JDBC CRUD utilisant Java et Mysql

1. Objectifs

- Apprendre à configurer votre environnement de développement et à l'utiliser pour se connecter à la base de données MySQL afin d'effectuer des requêtes
- Comprendre la notion de projet web dynamique Eclipse
- Créer un projet web dynamique

2. Présentation

1. **Le terme JDBC**
   - **JDBC représente la connectivité de base de données Java. JDBC est une API Java pour interfacer et exécuter la question avec la base de données. C'est un morceau de JavaSE (Java Standard Edition).**
   - **L'API JDBC utilise des pilotes JDBC pour se connecter à la base de données.**

2. **Le terme CRUD**
   - **Le terme CRUD est étroitement lié avec la gestion des données numériques. Plus précisément, CRUD est un acronyme des noms des quatre opérations de base de la gestion de la persistance des données et applications :**
     1. **Create (créer)**
     2. **Read ou Retrieve (lire)**
     3. **Update (mettre à jour)**
     4. **Delete ou Destroy (supprimer)**
   - **Le terme CRUD résume les fonctions qu'un utilisateur a besoin d'utiliser pour créer et gérer des données. Divers processus de gestion des données sont basés sur CRUD, cependant les opérations sont spécifiquement adaptées aux besoins des systèmes et des utilisateurs, que ce soit dans la gestion des bases de données ou pour l'utilisation des applications.**
   - **Il existe de nombreux Framework avec CRUD, qui sont basés sur une grande variété de langages et de plateformes. Quelques exemples se trouvent dans le tableau suivant:**

Langages ou plateformes	Framework
Java	**JDBC (The Java Database Connectivity), Hibernate, JBoss Seam, Isis**
PHP	**Yii, CakePHP, Zikula, Symfony, TYPO3 Flow, Laravel**

apcpedagogie.com

Perl	**Catalyst, Gantry**
Python	**Django, SQLAlchemy, web2py**
Groovy	**Grails**
.NET	**NHibernate, ADO.NET/Entity Framework**
Ruby	**Ruby on Rails**
JavaScript	**Backbone.js, AngularJS**

3. Création d'une application JDBC

- Les six étapes suivantes sont impliquées dans la création d'une application JDBC
  1. **Importer les packages**: nécessite que vous incluiez les packages contenant les classes JDBC nécessaires à la programmation de la base de données. Le plus souvent, il suffit d' utiliser `import java.sql. * .`
  2. **Enregistrer le pilote JDBC**: nécessite que vous initialisiez un pilote pour pouvoir ouvrir un canal de communication avec la base de données.
  3. **Ouvrir une connexion**: nécessite l'utilisation de la méthode `DriverManager.getConnection ()` pour créer un objet Connection, qui représente une connexion physique avec la base de données.
  4. **Exécuter une requête**: nécessite l'utilisation d'un objet de type Statement pour créer et soumettre une instruction SQL à la base de données.
  5. **Extraire les données de l'ensemble de résultats**: nécessite que vous utilisiez la méthode `ResultSet.getXXX ()` appropriée pour récupérer les données de l'ensemble de résultats.
  6. **Nettoyer l'environnement**: nécessite de fermer explicitement toutes les ressources de la base de données plutôt que de s'appuyer sur le garbage collection de la JVM.

4. Étapes générales de réalisation

1. **ÉTAPE 1. Importer les packages requis**

```
import java.sql.*;
public class Exemple {
```

2. **Introduir le nom du pilote JDBC et l'URL de la base de données**

```
static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
static final String DB_URL = "jdbc:mysql://localhost/EMP";
```

3. **Fournir les informations d'identification de la base de données**

```
static final String USER = "username";
static final String PASS = "password";
```

4. **Fournir une méthode principale**

```
   public static void main(String[] args) {
```

5. **Déclarer les variables nécessaires**

```
   Connection conn = null;
Statement stmt = null;
try{
```

6. **ÉTAPE 2: Enregistrer le pilote JDBC**

```
Class.forName("com.mysql.jdbc.Driver");
```

7. **ÉTAPE 3: ouvrir une connexion**

```
 System.out.println("Connecting to database...");
      conn = DriverManager.getConnection(DB_URL,USER,PASS);
```

8. **ÉTAPE 4: Exécuter une requête**

```
System.out.println("Creating statement...");
      stmt = conn.createStatement();
      String sql;
      sql = "SELECT champ01, champ02, .... FROM nom_table";
      ResultSet rs = stmt.executeQuery(sql);
```

apcpedagogie.com

9. **ÉTAPE 5: Extraire les données de l'ensemble de résultats**

```
while(rs.next()){
         //Récupérer par nom de colonne
         int id  = rs.getInt("champ_id");
         int age = rs.getInt("champ_01");
         String first = rs.getString("first");
         String last = rs.getString("last");

         //Afficher les valeurs
         System.out.print("ID: " + champ_id);
         System.out.print(", champ 01: " + champ_01);
         System.out.print(", First: " + first);
         System.out.println(", Last: " + last);
      }
```

