Chapitre 4 : Accès aux bases de données avec JDBC

Introduction

- JDBC (Java DataBase Connector) est une API chargée de communiquer avec les bases de données en Java.
- Les classes et interfaces de l'API JDBC figurent dans le package java.sql : `import java.sql.*;`
- JDBC peut être utilisé pour accéder à n'importe quelle base de données à partir de:
  - Simple application Java
  - Une servlet
  - Page JSP, …

Travail avec une base de données

- JDBC permet de travailler avec les base de données de la même façon quelque soit leur fournisseur (Oracle, SQL Server, MySQL, PostgreSQL,…).
- Il suffit de télécharger la bibliothèque qui assure la communication entre Java et cette base de donnée.
- Cette bibliothèque s'appelle Driver ou Pilote ou Connecteur.
- Elle figure sur le site du fournisseur du SGBDR utilisé.

Etapes d'interaction avec une BDD

1. Chargement du pilote
2. Etablissement de la connexion
3. Création des objets encapsulant les requêtes
4. Exécution des requêtes
5. Parcours des résultats dans le cas d'une requête de sélection
6. Fermeture des objets résultats, requêtes et connexion

Chargement du pilote

- Pour se connecter à une base de données il faut charger son pilote.
- La documentation de la Bdd utilisée fournit le nom de la classe à utiliser.
- Exemple de pilote MYSQL :
  - Le connecteur MySQL pour Java se nomme comme cet exemple : " **mysql-connector-java-5.1.23-bin.jar**"
- Exemple de pilote Derby (intégré à Netbeans pour les tests)
  - Il se nomme Java DB Driver (est composé de 3 jars)
  - A ajouter par le menu (Add Library)

Chargement du pilote

- Le chargement se fait comme suit : Class.forName("nom_classe_acces_bdd");
- Exemple :
  - Dans le cas de la Bdd MySQL, ce chargement est comme suit : Class.forName(com.mysql.jdbc.**Driver**)
  - Dans le cas de la Bdd Derby (notée Java DB), ce chargement est comme suit : Class.forName(org.apache.derby.jdbc.**ClientDriver**)
- Une fois chargée, la classe JDBC qui se nomme **DriverManager** prend en charge le driver pour communiquer avec la base de donnée.

Classes de l'API JDBC

- Les classes et interfaces les plus usuelles sont les suivantes:
  - **DriverManager** (classe): charge et configure le driver de la base de données.
  - **Connection** (interface): réalise la connexion et l'authentification à la base de données.
  - **Statement** (interface): contient la requête SQL et la transmet à la base de données.
  - **PreparedStatement** (interface): représente une requête paramétrée
  - **ResultSet** (interface): représente les résultat d'une requête de sélection.

Etablissement de la connexion

- Pour se connecter à une base de données, il faut disposer d'un objet **Connection** créé grâce au DriverManager en lui passant :
  - l'URL de la base à accéder , Le login, Le mot de passe
- Exemples:
  - String url="jdbc:mysql://localhost/mydb"; // exemple URL BDD MySQL
  - String url="jdbc:derby://localhost:1527/EtudiantsDB3"; // exemple URL BDD Derby
  - String login="root";
  - String password="motdepasse";
  - **Connection** con=**DriverManager.getConnection**(url, login, password);

Exécution de requêtes SQL

- L'interface **Statement** permet d'envoyer des requêtes SQL à la base de données.
- Un objet Statement est créé grâce à un objet Connection de la façon suivante : **Statement** st = **con**.createStatement();
- Il possède deux méthodes :
  - **executeUpdate()** : Insertion, suppression, mise à jour.
    - int n= st.executeUpdate("**INSERT INTO Etudiant VALUES (3452,'Taha','Ali')**");
  - **executeQuery()** : Selection.
    - **ResultSet** res= stm.**executeQuery**("**SELECT * FROM Etudiant**");

Requêtes avec paramètres

- L'interface **preparedStatement** permet d'envoyer des requêtes SQL à la base de données en prenant des paramètres.
- Ces paramètres sont représentés par des points d'interrogation(**?**) et doivent être spécifiés avant l'exécution.
- Exemple :
  - **PreparedStatement** p= con.**prepareStatement**("select* from Etudiant where cne=? And nom= ? ");

Requêtes avec paramètres

```
p.setInt(1, 3452345);
p.setString(2, "Alaoui");
ResultSet resultats = p.executeQuery();
```


Résultat d'une requête de sélection

- Une requête de sélection retourne un **ResultSet**
- ResultSet est un ensemble d'enregistrements constitués de colonnes qui contiennent les données.
- Les principales méthodes :
  - **next()** : se déplace sur le prochain enregistrement : retourne false si la fin est atteinte. Le curseur pointe initialement juste avant le premier enregistrement.
  - **getInt(**int/String**)** : retourne le contenu de la colonne dont le numéro (resp. le nom) est passé en paramètre sous forme d'entier.

