Fiche MEMO création projet webapp (Jersey + Hibernate + Tomcat) à partir de Netbeans

- Netbeans > new project > Web Application

- Créer fichier : webapp > WEB-INF > web.xml
  - `webapp/WEB-INF/web.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<!--<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" x
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation=

  <display-name>JakartaDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.jsp</welcome-file>
    <welcome-file>default.htm</welcome-file>
  </welcome-file-list>


   <servlet>
        <servlet-name>Jersey Web app</servlet-name>
        <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
        <init-param>
            <param-name>jersey.config.server.provider.packages</param-name>
            <param-value>com.poe.documentationonline.api</param-value>
        </init-param>
        <init-param>
            <param-name>jersey.config.server.provider.scanning.recursive</param-name>
            <param-value>true</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>Jersey Web app</servlet-name>
        <url-pattern>/api/*</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>


</web-app>
```

- corriger bug version dans pom.xml

  remplacer la version dans:

```
<artifactId>maven-war-plugin</artifactId>
<version>2.3</version>
```

  par:

```
<artifactId>maven-war-plugin</artifactId>
<version>3.1.0</version>
```

- Ajouter les dependences de Jersey dans `pom.xml`

```
<!--JAX-RS avec Jersey-->
<dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-web-api</artifactId>
    <version>7.0</version>
    <scope>provided</scope>
</dependency>

<dependency>
    <groupId>org.glassfish.jersey.containers</groupId>
    <artifactId>jersey-container-servlet</artifactId>
    <version>2.28</version>
</dependency>

<dependency>
    <groupId>org.glassfish.jersey.inject</groupId>
    <artifactId>jersey-hk2</artifactId>
    <version>2.28</version>
</dependency>

<!-- pour le format JSON -->
<dependency>
    <groupId>org.glassfish.jersey.media</groupId>
    <artifactId>jersey-media-json-jackson</artifactId>
    <version>2.28</version>
</dependency>

<!-- pour le format XML -->
<dependency>
    <groupId>org.glassfish.jersey.media</groupId>
    <artifactId>jersey-media-jaxb</artifactId>
    <version>2.28</version>
</dependency>

<!--org.glassfish.jersey.server.ContainerException: java.lang.NoClassDefFoundError: javax/xml/bind/annotation/XmlElement-->
<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->
<dependency>
    <groupId>javax.xml.bind</groupId>
    <artifactId>jaxb-api</artifactId>
    <version>2.3.1</version>
</dependency>
<dependency>
    <groupId>org.glassfish.jaxb</groupId>
    <artifactId>jaxb-runtime</artifactId>
    <version>2.3.1</version>
</dependency>
```

▼ Créer une classe API contenant les routes de l'API

com.poe.documentationonline.api.DocApi.java

```
package com.poe.documentationonline.api;

import com.poe.documentationonline.dao.CategoryDAO;
import com.poe.documentationonline.entity.Category;
import javax.ws.rs.GET;
import javax.ws.rs.Path;

@Path("doc")
public class DocApi {


    @GET()
    public String test(){

        Category cat1 = new Category();
        cat1.setName("HTML");
        CategoryDAO.create(cat1);
        return "Hello";
    }

}
```

▼ Configurer le nom du package contenant la classe Java de l'API Jersey

`webapp/WEB-INF/web.xml`

```
<param-value>com.poe.documentationonline.api</param-value>
```

▼ Ajouter dans le pom.xml les dependences pour la connexion à la base de données

`pom.xml`

```
<!-- Database -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>5.5.4.Final</version>
</dependency>

<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.29</version>
</dependency>
```

▼ Créer le fichier de configuration de Hibernate

`src/main/resources/META-INF/persistence.xml`

attentions au points suivants:

- `<property name="hibernate.hbm2ddl.auto" value="create" />`
- Le nom de l'unité qu'il faudra réutiliser plus tard : `<persistence-unit name="doc">`
- Le nom de la base de données dans : `<property name="hibernate.connection.url" value="jdbc:mysql://localhost:3306/documentationonline" />`
- login/password du compte

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<persistence 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.x
             version="2.2">
    <persistence-unit name="doc">
        <properties>
            <property name="hibernate.hbm2ddl.auto" value="create" />
            <!-- database connection -->
            <property name="hibernate.connection.driver_class" value="com.mysql.cj.jdbc.Driver" />
            <property name="hibernate.connection.url" value="jdbc:mysql://localhost:3306/documentationonline" />
            <property name="hibernate.connection.user" value="root" />
            <property name="hibernate.connection.password" value="mypassword" />
            <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL57Dialect" />
            <property name="hibernate.show_sql" value="true" />
        </properties>
    </persistence-unit>
</persistence>
```

▼ Créer les fichiers Java

▼ Exemple Entity

```
package com.poe.documentationonline.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Category {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(columnDefinition = "INTEGER")
    private Long id;

    private String name;

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }



}
```

- ▼ Exemple DAO

```
package com.poe.documentationonline.dao;

import com.poe.documentationonline.entity.Category;
import com.poe.documentationonline.entity.jpa.EntityManagerSingleton;
import javax.persistence.EntityManager;
import javax.persistence.EntityTransaction;


public class CategoryDAO {

        public static void create(Category c) {
        EntityManager entityManager = EntityManagerSingleton.getEntityManager();

        EntityTransaction tx = entityManager.getTransaction();
        tx.begin();
        entityManager.persist(c);
        tx.commit();
    }
}
```

- ▼ EntityManagerSingleton -> penser à maj le nom de l'unité utilisé dans le fichier de config persistence.xml

```
package com.poe.documentationonline.entity.jpa;


import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

public class EntityManagerSingleton {

    private static EntityManager entityManager;

    public static EntityManager getEntityManager() {

        if(entityManager == null) {
            EntityManagerFactory emf = Persistence.createEntityManagerFactory("doc");
            entityManager = emf.createEntityManager();
        }

        return entityManager;
    }
}
```

▼ Créer la base de données dans MySQL

```
CREATE DATABASE monprojet;
```

▼ Builder et deployer le projet

▼ Tester et vérifier que la table a été crée