



Kurzes Tutorial über

Java Database Connectivity

(JDBC)

von

mraab

Inhaltsverzeichnis

Inhaltsverzeichnis	II
Abbildungsverzeichnis.....	III
Tabellenverzeichnis.....	III
Abkürzungsverzeichnis	III
1 Allgemeines.....	1
1.1 e-Portfolio.....	1
1.2 MySQL.....	1
1.3 Java Database Connectivity (JDBC)	1
2 Vorgehensweise	2
2.1 Download MySQL 5.6.17	2
2.2 Installation MySQL 5.6.17	2
2.3 Datenbank mit MySQL erstellen	3
2.4 Datenbankverbindung über Eclipse	5
2.4.1 Download MySQL Connector/J	5
2.4.2 Datenbank-Treiber der Klassenbibliothek hinzufügen.....	6
2.5 Beispiel einer Standard-Abfrage (Quellcode)	7
3 Forum	9

Abbildungsverzeichnis

Abbildung 1: Login SQL-Server	3
Abbildung 2: create Database	3
Abbildung 3: Datenbank prüfen für Veränderungen markieren	4
Abbildung 4: create table	4
Abbildung 5: insert into table	4
Abbildung 6: Datenbankinhalt prüfen mit Select From	5
Abbildung 7: Projekt und Klasse anlegen	5
Abbildung 8: Build Path.....	6
Abbildung 9: Add External JARs.....	7
Abbildung 10: import SQL-Funktionen	7
Abbildung 11: try-Anweisung	8
Abbildung 12: catch-Anweisung	8
Abbildung 13: Ausgabe.....	9

Tabellenverzeichnis

Tabelle 1: Abkürzungsverzeichnis.....	III
---------------------------------------	-----

Abkürzungsverzeichnis

Begriff	Erklärung
MySQL	My Structered Query Language
JDBC	Java Database Connectivity

Tabelle 1: Abkürzungsverzeichnis

1 Allgemeines

1.1 e-Portfolio

Das e-Portfolio ist eine Methode um einen Lernprozess zu dokumentieren bzw. zu reflektieren. Dabei werden verschiedene digitale Medien genutzt. Ein häufiges Vorgehen ist die Nutzung eines eigenen Blogs. Hierbei werden durch eine Art Tagebuch alle Fortschritte online festgehalten.

1.2 MySQL

Eines der verbreitetsten Open-Source-Datenbankverwaltungssysteme, die momentan auf dem Markt sind. Hauptsächlich Anwendungsgebiet von MySQL ist die Datenspeicherung von Webservices mittels eines MySQL-Servers.

1.3 Java Database Connectivity (JDBC)

JDBC ist eine auf der Java-Plattform ausführbare Datenbankschnittstelle. Zu den Hauptaufgaben zählen somit, ausgehend von der IDE Java Eclipse, der Aufbau und die Verwaltung einer Datenbankverbindung. Ausgelegt ist JDBC vor allem auf relationale Datenbankmodelle und unterstützt dabei Datenbanken verschiedener Hersteller.

2 Vorgehensweise

2.1 Download MySQL 5.6.17

Für eine erfolgreiche Umsetzung wird die Software MySQL 5.6.17 verwendet. Diese wird als zip-Archiv für Microsoft Windows Betriebssysteme kostenfrei heruntergeladen. MySQL 5.6.17 ist je nach Architektur in der 32- oder 64-Bit-Version unter folgendem Download-Link: <http://dev.mysql.com/downloads/mysql/> erhältlich.