Résultat d'une requête de sélection

- **getFloat(int/String)** : retourne le contenu de la colonne sous forme de nombre flottant.
- **getDate(int/String)** : retourne le contenu de la colonne sous forme de date.
- **Close()** : ferme le ResultSet

Résultat d'une requête de sélection

- Exemple :

```
ResultSet res= st.executeQuery("SELECT * FROM Etudiant");
while (res.next()) {
    System.out.println("CNE= "+res.getString(1)+" Nom= " +
        res.getString(2)+" Prénom= "+res.getString(3));
}
res.close();
```

Enoncé TP

- Création de la base de données :
  - Utiliser MYSQL ou DERBY pour créer un base de données ayant une table ETUDIANT composée des champs :
    - CNE
    - NOM
    - PRENOM

Enoncé TP

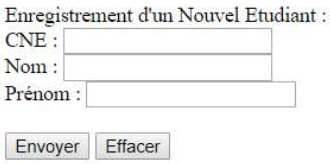
- Création de l'application Web :
  - L'application web démarre avec une page d'accueil (HTML) avec deux liens hypertextes:
    - Un lien d'insertion de nouveaux étudiants
    - Un lien d'affichage des étudiants déjà insérés

Gestion des étudiants :

- Insérer nouvel Etudiant
- Afficher liste Etudiants

Enoncé TP

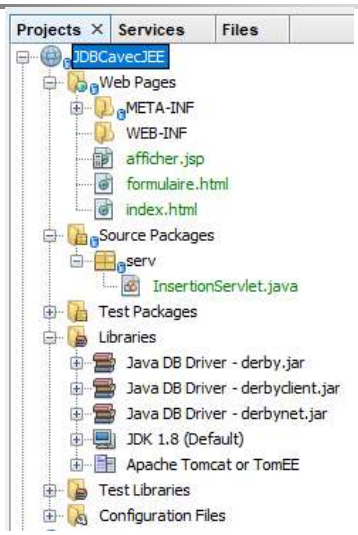
- Le premier lien pointe vers un formulaire HTML qui permet de saisir le CNE, le nom, et le prénom d'un étudiant.
- Ce formulaire est traité par une **Servlet** qui se charge de faire l'insertion dans la base de données.
- Après l'insertion dans la base de données la servlet affiche un message d'insertion réussie et un lien pour revenir à la page d'accueil.



Enoncé TP

- Le deuxième lien pointe vers une page **JSP** qui se charge d'accéder à la base de données et afficher les données dans un tableau HTML.
- La page JSP à son tour affiche un lien pour revenir à la page d'accueil.

Solution : Structure du projet sous Netbeans



Solution : index.html

```
<!DOCTYPE html>
<html>
   <head>
      <title>Gestion des étudiants</title>
   </head>
   <body>
      <h2>Gestion des étudiants : </h2>
      <ul type="disc">
         <li><a href="formulaire.html">Insérer nouvel Etudiant</a></li>
         <li><a href="afficher.jsp"> Afficher liste Etudiants</a></li>
      </ul>
   </body>
</html>
```

Solution : formulaire.html

```
<html>
   <head> <title>formulaire</title> </head>
   <body>
      <form method="post" action="traitementFormulaire">
         Enregistrement d'un Nouvel Etudiant : <br>
         CNE : <input type="text" name="cne"><br>
         Nom : <input type="text" name="nom"><br>
         Prénom : <input type="text" name="prenom"><br> <br>
         <input type="submit" value="Envoyer">
         <input type="reset" value="Effacer">
      </form>
   </body>
</html>
```


Solution : InsertionServlet.java

```
package serv;
import java.io.*;
import javax.servlet.*;
import javax.servlet.annotation.*;
import javax.servlet.http.*;
import java.sql.*;
@WebServlet(name = "InsertionServlet", urlPatterns = {"/traitementFormulaire"})
public class InsertionServlet extends HttpServlet {

   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
```

Solution : InsertionServlet.java

```
try { String fcne = request.getParameter("cne");
      String fnom = request.getParameter("nom");
      String fprenom = request.getParameter("prenom");
      String url = "jdbc:derby://localhost:1527/EtudiantsDB3";
      String driver = "org.apache.derby.jdbc.ClientDriver";
      Class.forName(driver);
      Connection con= DriverManager.getConnection(url, "root", "root");
      PreparedStatement stmt = con.prepareStatement("insert into ETUDIANT(CNE ,NOM, PRENOM)
values (?,?,?)");
          stmt.setString(1, fcne);
          stmt.setString(2, fnom);
          stmt.setString(3, fprenom);
          stmt.executeUpdate();
          stmt.close();
          con.close();
```


Solution : InsertionServlet.java

```
 out.println("insertion nouvel étudiant réussie<br>");
 out.println("<a href=\"index.html\"> retour à la page index </a>");
 }
 catch (ClassNotFoundException | IllegalAccessException | InstantiationException | SQLException e) {
          out.println("Erreur : " + e.getMessage() + " source : " + e.getStackTrace());
  }
}
@Override
   protected void doGet(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {  processRequest(request, response);  }
 protected void doPost(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException { processRequest(request, response); }
}
```

Solution : afficher.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8" import="java.sql.*"%>
<!DOCTYPE html>
<%
   String url = "jdbc:derby://localhost:1527/EtudiantsDB3";
   String driver = "org.apache.derby.jdbc.ClientDriver";
   Class.forName(driver).newInstance();
   Connection con;
   con = DriverManager.getConnection(url, "root", "root");
   PreparedStatement stmt = con.prepareStatement("select * from ETUDIANT");
   ResultSet rs = stmt.executeQuery();
%>
```


Solution : afficher.jsp

```
<html>
   <head>
     <title>Affichage JSP</title>
   </head>
   <body>
      <h2> Liste des Etudiants Inscrits : </h2>
      <table border="1">
         <thead>
            <tr>
               <th>CNE</th>
               <th>NOM</th>
               <th>PRENOM</th>
            </tr>
         </thead>
```

Solution : afficher.jsp

```
<tbody>
            <% while (rs.next()) {%>
               <tr> <td><%=rs.getString(1)%></td>
                    <td><%=rs.getString(2)%></td>
                    <td><%=rs.getString(3)%></td>
                </tr> <%}%>
            </tbody>
        </table>
        <% rs.close();
           stmt.close();
           con.close();
        %>
        <a href="index.html"> retour à la page d'accueil </a>
    </body>
</html>
```