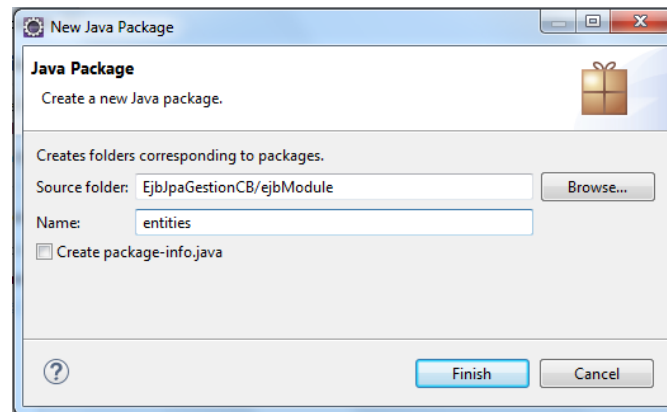


Editer ce fichier graphiquement. Aller à l'onglet Connection, configurer la connexion (transaction Type : JTA) et indiquer le nom de la source de données créées auparavant avec Wildfly.





6. Créer le package entitiessousEjbModule.



7. Créer sous ce package les deux classes entités Compte et Proprietaire (New / JPAEntity).
8. Annoter ces deux entités (@Table , @Id).

```
package entities;

import java.io.Serializable;
import java.sql.Date;

import javax.persistence.*;

@Entity
@Table(name="compte", schema = "gestioncb")
public class Compte implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    private int idCompte;
    private int idProp;
    private Date dateCreation;
    private double solde;

    public Compte() {
        super();
    }
}
```

```
}

    public Compte(int idCompte, int idProp, Date dateCreation, double solde) {
        super();
        this.idCompte = idCompte;
        this.idProp = idProp;
        this.dateCreation = dateCreation;
        this.solde = solde;
    }

    public int getIdCompte() {
        return idCompte;
    }

    public void setIdCompte(int idCompte) {
        this.idCompte = idCompte;
    }

    public int getIdProp() {
        return idProp;
    }

    public void setIdProp(int idProp) {
        this.idProp = idProp;
    }

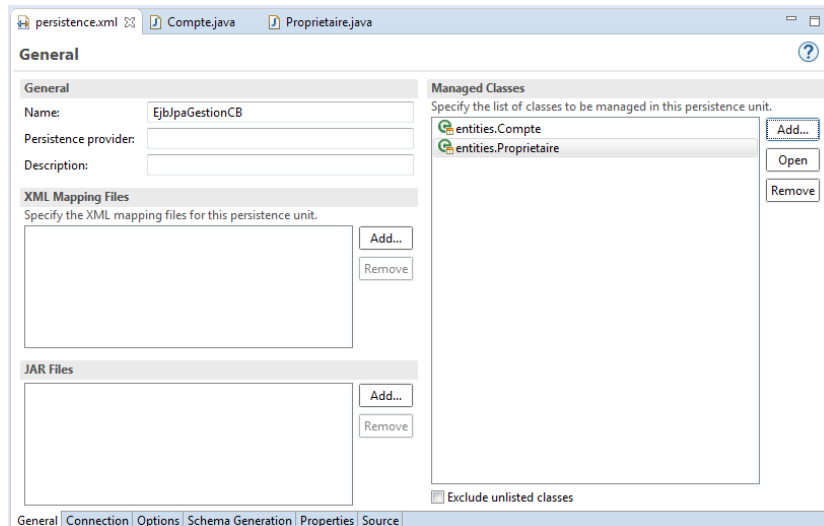
    public Date getDateCreation() {
        return dateCreation;
    }

    public void setDateCreation(Date dateCreation) {
        this.dateCreation = dateCreation;
    }

    public double getSolde() {
        return solde;
    }

    public void setSolde(double solde) {
        this.solde = solde;
    }
}
```

