**Fach: Projekt UNO (Projekt I + Projekt II)**

Dokumentation:Datenbank

**Die SQLite Datenbank (DB)**

Externe Library: [**SQLite JDBC**](https://mvnrepository.com/artifact/org.xerial/sqlite-jdbc)**»**[**3.50.2.0**](https://mvnrepository.com/artifact/org.xerial/sqlite-jdbc/3.50.2.0)

Source Link: <https://mvnrepository.com/artifact/org.xerial/sqlite-jdbc>

Version: Aktuell 3.50.2.0 vom 30.06.2025 (Stand 08.07.2025)

Datei: sqlite-jdbc-3.50.2.0.jar (gespeichert im Projekt im Ordner „lib“

|  |  |  |
| --- | --- | --- |
| **Properties** | **Status** | **Bemerkungen** |
| Server | Serverlos/Dateienbasiert | Datei: uno.sqllite |
| JDBS Driver | sqlite-jdbc-3.50.2.0.jar | ./lib/ sqlite-jdbc-3.50.2.0.jar |
| Output | .\uno.sqllite | In IntellegiIdea(Community) wird die Datei nur in binärischem Format angezeit. Deswegen für Windows wurde noch extra DB-Browser für SQLite (Version 3.13.1). Damit wurde das Ziel „die Tabelle „Sessions“ erreicht |
|  |  |  |
|  |  |  |

Datenbank Beschreibung:

Datenbank ist mit SQLLite Library verwendet bzw. gespeichert. Datenbank funktioniert ohne Server. Dateien können direkt aus einer Datei heraus benutzen („uno.sqllite“). Also eine einzige lokale Datei enthält die gesamte Datenbank.

**Die Klasse SQLLiteClient.java**

|  |  |  |  |
| --- | --- | --- | --- |
| **Funktion** | **Klasse** | **Methode** | **Bemerkung** |
| *Verbindung zu DB*  *erstellen* | SQLLiteClient.java | public **SQLLiteClient**(String dbName) throws SQLException {  this.connection = DriverManager.*getConnection*("jdbc:sqlite:" + dbName); } | Es ist eine vorgegebene Klasse, sie wurde nur leicht bearbeitet/korrigiert entsprechend unserer Implementation |
| *Prüft, ob eine bestimmte Tabelle in der Datenbank existiert* | SQLLiteClient.java | public boolean tableExists(String tableName) | Ziel: beim Anfang eines Spiels die Tabelle „Sessions“ leeren (Anforderung) |
| *Führt ein SQL-Statement (z.B. CREATE, INSERT, UPDATE, DELETE) aus* | SQLLiteClient.java | public void executeStatement(String sqlStatement) | In unserem Fall: für Erstellen der Tabelle „Sessions“ |
| *Führt eine SQL-SELECT-Abfrage aus und gibt die Ergebnisse als Liste von HashMaps zurück* | SQLLiteClient.java | public ArrayList<HashMap<String, String>> executeQuery(String sqlQuery) | In unserem Fall: für Anzeigen des Inhalts der Tabelle „Sessions“ (am Ende des Spiels oder nach dem „Exit“) |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Die Klasse DatabaseHelper.java**

|  |  |  |  |
| --- | --- | --- | --- |
| **Funktion** | **Klasse** | **Methode** | **Bemerkung** |
| *Initialisiert die Tabelle, wenn sie noch nicht existiert* | DatabaseHelper.java | public void initDatabaseIfNeeded() | **TODO: Vergleichen mit Create Table** |
| *Um die Tabelle „Sessions“ zu leeren* | DatabaseHelper.java | public void resetSessions() | *//Um DB vor das neue Spiel leeren* |
| *Speichern „currentStatus“, falls wurde „Exit“ (case: 0) gedruckt* | DatabaseHelper.java | public void saveGameState(ArrayList<Player> players, int sessionId, int roundNumber) | *//Game State wird gesspeichert: beim Exit(0)-also, das Spiel noch nicht beendet //momentan mehr als Test für DB Connection und Speichern gedacht* |
| *Speichern alle Result, falls jemand hat gewonnen* | DatabaseHelper.java | public void saveRoundResult(String name, int sessionId, int roundNumber, int point, boolean isBot) | Name ändern? Gesamt ergebnis |
| *SELECT \* from* |  |  | *//TODO: beim Ende des Spiels die Daten aus der Datenbank anzeigen* |