JDBC-Quicktutorial

1 Klassen für das Abspeichern der Daten

Zum Speichern der Daten wird für jede Tabelle in der SQL-Struktur eine eigenen Java-Klasse definiert. Hier wird beispielhaft die Klasse `Book` betrachtet.

```
public int id;
public String title;
public String author;
public String isbn;
public int category_id;
```

Codesnippet 1: Felder der Book-Klasse

Für das sinnvolle Verwenden der `Book`-Klasse wird noch ein Konstruktor und eine `toString()`-Methode benötigt. Diese können mit Hilfe der Menüs *Source* → *Generate Constructor using Fields...* und *Source* → *Generate toString()...* in der Menüleiste generiert werden.

```
public Book(int id, String title, String author, String isbn, int category_id) {
    this.id = id;
    this.title = title;
    this.author = author;
    this.isbn = isbn;
    this.category_id = category_id;
}
```

Codesnippet 2: Konstruktor der Book-Klasse

```
@Override
public String toString() {
    return "Book [id=" + id + ", title=" + title + ", author=" + author + ", isbn=" + isbn + ",
 ↪  category_id=" + category_id + "]";
}
```

Codesnippet 3: toString() der Book-Klasse

2 DBConn für die Datenbankverbindung

Die Programmlogik, welche für die Kommunikation mit der Datenbank benötigt wird, wird in der Java-Klasse `DBConn` programmiert.

2.1 Verbindungsaufbau

Im Konstruktor wird eine Verbindung aufgebaut, in den Methoden werden Datenbankabfragen (SELECTs und INSERTs) ausgeführt.

```
public DBConn() throws ClassNotFoundException, SQLException {
    Driver driver = new com.mysql.jdbc.Driver();
    DriverManager.registerDriver(driver);

    conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/bibliothek", "root", "");
    stm = conn.createStatement();
}
```

Codesnippet 4: Verbindungsaufbau zum MySQL-Server im Konstruktor

2.2 Auslesen der Datenbank

Wenn eine Tabelle aus der Datenbank abgefragt wird, wird dafür eine eigene Methode geschrieben. In der Methode passieren folgende Dinge:

- Anlegen einer Liste in der das Ergebnis gespeichert wird.
- Speichern des SQL-Befehls in einen String
- Ausführen des SQL-Befehls mithilfe von `stm.executeQuery(sql)`
- Durchgehen des `ResultSet`s mithilfe von `while (rs.next()) {...}`
  - Erstellen eines neuen Objekts von der zugehörigen Datenklasse (siehe Abschnitt 1)
  - Hinzufügen des eben erstellen Objekts zur Ergebnisliste
- Returnen der Ergebnisliste

```
public List<Book> selectBooks(int id_categories) throws SQLException{
    List<Book> result = new ArrayList<>();

    String sql = "SELECT * FROM books WHERE category_id = " + id_categories;
    ResultSet rs = stm.executeQuery(sql);

    while (rs.next()) {
        Book b = new Book(rs.getInt("id"), rs.getString("title"), rs.getString("author"),
↪   rs.getString("ISBN"), rs.getInt("category_id"));
        result.add(b);
    }

    return result;
}
```

Codesnippet 5: Abfragen der Tabelle `books`

2.3 Schreiben in die Datenbank

Zum Schreiben von Dokumenten in die Datenbank werden auch eigene Methoden definert. Hier z. B. wird ein Buch in die Datenbank eingefügt.

```java
public void insertBook(Book b) throws SQLException{
    String sql = "INSERT INTO books (title, author, ISBN, category_id) values (?,?,?,?)";
    PreparedStatement pstm = conn.prepareStatement(sql);
    pstm.setString(1, b.title);
    pstm.setString(2, b.author);
    pstm.setString(3, b.isbn);
    pstm.setInt(4, b.category_id);
    pstm.execute();
}
```

Codesnippet 6: Schreiben in die Tabelle `books`

3 JSPs

Will man nun z. B. eine Liste aus Büchern (`List<Book>`) in tabellarischer Form darstellen kann man folgenden Code verwenden. Das Objekt `db` wurde hier mit `DBConn db = new DBConn();` zu Beginn der JSP erstellt.

```jsp
<%
List<Book> books = db.selectBooks(c.id);
%>
<table border="1px">
    <tr><td>Titel</td><td>Autor</td><td>ISBN</td><td></td></tr>
    <% for(Book b : books) { %>
        <tr>
            <td><%= b.title %></td>
            <td><%= b.author %></td>
            <td><%= b.ISBN %></td>
            <td><a href="DeleteServlet?id=<%= b.id %>" > &times;</a></td>
        </tr>

    <% } %>
</table>
```

Codesnippet 7: Erstellen einer HTML-Tabelle