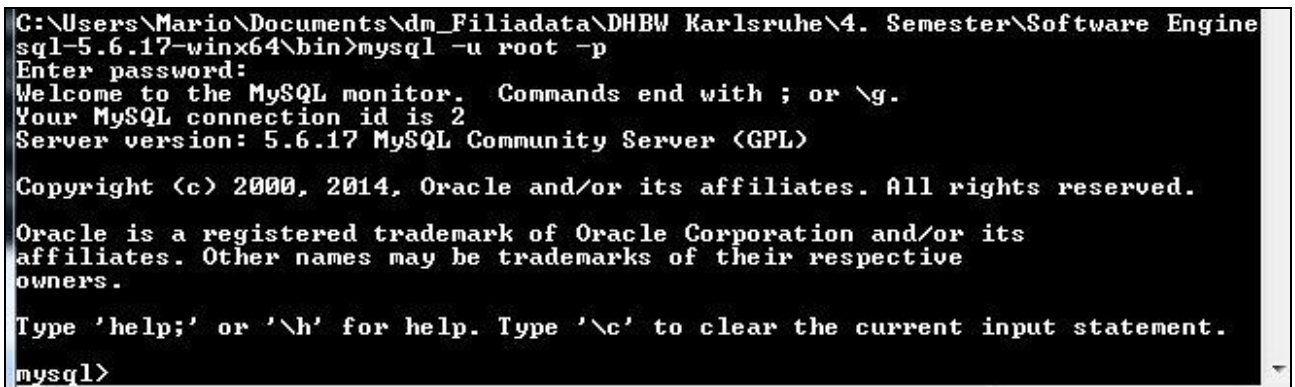
2.2 Installation MySQL 5.6.17

Der erste Schritt zu einer erfolgreichen Installation, ist das Entpacken des zip-Archivs. Nun wird um eine Datenbank zu erstellen die *mysql.exe* gestartet. Daraufhin öffnet sich die Kommandozeile und es können SQL-Befehle eingegeben werden. Falls die erforderlichen SQL-Services nun nicht im Hintergrund gestartet werden, dies bemerkt man spätestens wenn kein einziger SQL-Befehl akzeptiert wird oder wenn nach dem Start der *mysql.exe* per Mausklick das Fenster der Kommandozeile sofort und automatisch schließt, muss die Anwendung *mysqld.exe* gestartet werden. Ein Neustart des Systems hätte diese Problem auch behoben, wäre aber um einiges aufwendiger gewesen.

Alternativ zum Starten der *mysql.exe* bzw. *mysqld.exe* per Mausklick, kann diese auch über die *cmd.exe* ausgeführt werden. Dabei wechselt man mit dem Befehl „*cd Pfad/mysql/bin*“ in den bin-Ordner von MySQL 5.6.17 und gibt über den Befehl „*mysql -install*“ die Anweisung zur Installation der *mysql.exe*. Anschließend wird mit „*NET START MySQL*“ der notwendige SQL-Service gestartet. Hat die Installation geklappt, wartet der SQL-Server auf die erste „Anmeldung“ des Users (siehe Kapitel 2.3).

2.3 Datenbank mit MySQL erstellen

Nach dem Start der cmd.exe (Kommandozeile) wechselt man in den Ordner mysql/bin, in dem man nach der Installation eigentlich schon ist. Der Befehl „*mysql -u root -p*“ sorgt für das erste bekannt machen am SQL-Server des Users. Zudem wird man mit diesem Befehl als root/admin angemeldet und besitzt alle erforderlichen Berechtigungen um eine Datenbank zu erstellen und mit Inhalt zu füllen. Bei der Passwortabfrage wird standardmäßig ein „Blank“ als Passwort erwartet. (siehe Abbildung 1)



```
C:\Users\Mario\Documents\dm_Filiadata\DHBW Karlsruhe\4. Semester\Software Engine  
sql-5.6.17-winx64\bin>mysql -u root -p  
Enter password:  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 2  
Server version: 5.6.17 MySQL Community Server (GPL)  
  
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql>
```

Abbildung 1: Login SQL-Server

Nach der „Anmeldung“ am Server, erwartet die Kommandozeile ausschließlich SQL-Befehle. Um eine Datenbank anzulegen wird, wie in Abbildung 2 zu sehen, der Befehl „*create Database DatabaseName*“ genutzt, um die Datenbank ePortfolio zu erstellen.



```
mysql> create database ePortfolio;  
Query OK, 1 row affected (0.01 sec)  
  
mysql>
```

