Cours : Accès aux Données en Java (ORM, JDBC, JPA)

Partie 1 : Accès aux données avec JDBC (ORM manuel)

Principe

JDBC est l'API Java de base pour accéder à une base de données relationnelle. Le développeur écrit manuellement les requêtes SQL et mappe les résultats dans des objets Java.

Étapes typiques

1. Charger le pilote JDBC
2. Ouvrir une connexion
3. Créer un PreparedStatement
4. Exécuter la requête
5. Parcourir les résultats avec ResultSet
6. Fermer les ressources

Exemple de code

```
Class.forName("com.mysql.cj.jdbc.Driver");
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/db", "root", "1234");

PreparedStatement ps = conn.prepareStatement("INSERT INTO users(name,email) VALUES (?, ?)");
ps.setString(1, "Anas");
ps.setString(2, "anas@gmail.com");
ps.executeUpdate();

ps.close();
conn.close();
```

Avantages / Inconvénients

Avantages :

- Contrôle complet sur les requêtes SQL
- Pas de dépendance externe

Inconvénients :

- Beaucoup de code répétitif
- Risque d'erreurs manuelles
- Pas de mapping objet-relationnel automatique

Cours : Accès aux Données en Java (ORM, JDBC, JPA)

Partie 2 : Hibernate avec hibernate.cfg.xml (sans JPA)

Principe

Hibernate peut être utilisé seul sans passer par JPA, via le fichier de configuration hibernate.cfg.xml et les classes SessionFactory, Session et Transaction.

Exemple de code

```
Configuration cfg = new Configuration();
cfg.configure("hibernate.cfg.xml");
SessionFactory factory = cfg.buildSessionFactory();

Session session = factory.openSession();
Transaction tx = session.beginTransaction();

User u = new User();
u.setName("Anas");
u.setEmail("anas@gmail.com");
session.save(u);

tx.commit();
session.close();
factory.close();
```

Avantages / Inconvénients

Avantages :

- Moins de dépendance à JPA

- Configuration plus libre

Inconvénients :

- Spécifique à Hibernate

- Moins portable

- Moins intégré aux frameworks modernes comme Spring

Partie 3 : JPA avec fichier persistence.xml (standard)

Principe

JPA est une API standard de Java EE pour la persistance des objets. Elle utilise un fichier de configuration persistence.xml pour définir les entités et la connexion à la base.

Cours : Accès aux Données en Java (ORM, JDBC, JPA)

Exemple de code

```
EntityManagerFactory emf = Persistence.createEntityManagerFactory("ma-unite-jpa");
EntityManager em = emf.createEntityManager();
em.getTransaction().begin();

User user = new User();
user.setName("Anas");
user.setEmail("anas@gmail.com");
em.persist(user);

em.getTransaction().commit();
em.close();
emf.close();
```

Avantages / Inconvénients

Avantages :

- Standard Java EE

- Compatible avec Spring Boot, Jakarta EE

- Portabilité entre fournisseurs (Hibernate, EclipseLink...)

Inconvénients :

- Peut sembler plus complexe au début

- Requiert une configuration initiale précise