

TP5-DAO : JEE

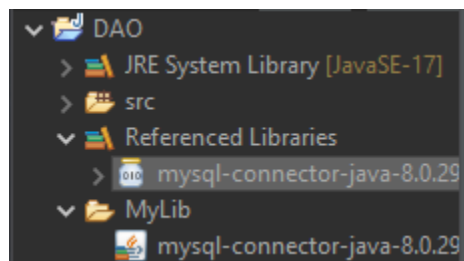
☰ Tags	
▼ status	sent

Par : IBTISSAM MASSA - GINFO2

DAO

L'objectif de ce TP est de se familiariser avec les mécanismes d'accès aux bases de données en utilisant le DAO,

1. Lancer votre IDE
2. Créer un nouveau projet Java (simple projet).
3. Créer un nouveau répertoire MyLib dans votre projet Eclipse
4. Copier le fichier mysql-connector-java-XXXX-bin.jar dans le répertoire MyLib
5. Ajouter le fichier.jar au Build Path de votre projet



6. Créer un package packDAO
7. Créer le bean BookBean avec les attributs suivants :
 - private long book_id;
 - private String title;

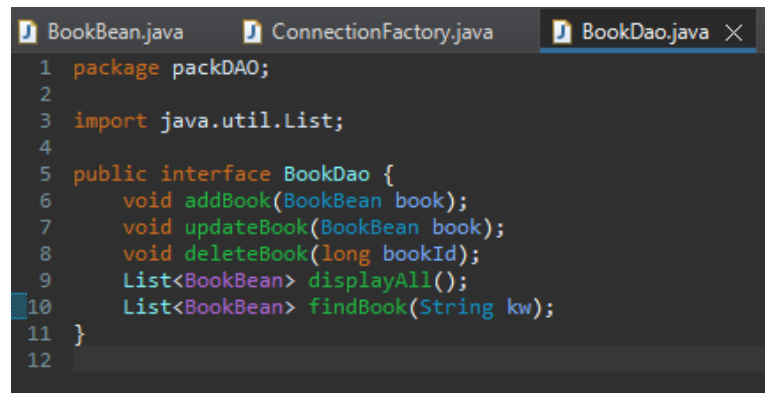
- private String author;
- private float price;

```
1 package packDAO;
2
3 public class BookBean {
4     private long book_id;
5     private String title;
6     private String author;
7     private float price;
8
9     public BookBean(long book_id, String title, String author, float price) {
10         this.book_id = book_id;
11         this.title = title;
12         this.author = author;
13         this.price = price;
14     }
15
16     public long getBook_id() {
17         return book_id;
18     }
19     public void setBook_id(long book_id) {
20         this.book_id = book_id;
21     }
22     public String getTitle() {
23         return title;
24     }
25     public void setTitle(String title) {
26         this.title = title;
27     }
28     public String getAuthor() {
29         return author;
30     }
31     public void setAuthor(String author) {
32         this.author = author;
33     }
34     public float getPrice() {
35         return price;
36     }
37 }
```

8. Créer la classe ConnectionFactory pour établir la connexion avec la BD biblio

```
BookBean.java  ConnectionFactory.java X
1 package packDAO;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class ConnectionFactory {
8     private static final String URL = "jdbc:mysql://localhost:3306/biblio";
9     private static final String USER = "MyUser";
10    private static final String PASSWORD = "1234";
11
12    public static Connection getConnection() {
13        try {
14            return DriverManager.getConnection(URL, USER, PASSWORD);
15        } catch (SQLException e) {
16            throw new RuntimeException("Erreur de connexion à la base de données", e);
17        }
18    }
19 }
20 }
```

9. Créer l'interface BookDao et ajouter les méthodes suivantes :
- a. addBook(BookBean book) : ajouter u nouveau book
 - b. updateBook(BookBean book) : modifier un book existant
 - c. deleteBook(Long book_id) : supprimer un book
 - d. displayAll() : afficher tous les books
 - e. findBook(String kw) : trouver un book avec un mot clé (kw)



```
1 package packDAO;
2
3 import java.util.List;
4
5 public interface BookDao {
6     void addBook(BookBean book);
7     void updateBook(BookBean book);
8     void deleteBook(long bookId);
9     List<BookBean> displayAll();
10    List<BookBean> findBook(String kw);
11 }
12
```