Abbildung 2: create Database

Um zu Prüfen, ob die Datenbank erstellt wurde, wird der Befehl „*show databases*“ angewandt. Der Befehl „*use ePortfolio*“ gibt an, dass die nachfolgenden Befehle für die Datenbank ePortfolio gelten. Mit „*show tables*“ wird der Datenbank-Inhalt angezeigt (enthaltene Tabellen). Zu diesem Zeitpunkt ist die Datenbank jedoch noch leer. (siehe Abbildung 3)

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| eportfolio      |
| mysql          |
| performance_schema |
| test          |
+-----+
5 rows in set (0.00 sec)

mysql> use ePortfolio;
Database changed
mysql> show tables;
Empty set (0.04 sec)

mysql>
```

Abbildung 3: Datenbank prüfen für Veränderungen markieren

Um die Datenbank mit Inhalt zu füllen wird die erste Tabelle eingefügt. Die Tabelle Student wird mit den Spalten Nachname, Vorname und MatrikelNr (siehe Abbildung 4) erstellt.

```
mysql> create table student(Nachname char(25), Vorname char(25), MatrikelNr int);
Query OK, 0 rows affected (0.12 sec)

mysql>
```

Abbildung 4: create table

Um auch die Tabellen mit Inhalt zu füllen wird der in Abbildung 5 angegebene Befehl genutzt.

```
mysql> insert into student values('Raab', 'Mario', 6176919);
Query OK, 1 row affected (0.00 sec)

mysql>
```

Abbildung 5: insert into table

Damit die erstellte Datenbank komplett angezeigt wird, kann zum Testen eine erste Abfrage eingetippt werden (siehe Abbildung 6).

```
mysql> insert into student values('Mustermann', 'Max', 0815);
Query OK, 1 row affected (0.43 sec)

mysql> insert into student values('Eberhardt', 'Fritz', 1234567);
Query OK, 1 row affected (0.00 sec)