10. **ÉTAPE 6: Environnement de nettoyage**

```
rs.close();
      stmt.close();
      conn.close();
   }catch(SQLException se){
      //Gérer les erreurs pour JDBC
      se.printStackTrace();
   }catch(Exception e){
      //Gérer les erreurs pour Class.forName
      e.printStackTrace();
   }finally{
      //enfin fermer les ressources
      try{
         if(stmt!=null)
            stmt.close();
      }catch(SQLException se2){
      }// nothing we can do
      try{
         if(conn!=null)
            conn.close();
      }catch(SQLException se){
         se.printStackTrace();
      }//end finally try
   }//end try
   System.out.println("Goodbye!");
```

apcpedagogie.com

```
}//fin méthode main
}//end Fin_Exemple
```

5. Obtenir une connexion à une base de données Mysql

- Cliquer sur l'image pour l'agrandir



```
package Mysql;
import java.sql.*;
public class Mysql {

    public static void main( String args[] ) {
        Connection conn = null;
        Statement stm = null;
        ResultSet rst=null;
        String usr = "aaaaaa";
        String Bd = "jdbc:mysql://localhost:3306/depinfo2015";
        String mp = "aaaaa";
        String sql2 = "select nomgroupe from groupes where codegroupe ='inf1'";
        try {
            Class.forName("com.mysql.jdbc.Driver");

            System.out.println("Pilote Chargé");
             }
        catch (ClassNotFoundException e) {
            System.err.println(e.getMessage());
            System.exit(0);
             }
        try {
            conn = DriverManager.getConnection(Bd, usr, mp);
            System.out.println("Pilote Chargé, Connexion ouverte");

            try(
                stm = conn.createStatement();
                rst = stm.executeQuery(sql2);
                if (rst.next())
                {
                    System.out.println("le nom du groupe est" + "   "+ rst.getString(1) );
                }
               }

            catch (SQLException sql)
            {
                System.out.println(sql.getMessage());
                System.exit(0);
            }

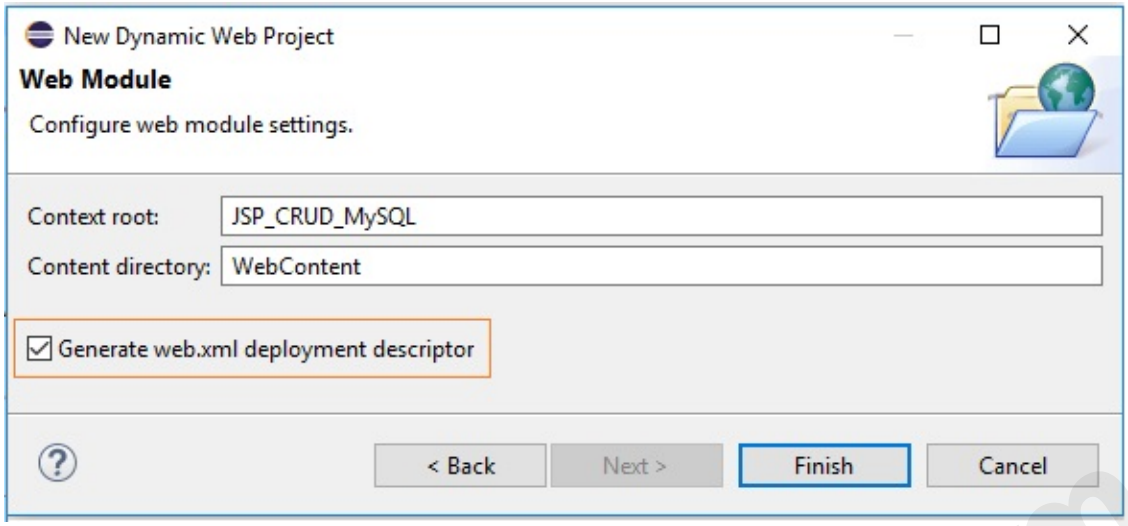
            finally {
                stm.close();
                rst.close();
                    }
        }
```

6. Activité

1. **Partie 1 : Création du projet web dynamique**

   - A partir d'Eclipse, faire menu `File` → `New` → `Dynamic Web Project`
   - Choisir le nom "JSPCrud" dans écran suivant
     - noter que le projet est associé à un serveur d'application, ici Tomcat 9. Cela est nécessaire. C'est une bonne pratique d'intégrer un serveur d'application Java dans Eclipse avant de créer un projet web dynamique.
   - Puis cliquez sur le bouton `Next` pour les deux écrans qui suivent
   - Cocher la case "Generate web.xml deployment descriptor"

apcpedagogie.com



Sommaire du cours JSP

Source : https://apcpedagogie.com/jdbc-crud-utilisant-java-et-mysql · téléchargé le 07/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com