10. Créer la classe BookDaoImpl et définir les différentes méthodes

```

BookBean.java  ConnectionFactory.java  BookDao.java  BookDaoImpl.java
1 package packDAO;
2
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.util.ArrayList;
8 import java.util.List;
9
10 public class BookDaoImpl implements BookDao {
11     private Connection connection = ConnectionFactory.getConnection();
12
13     @Override
14     public void addBook(BookBean book) {
15         try {
16             String sql = "INSERT INTO book (title, author, price) VALUES (?, ?, ?)";
17             PreparedStatement statement = connection.prepareStatement(sql);
18             statement.setString(1, book.getTitle());
19             statement.setString(2, book.getAuthor());
20             statement.setFloat(3, book.getPrice());
21             statement.executeUpdate();
22         } catch (SQLException e) {
23             e.printStackTrace();
24         }
25     }
26
27     @Override
28     public void updateBook(BookBean book) {
29         try {
30             String sql = "UPDATE book SET title = ?, author = ?, price = ? WHERE book_id = ?";
31             PreparedStatement statement = connection.prepareStatement(sql);
32             statement.setString(1, book.getTitle());
33             statement.setString(2, book.getAuthor());
34             statement.setFloat(3, book.getPrice());
35             statement.setLong(4, book.getBook_id());
36             statement.executeUpdate();
37         } catch (SQLException e) {
38             e.printStackTrace();
39         }
40     }
41
42     @Override
43     public void deleteBook(long bookId) {
44         try {
45             String sql = "DELETE FROM book WHERE book_id = ?";
46             PreparedStatement statement = connection.prepareStatement(sql);
47             statement.setLong(1, bookId);
48             statement.executeUpdate();
49         } catch (SQLException e) {
50             e.printStackTrace();
51         }
52     }
53

```

```

42  @Override
43  public void deleteBook(long bookId) {
44      try {
45          String sql = "DELETE FROM book WHERE book_id = ?";
46          PreparedStatement statement = connection.prepareStatement(sql);
47          statement.setLong(1, bookId);
48          statement.executeUpdate();
49      } catch (SQLException e) {
50          e.printStackTrace();
51      }
52  }
53
54  @Override
55  public List<BookBean> displayAll() {
56      List<BookBean> books = new ArrayList<>();
57      try {
58          String sql = "SELECT * FROM book";
59          PreparedStatement statement = connection.prepareStatement(sql);
60          ResultSet resultSet = statement.executeQuery();
61
62          while (resultSet.next()) {
63              long bookId = resultSet.getLong("book_id");
64              String title = resultSet.getString("title");
65              String author = resultSet.getString("author");
66              float price = resultSet.getFloat("price");
67
68              books.add(new BookBean(bookId, title, author, price));
69          }
70      } catch (SQLException e) {
71          e.printStackTrace();
72      }
73      return books;
74  }
75
76  @Override
77  public List<BookBean> findBook(String kw) {
78      List<BookBean> books = new ArrayList<>();
79      try {
80          String sql = "SELECT * FROM book WHERE title LIKE ? OR author LIKE ?";
81          PreparedStatement statement = connection.prepareStatement(sql);
82          statement.setString(1, "%" + kw + "%");
83          statement.setString(2, "%" + kw + "%");
84          ResultSet resultSet = statement.executeQuery();
85
86          while (resultSet.next()) {
87              long bookId = resultSet.getLong("book_id");
88              String title = resultSet.getString("title");
89              String author = resultSet.getString("author");
90              float price = resultSet.getFloat("price");
91
92              books.add(new BookBean(bookId, title, author, price));
93          }
94      } catch (SQLException e) {
95          e.printStackTrace();
96      }
97      return books;
98  }
99  }

```

11. Créer une nouvelle classe avec une méthode main et exécuter les opérations suivantes
 - a. Afficher tous les enregistrements de la table book
 - b. Insérer un nouveau book
 - c. Afficher le nouveau book ajouté
 - d. Modifier le prix de ce book
 - e. Supprimer un book

```

BookBean.java  ConnectionFactory.java  BookDao.java  BookDaoImpl.java  App.java X
1 package packDAO;
2
3 import java.util.List;
4
5 public class App {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9         BookDao bookDao = new BookDaoImpl();
10
11         //Afficher tous les enregistrements de la table
12         List<BookBean> allBooks = bookDao.displayAll();
13         for (BookBean book : allBooks) {
14             System.out.println(book.getBook_id() + " " + book.getTitle() + " " + book.getAuthor() + " " + book.ge
15         }
16
17         //Insérer un nouveau livre
18         BookBean newBook = new BookBean(0, "Nouveau Livre", "Nouvel Auteur", 40);
19         bookDao.addBook(newBook);
20
21         //Afficher le nouveau livre ajouté
22         List<BookBean> foundBooks = bookDao.findBook("Nouveau Livre");
23         if (!foundBooks.isEmpty()) {
24             System.out.println("Nouveau livre ajouté :");
25             for (BookBean book : foundBooks) {
26                 System.out.println(book.getBook_id() + " " + book.getTitle() + " " + book.getAuthor() + " " + boc
27             }
28         }
29
30         //Modifier le prix du livre
31         newBook.setPrice(40);
32         bookDao.updateBook(newBook);
33
34         //Supprimer un livre
35         bookDao.deleteBook(newBook.getBook_id());
36     }
37 }
38 }

```

```

1 Livre 1 Auteur 1 19.99
2 Livre 2 Auteur 2 15.99
3 Livre 3 Auteur 3 24.99
4 Livre 4 Auteur 4 12.99
Nouveau livre ajouté :
6 Nouveau Livre Nouvel Auteur 40.0

```