mysql> select * from student;
+-----+-----+-----+
| Nachname | Vorname | MatrikelNr |
+-----+-----+-----+
| Raab     | Mario   | 6176919    |
| Mustermann | Max    | 815        |
| Eberhardt | Fritz   | 1234567    |
+-----+-----+-----+
3 rows in set (0.01 sec)
```

Abbildung 6: Datenbankinhalt prüfen mit Select From

Wenn ein Befehl in der Kommandozeile akzeptiert wird, erhält man zudem die Meldung Query OK... und man weiß dadurch genau, dass alles in Ordnung geht.

2.4 Datenbankverbindung über Eclipse

Zuerst wird ein neues Java Projekt angelegt bzw. ein vorhandenes Projekt ausgewählt. Hier in diesem Beispiel wurde zum Testen ein neues Projekt angelegt. (siehe Abbildung 7)

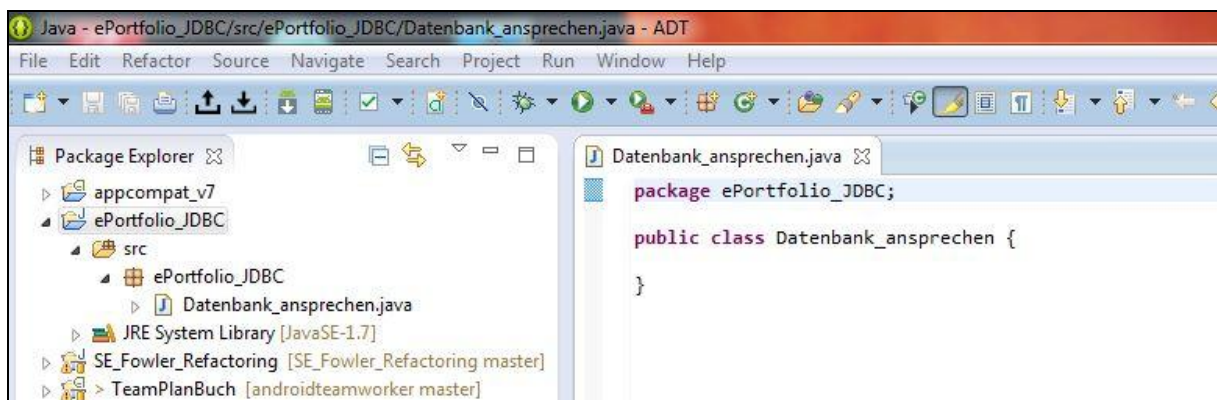


Abbildung 7: Projekt und Klasse anlegen

2.4.1 Download MySQL Connector/J

Den erforderlichen Treiber für die Datenbankverbindung kann man unter folgendem Link downloaden.

Download Connector/J: <http://dev.mysql.com/downloads/connector/j/>

2.4.2 Datenbank-Treiber der Klassenbibliothek hinzufügen

Für jede Datenbank der unterschiedlichen Hersteller sind eigene Treiber erforderlich. Es muss also für MySQL der Treiber Connector/J in die Java-Klassenbibliothek eingebunden werden.

JDBC ist Teil der Java Standard Edition, die JDBC-Klassen liegen somit in den Java packages `java.sql` und `javax.sql`.

