Développement WEB : JEE

TP 3 : JDBC

Objectifs :

- Se connecter à une base de données avec JAVA
- Développement des CRUD dans une application JAVA
- Découvrir l'API JDBC
- Utiliser les fichiers properties

Exercice 1 : Gestion des Clients d'une entreprise

On souhaite créer une simple application pour la gestion des clients. Pour ce faire, on propose la classe Client caractérisée par : id, nom et prénom

1- Développement de la couche Modèle (Entity)

- Créer la classe client dans le package "tp3.ex1.entities".

2- Développement de la couche de données

Créer la table client dans une base de données nommée "tp3JDBC".

3- Développement de la couche accès aux données DAO

- Créer la classe Connexion dans le package "tp3.ex1.dao"
- Créer une interfaces IDao dans le package "tp3.ex1.dao", cette interface contient les méthodes :
    - boolean create (T o) : Méthode permettant d'ajouter un objet o de type T.
    - boolean delete (T o) : Méthode permettant de supprimer un objet o de type T.
    - boolean update (T o) : Méthode permettant de modifier un objet o de type T.
    - T findById (int id) : Méthode permettant de renvoyer un objet dont id est passé en paramètre.
    - List <T> findAll ( ) : Méthode permettant de renvoyer la liste des objets de type T.
- Créer la classe ClientDaoImpl qui implémente l'interface ClientDao héritant de l'interface IDao dans le package "tp3.ex1.dao" et redéfinir toutes les méthodes de l'interfaces.

4- Développement de la couche service

- Créer la classe ClientServiceImpl qui implémente l'interface ClientService dans le package "tp3.ex1.service" et redéfinir la méthode List <Client> getAllClients (), qui renvoi la liste des clients triée par ordre alphabétique de leurs noms.

5- Développement de la couche présentation

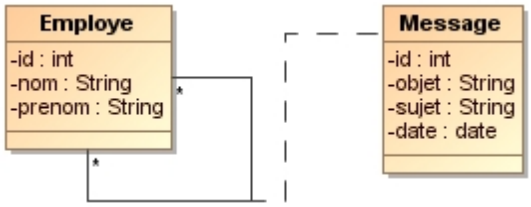
Dans une classe de test :

- Créer 5 clients
- Afficher le client dont id = 3
- Supprimer le client dont id = 3
- Modifier le client dont id = 2
- Afficher la liste des clients triée

Exercice 2 : Système de messagerie

Une société souhaite mettre en place un système de messagerie entre ses employés.

Les travaux de l'équipe chargée de l'analyse et de la conception ont abouti au diagramme de classe suivant :



1- Créer la base de données relationnelle correspondante au diagramme de classe ci-dessus

2- Créer dans la racine de votre projet un fichier properties nommé jdbc.properties qui contient les informations de connexion à la base de données messagerie.

```
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/messagerie
jdbc.username=root
jdbc.password=
```

3- Créer une classe Connexion permettant d'ouvrir une connexion à la base de données dans le package « tp3.ex2.dao »

4- Créer les classes Employe et Message dans le package « tp3.ex2.entities »

5- Créer l'interface IDao suivante dans le package «tp3.ex2.dao » :

6- Créer les classes EmployeDaoImpl et MessageDaoImpl qui implémentent les interfaces EmployeDao et MessageDao héritant de l'interface IDao :

   a. l'interface MessageDao doit ajouter deux méthodes :

      i. getMessagesBySender(Employe sender) : List<Message>
      ii. getMessagesByReceiver(Employe receiver) : List<Message>

7- Créer les classe EmployeServiceImpl et MessageServiceImpl qui implémentent l'interface EmployeService et MessageService et redéfinir la méthode getAllEmploye () et getAllMessages (), qui renvoient les listes demandées triées (par noms pour les employés et par date pour les messages)

8- Créer une classe Messagerie pour tester les CRUD dans le package « tp3.ex2.test », dans cette classe créer 3 employés :

   - Le 1er employé envoie un message aux deux autres ;
   - Le 2ème employé envoie un message aux deux autres ;
   - Afficher les messages reçus par le 3ème employé.