

1. JPA algemeen.

1.1. Eén domeinklasse en drie objecten.

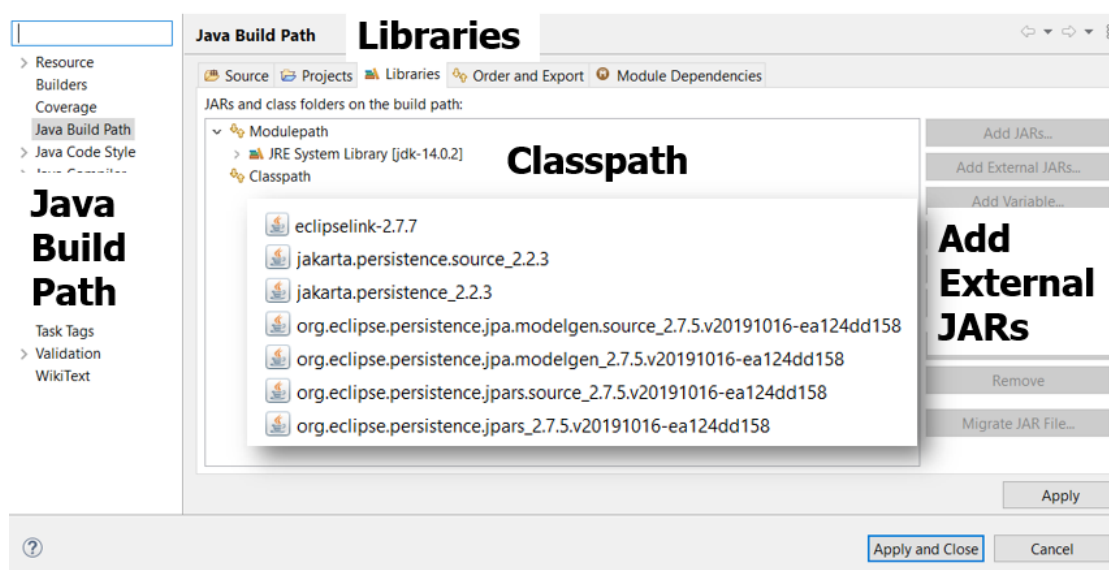
We maken één domeinklasse **Docent** aan. We instantiëren drie docent objecten en zorgen ervoor dat die in de databank **schooldb** in de tabel **Docenten** worden bijgehouden.

Van een docent houden we de **docentNr** (int), **voornaam** (String) , **familienaam** (String) en **wedde** (BigDecimal) bij.

Maak een Java Application project aan.

Voeg de EclipseLink (JPA2.1) jars toe aan het project (Zie chamilo [jpajars zip met alle external jars voor JPA](#))

->project properties/Java-Build-Path/Libraries/Add External JARs bij het classpath).



Maak de **Docent** klasse aan in het domein.

Voeg de gepaste annotations toe.

We voorzien als primary key in de database een autonummering. In de docentklasse voorzien we hiervoor een attribuut met naam **id**.

id is de primary key en is autonummer in de database.

`@GeneratedValue(strategy = GenerationType.IDENTITY)`

We willen dat het attribuut **docentnr** met een veld **PERSONEELSNR** wordt gemapped.

Voorzie ook setters, getters, constructor en toString methoden. Zorg ervoor dat twee docentobjecten gelijk zijn als hun docentnr gelijk is.

Voorzie ook een (protected) default constructor, verplicht voor een entity klasse.

Maak in MySQL workbench een lege database aan met de naam **schooldb**.

Voorzie User Name: **root** en Password: **rootp**

Maak een Persistence Unit.

In de src map maak je een nieuwe map META-INF.

In de META-INF map maak je een xml file met de naam **persistence.xml**.

Inhoud van het persistence.xml bestand:

```
<?xml version="1.0" encoding="UTF-8"?>

<persistence version="2.2" 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_2.xsd">
  <persistence-unit name="school" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <class>domein.Docent</class>
    <properties>
      <property name="javax.persistence.jdbc.url"
        value="jdbc:mysql://localhost:3306/schooldb?serverTimezone=UTC"/>
      <property name="javax.persistence.jdbc.user" value="root"/>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>
      <property name="javax.persistence.jdbc.password" value="root"/>
      <property name="javax.persistence.schema-generation.database.action" value="create"/>
    </properties>
  </persistence-unit>
</persistence>
```

Mogelijke waarden voor de property **javax.persistence.schema-generation.database.action** zijn none, create, drop-and-create, drop.

Maak een startup om de objecten aan te maken en persistent te maken.

Maak een nieuw package main aan.

Maak een java klasse MAINoef1 met de main methode in.

In de main methode:

Maak drie Docent objecten aan.

Maak een EntityManager aan.

gebruik hiervoor de EntityManagerFactory.

Een EntityManagerFactory instance maken vraagt veel tijd.

– Je maakt zo'n instance één keer per applicatie.

– Je houdt hem best bij via een singleton utility class.

Maak een package util aan. Plaats daar het singleton JPAUtil dat een EntityManagerFactory object herbergt. Je dient voor de creatie hiervan de persistence unit naam door te geven: **school**.

Met de createEntityManager van de factory krijg je dan een instance.

Start een transactie.


Persist de drie Docent objecten.

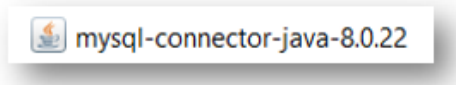
Commit de transactie.

Sluit de entityManager

Sluit de factory.

Voor je de applicatie kan runnen dien je ook nog de een jar toe te voegen aan het project, nodig voor de database drivers (zie chamilo [mysql-connector-java-8.0.22 client driver voor jdbc mysql connectie](#)).

 Classpath



Run de applicatie en bekijk het resultaat in de MySQLWorkBench.

1.2. Een object wijzigen.

Maak een java klasse MAINoef2 met de methode main in. Voorzie de nodige stappen om een nieuwe transactie uit te voeren. MAINoef1 is uitgevoerd. In de database zijn drie records.

Je haalt het docentobject met id 2 op en verhoogt zijn wedde met 200 euro.

Je kan selectief één object ophalen via de entitymanager door de find bewerking:

-->em.find(Docent.class, 2L)

Het eerste argument bepaalt de klasse van het object dat gezocht wordt en het tweede argument is een voorkomen van de primaire sleutel. Indien geen record gevonden wordt, krijgen we een null referentie.

Bekijk het resultaat in de DB.