Zum Einbinden des Connector/J wird ein Rechtsklick auf die erstellte Klasse ausgeführt. Der Menüpunkt Build Path kann über den Menüpunkt Configure Build Path... erreicht werden. (siehe Abbildung 8)

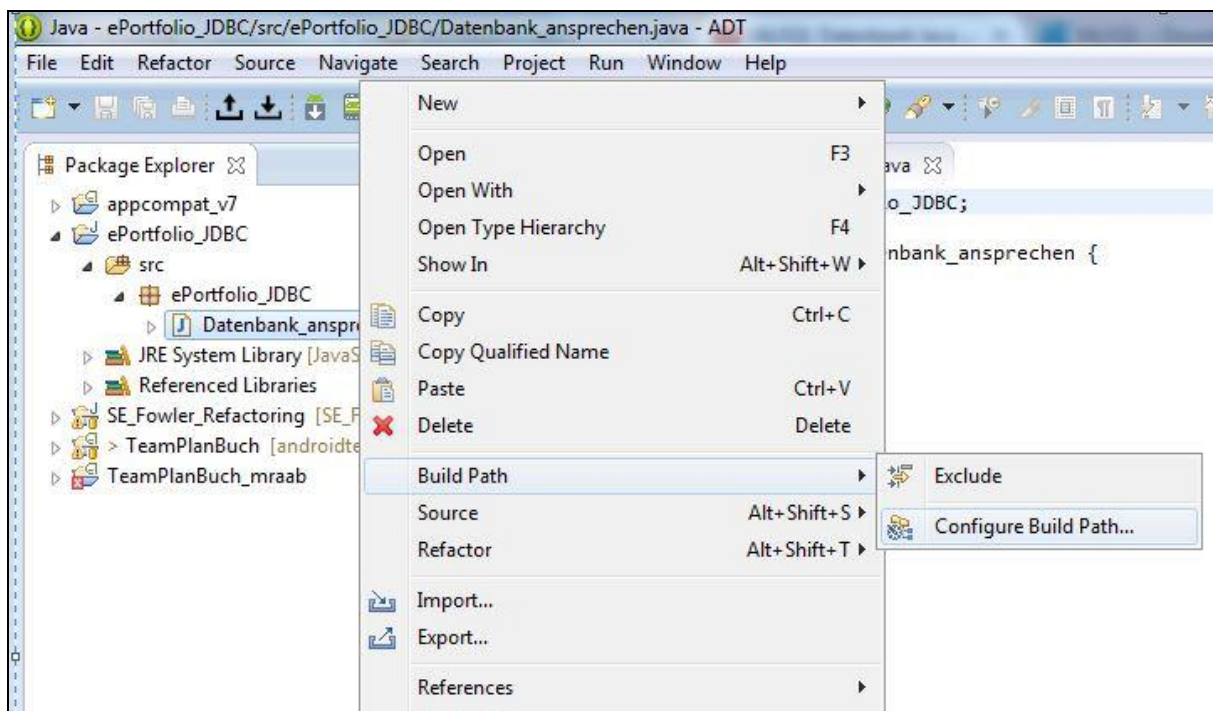


Abbildung 8: Build Path

Im Menüpunkt „Java Build Path - Libraries“ wird über das Auswahlménü „Add External JARs...“ der Pfad zur mysql-connector-java-bin.jar Datei hinzugefügt. Wodurch der treiber mit der Klasse „verknüpft/verbunden“ wird. (siehe Abbildung 9)

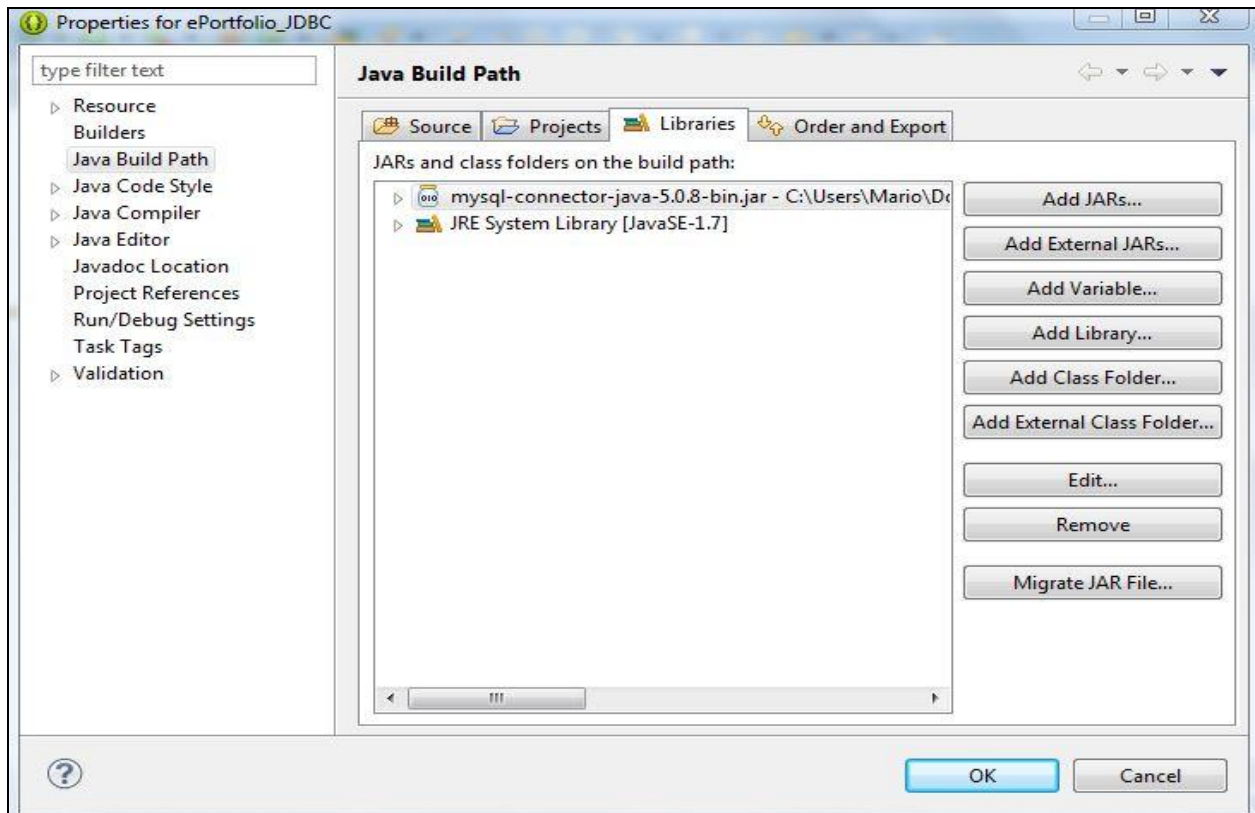


Abbildung 9: Add External JARs...

2.5 Beispiel einer Standard-Abfrage (Quellcode)

Damit zum Testen der Datenbankverbindung eine einfache SQL-Abfrage ausgeführt werden kann, wird zuerst das java-package „java.sql.*“ in die Java IDE Eclipse importiert. (siehe Abbildung 10)

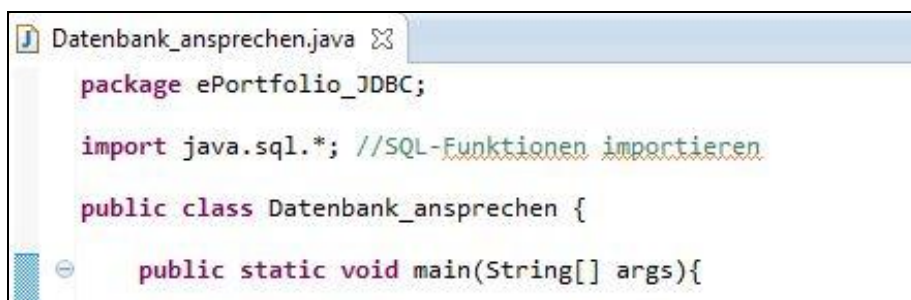


Abbildung 10: import SQL-Funktionen

Mittels `Class.forName("com.mysql.jdbc.Driver").newInstance();` wird der Treiber initialisiert. Anschließend wird die Verbindung zur Datenbank implementiert. Dabei muss der URL/Pfad/Speicherort der Datenbank (hier: `jdbc:mysql://localhost/ePortfolio`), der User (hier: `root`) und das Passwort (hier: leer) mit übergeben werden. In der while-Schleife wird für jedes Tupel in der Tabelle eine Ausgabe gemacht. (siehe Abbildung 11)

