Workshop N° 3

EJB Web Service et JPA

1. Communication EJB Web Service et JPA :

Modifier l'EJB session `ServiceRessource.java` afin de :

a) Déclarer un champ privé de type EntityManager

b) Annoter le gestionnaire d'entités avec l'annotation `@PersistenceContext`. L'annotation @PersistenceContext sur un champ de type EntityManager permet d'injecter un contexte de persistance. Cette annotation possède un attribut *unitName* qui précise le nom de l'unité de persistance. Il faut utiliser le nom de l'unité de persistance déclarée dans le fichier persistence.xml

```
@PersistenceContext(unitName="MyFirstEJBWS")

    private EntityManager em;
```

2. Implémentation des services web Rest

Modifiez l'EJB session `ServiceRessource.java` afin d'ajouter une méthode permettant de récupérer tous les apprenants de la table Learner de la base JPAbase.

Cette méthode doit avoir les caractéristiques suivantes:

Méthode	URL	Méthode http	Succès/Echec	Code Statut
getAllLearners() : retourne la liste de tous les apprenants dans la base en format json	RestApi/learners/	GET	Succès	200
			Succès avec liste vide --> renvoyer le message «`There is no saved learner in the DB`»	200

Le code de la méthode est comme suit :

```
@Path("learners")
    @GET
    @Produces(value= {MediaType.APPLICATION_JSON})
    public Response getAllLearners()
    {
       List<Learner> learners = new ArrayList<Learner>();

       TypedQuery<Learner> query =
       em.createNamedQuery("Learner.findAll",Learner.class);
       learners = query.getResultList();

       if(learners.size()!=0)
```

```
            if(learners.size()!=0)
            return Response.status(Status.FOUND).entity(learners).build();
      else
            return Response.status(Status.NOT_FOUND).entity("There is no saved
learner in the DB").build();
    }
```

Veuillez notez qu'une NamedQuery est tout simplement une *requête JPQL* associée à un *nom* (dans l'exemple ci-dessus le nom de la requête est `"Learner.findAll"`) qui permet de l'identifier. Une requête JPQL permet d'interagir avec le monde relationnel d'un point de vue « Objet ». On ne travaille pas directement sur les tables et les colonnes de la base de données, mais sur les classes et les attributs.

Toutes les requêtes NamedQuery doivent être déclarées au niveau de la classe entité JPA correspondante.

Lors de la génération automatique des entités JPA à partir des tables de la base, une requête `findAll` de type NamedQuery est générée automatiquement pour chaque entité.

Exp: Pour l'entité Learner.java

```
@Entity
@NamedQuery(name="Learner.findAll", query="SELECT l FROM Learner l")
public class Learner implements Serializable {
      private static final long serialVersionUID = 1L;

      @Id
      @GeneratedValue(strategy=GenerationType.IDENTITY)
      private int custId;

      private String city;

      private String email;

      private String name;

      public Learner() {
      }

      /* Le reste de la classe */
}
```

Cette requête permet de retourner tous les enregistrements de la table Learner.

Execution d'une requête de type NamedQuery

Pour exécuter une requête JPQL et récupérer la liste des enregistrements sous forme d'une liste d'objets du type correspondant, il faut :

- Commencer par créer une variable de récupération des objects :

```
List<Learner> learners = new ArrayList<Learner>();
```

- Créer la requête au niveau du gestionnaire des entités en utilisant la méthode **createNamedQuery(String** name, **Class**<T> resultClass) avec name le nom de la requête JPQL et resultClass la classe des objets à récupérer :

```
TypedQuery<Learner> query = em.createNamedQuery("Learner.findAll",Learner.class);
```

- Initialiser les paramètres de la requête si la requête contient des paramètres dynamiques (à voir dans la partie suivante du workshop)

- Exécuter la requête et récupérer la liste des résultats dans la variable initialement déclarée.

```
learners = query.getResultList();
```

3. Les services web Rest avec JPQL paramétrable

a- Ajouter une nouvelle requête NamedQuery au niveau de la classe Learner qui permet de récupérer la liste des apprenants ayant le même nom:

```
@NamedQuery(name="Learner.findAllByName", query="SELECT l FROM Learner l where l.name = ?1")
```

Il s'agit d'une requête paramétrable. Chaque paramètre de la requête est identifié par un «?» et un numéro.

b- Si une entité possède plus qu'une requête NamedQuery, alors toutes les requêtes doivent regroupées par {} et annotées par @NamedQueries

Le résultat dans notre cas sera comme suit:

```
@NamedQueries({
@NamedQuery(name="Learner.findAll", query="SELECT l FROM Learner l"),
@NamedQuery(name="Learner.findAllByName", query="SELECT l FROM Learner l where l.name = ?1")})
```

c- Créer un web service au niveau de la classe ressource qui permet de récupérer la liste des apprenants ayant le même nom.

d- Créer maintenant les requêtes JPQL (lorsque c'est nécessaire) et les web services suivants :

Méthode	URL
getLearnerByNameAndEmail(String name, String email)	MyRestRessource/learners/{name}/{email}
deleteLearnerById(int id)	MyRestRessource/learners/{id}

createLearner(Learner myLearner)	MyRestRessource/learners/