9. Configurer le fichier persistence.xml en ajoutant les classes gérées

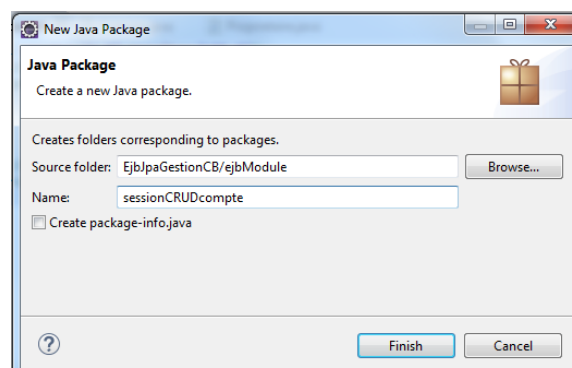


10. Vérifier que le fichier persistence.xml est bien configuré (source).

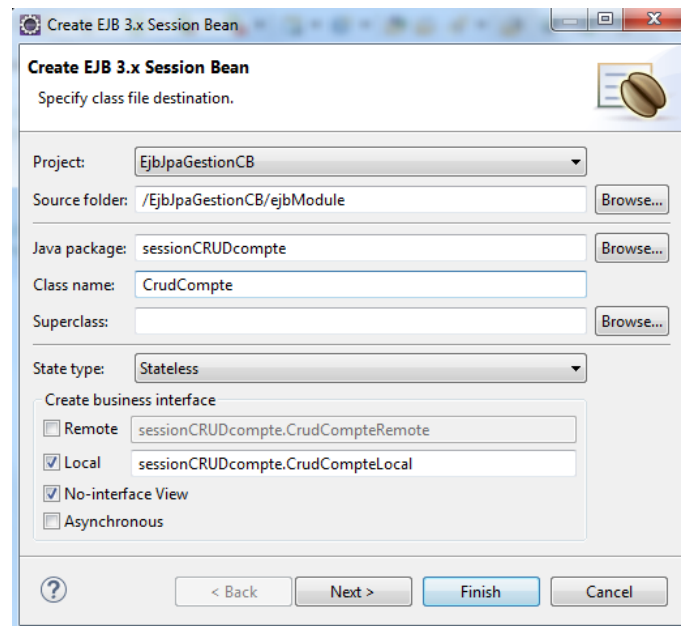
```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="EjbJpaGestionCB" transaction-type="JTA">
    <jta-data-source>java:/datasources/gestioncbJTA</jta-data-source>
    <class>entities.Compte</class>
    <class>entities.Proprietaire</class>
    <exclude-unlisted-classes>false</exclude-unlisted-classes>
  </persistence-unit>
</persistence>
```

## V. Création d'un EJB Session

1. Créer le package contenant le EJB session réalisant CRUD Compte.



## 2. Créer l'EJB session CrudCompte avec une interface locale



```
Package sessionCRUDcompte;

import java.util.List;

import javax.ejb.LocalBean;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.TypedQuery;

import entities.Compte;

@Stateless
@LocalBean
Public class CrudCompte implements CrudCompteLocal {

    // initierle contexte de persistance
    @PersistenceContext(unitName = "EjbJpaGestionCB")
    // déclarer l'entityManager qui vagérer la source de données externe jta
    private EntityManager entityManager ;

    public CrudCompte() {

    }

}
```

```
@Override
public void ajouterCompte(Compte c) {
    entityManager.persist(c);
}

@Override
public Compte rechercherCompte(int idCompteR) {
    Compte c=entityManager.find(Compte.class, idCompteR);
    return c;
}

@Override
public void deposer(int idCompte, double montant) {

    //à compléter
}

@Override
public void retirer(int idCompte, double montant) {

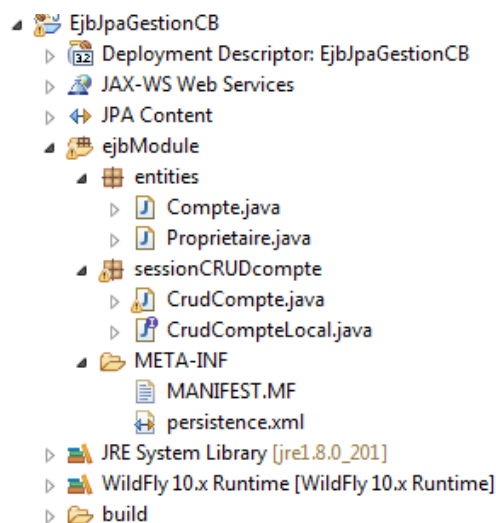
    //à compléter
}

@Override
public List<Compte> listerComptes() {
    TypedQuery <Compte> query = entityManager.createQuery("SELECT c FROM
Compte c", Compte.class);
    List<Compte> listComptes = query.getResultList();
    return listComptes;
}

@Override
public void modifier(int idCompteM) {
    Compte c=entityManager.find(Compte.class, idCompteM);
    entityManager.merge(c);
}

@Override
public boolean supprimerCompte(int idCompteS) {
    Compte c=entityManager.find(Compte.class, idCompteS);
```

```
        if (c != null) {  
            entityManager.remove(c);  
            return true;  
        }  
        else  
            return false;  
    }  
}
```



## VI. Création d'un client web

Reprendre le même travail du TP5 concernant le projet Web Dynamique qui représente l'application cliente du projet EJB développé (il faut suivre les étapes nécessaires pour l'exploitation de l'EJB).

## VII. Travail demandé

Effectuer toutes les étapes présentes sur le fascicule du TP présent.

Ajouter la page jsp et la servlet de l'objectif modifier un compte.

Développer les méthodes « débiter » et « créditer » (JPQL).

Ajouter l'objectif « lister les comptes ordonnés par propriétaire » (JPQL) à cette solution Java EE.

Refaire le même travail pour l'EJB Entity Proprietaire.

Modifier la méthode ajouterCompte pour qu'elle vérifie l'existence du propriétaire avant l'ajout.