Premiers pas JPA

Objectif

Faire persiter une première entité

Comprendre ce que fait JPA

Utiliser les annotations :

```
@Entity
@Id
@GeneratedValue
@Embedded
@Embeddable
```

Ce que l’on veut obtenir

Persister un objet Person avec son adresse (Address)

Vérifier la présence de la table personne avec les champs de Address

La présence d’un enregistrement dans la table Person

Action

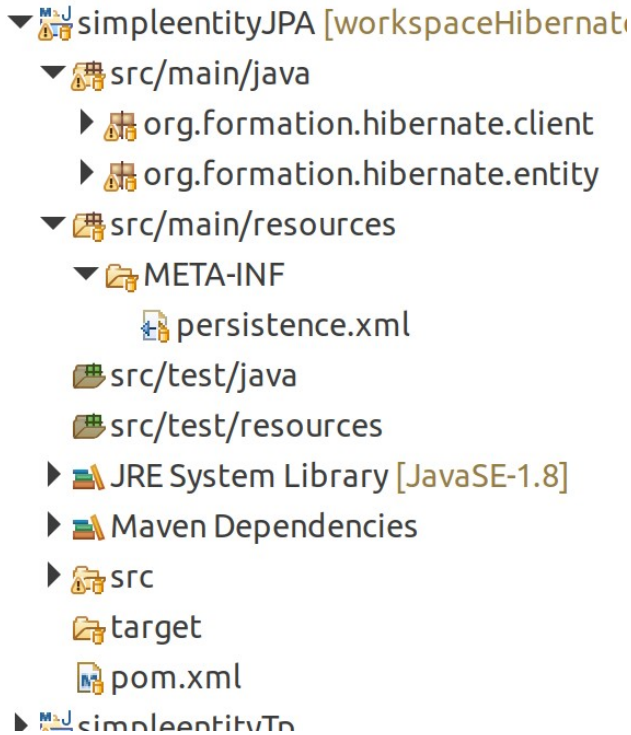
Créer une classe Person avec un nom et un age (int).

Générer le toString

Créer une classe Main avec une méthode main

Partie 1 : Configuration du projet

Créer un projet maven simple comme suit



Configurer le POM

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>org.formation</groupId>
    <artifactId>one2oneJPA11</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <dependencies>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-
java</artifactId>
            <version>5.1.31</version>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
```

```
            <version>5.4.2.Final</version>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-
entitymanager</artifactId>
            <version>5.4.2.Final</version>
        </dependency>
        <dependency>
            <groupId>com.h2database</groupId>
            <artifactId>h2</artifactId>
            <version>1.4.190</version>
        </dependency>
    </dependencies>
    <properties>

    <maven.compiler.source>11</maven.compiler.source>

    <maven.compiler.target>11</maven.compiler.target>
    </properties>
</project>
```

Configurer le pesistence.xml

Completer le persistence.xml selon les commentaires

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
    version="2.0" xmlns="http://java.sun.com/xml/ns/persistence">

    <persistence-unit name="pu" transaction-
type="RESOURCE_LOCAL">


        <properties>

            <!-- Database connection settings -->
            <!-- SQL dialect -->
            <!-- Create/update tables automatically using
mapping metadata -->
            <!-- Pretty print the SQL in the log file and
console -->
        </properties>

    </persistence-unit>
</persistence>
```

Partie 2 : Codage de l’Entity Person

Créer une classe Person avec les attributs suivant

id en Long

name en String

générer le toString

Question : dans quel package ?

Partie 3 : Persister l’Entity Person

Créer une classe TestClientPersist avec une méthode main dans le package org.formation.hibernate.client

Coder la structure suivante.

Avant de coder expliquer ce que fait ce code

```java
package org.formation.hibernate.client;

import javax.persistence.EntityManager;

public class TestClientPersist {
    public static void main(String[] args) {

        EntityManagerFactory emf = Persistence.createEntityManagerFactory("pu");
        EntityManager em = emf.createEntityManager();
        EntityTransaction txn = em.getTransaction();
        try {
            txn.begin();

// you code here !!!

            txn.commit();
        } catch (Exception e) {
            if (txn != null) {
                txn.rollback();
            }
            e.printStackTrace();
        } finally {
            if (em != null) {
                em.close();
            }
            if (emf != null) {
                emf.close();
            }
        }
    }
}
```

Ajouter le code pour persister un objet Person

```
EntityManagerFactory emf = Persistence.createEntityM
EntityManager em = emf.createEntityManager();
EntityTransaction txn = em.getTransaction();
try {
    txn.begin();
    Customer c = new Customer();
    c.setName("Bobby");
    em.persist(c);

    txn.commit();
} catch (Exception e) {
    if (txn != null) {
        txn.rollback();
    }
    e.printStackTrace();
} finally {
```

Tester et vérifier en base

Partie 4 : Ajouter la notion d'adresse

Créer une classe Address avec les attributs suivants

```
public class Address {

    private String street;
    private String city;
    private Long zipcode;
```

Ajouter l'annotation

Dans Person ajouter l'attribut address `@Embeddable`

l'annoter avec `@Embedded`

Persister une instance de Person avec son adresse et vérifier les colonnes de la table person

Partie 5 : Récupération et mise à jour

récupération

Noter l'id en base de l'enregistrement

récupérer l'enregistrement par son id avec la méthode find

modification

dans le même code, modifier le nom de la personne ; inutile de sauvegarder ou d'utiliser persist !

Pourquoi ?

Partie 6 : Suppression

Noter l'id en base de l'enregistrement

récupérer l'enregistrement par son id avec la méthode find

Supprimer avec la méthode remove

Dans tous les cas, bien regarder les sorties sur la console

Partie 7 : create create-drop et update

Dernier effort ! Utiliser testClientPersist et le lancer avec les différentes configurations de persistence.xml

Avec les valeurs

```
create-drop
create
update
<!-- Create/update tables automatically using mapping metadata -->
<property name="hibernate.hbm2ddl.auto" value="create-drop" />
```

Bravo !