Übungszettel – Datenpersistenz

Name: Michael Bogensberger

Datum: 02.02.2022

Inhalt

[2 Entwicklungsumgebung einrichten 3](#_Toc95296684)

[3 Projekt anlegen 3](#_Toc95296685)

[3.1 MySql Connector Dependency hinzufügen 3](#_Toc95296686)

[4 Verbindung zu Datenbank herstellen 3](#_Toc95296687)

[5 Einfaches CRUD Beispiel mit JDBC 4](#_Toc95296688)

[6 CRUD Beispiel erweitern 4](#_Toc95296689)

# Entwicklungsumgebung einrichten

Zunächst müssen wir die Entwicklungsumgebung einrichten. Dafür installieren wir uns zunächst XAMPP. Der Name "XAMPP" ist eine Abkürzung für Apache, MySQL, Perl und PHP. Das "X" am Anfang bezieht sich darauf, dass das Programm auf verschiedenen Betriebssystemen wie Windows, Linux oder Max OS X läuft. Gegebenen falls muss man in XAMPP Ports ändern. Falls man zum Beispiel die MySql Workbench installiert hat, kann es sein das der Port 3306 bereits belegt ist. Hierfür empfiehlt sich auf den Port 3307 zu wechseln. Danach kann man im Browser die URL: localhost aufrufen. Von dort an kann man in phpMyAdmin einsteigen.

# Projekt anlegen

Als nächsten legen wir in IntelliJ ein neues Maven Projekt an. Dafür wählen wir Maven aus und verwenden dieses Mal keinen Archetype.

## MySql Connector Dependency hinzufügen

Nun fügen wir die MySql Connector Dependency hinzu. Diese wird benötigt, um mit der Datenbank kommunizieren zu können. Dazu fügen wir folgendes in der pom.xml hinzu.

|  |
| --- |
| <dependencies>  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>8.0.27</version>  </dependency> </dependencies> |

# Verbindung zu Datenbank herstellen

In folgendem Codeabschnitt wird dargestellt wie man mithilfe der MySql Connector Dependency eine Verbindung zur Datenbank herstellen kann.

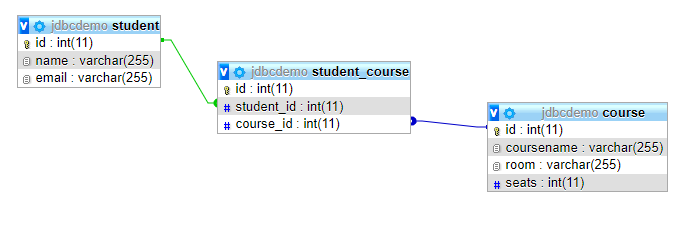
|  |
| --- |
| String connectionUrl = "jdbc:mysql://localhost:3306/jdbcdemo"**;** String username = "root"**;** String pw = ""**;** try(Connection conn = DriverManager.*getConnection*(connectionUrl**,** username**,** pw)) {  System.*out*.println("DB Verbindung ok")**;** } catch (SQLException e) {  System.*out*.println("DB Verbindung fehlgeschlagen " + e.getMessage())**;** } |

# Einfaches CRUD Beispiel mit JDBC

Nun war es die Aufgabe mithilfe der Videos ein einfaches CRUD Programm mithilfe von JDBC zu gestalten. Dabei geht es im wesentlichen um das auslesen von Studenten, dem einfügen von Studenten sowie dem löschen von Studenten. Der Code zum Beispiel ist hier zu finden: [jdbcCrudExample](https://github.com/MichaelBogensberger/DatenpersistenzFSE).

Hier ist der Aufbau der Student Tabelle zu sehen:

# CRUD Beispiel erweitern

Als nächstes sollten wir das vorherige Beispiel um eine weitere Tabelle ergänzen und dieses in Java implementieren. Dazu habe ich der Datenbank eine Kurs-Tabelle hinzugefügt. Jeder Student hat mehrere Kurse. Jeder Kurs hat zudem mehrere Studenten. Der Code ist im Gleichen GitHub Repository zu finden wie in der vorherigen Aufgabe.