```
public static void main(String[] args){  
    try{  
        Class.forName("com.mysql.jdbc.Driver").newInstance();  
        //Verbindung zur Datenbank herstellen (URL/User root/Passwort)  
        Connection verbindung = DriverManager.getConnection("jdbc:mysql://localhost/ePortfolio","root","");  
        //Verbindung nur lesend zugreifen  
        verbindung.setReadOnly(true);  
        Statement stmt = verbindung.createStatement();  
        ResultSet ergebnis = stmt.executeQuery("Select * from student");  
        //Ausführung des SQL-Befehls  
        while(ergebnis.next()){  
            //Ausgabe des Tabelleninhalts (Spalte 1,2 und 3)  
            System.out.println(ergebnis.getString(1)+" "+ergebnis.getString(2)+" "+ergebnis.getInt(3));  
        }  
        ergebnis.close();  
        stmt.close();  
        verbindung.close();  
    }  
}
```

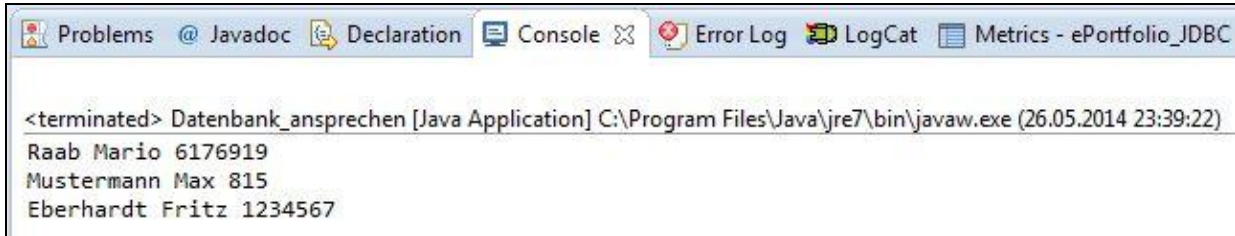
Abbildung 11: try-Anweisung

Sobald die try-Anweisung nicht ausgeführt wird, ist ein Fehler aufgetreten und deshalb wird die catch-Anweisung ausgeführt, die nur die Meldung ausgibt, dass ein Fehler aufgetreten ist. (siehe Abbildung 12)

```
//Fehlermeldung ausgeben falls try Anweisung fehlschlägt  
catch(Exception e){  
    System.out.println("**** FEHLERMELDUNG **** ->" +e);  
}  
}  
}
```

Abbildung 12: catch-Anweisung

Wurden alle Befehle erfolgreich akzeptiert, wird nach Ausführung des Quellcodes folgende Ausgabe gemacht. (siehe Abbildung 13)



```
<terminated> Datenbank_ansprechen [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (26.05.2014 23:39:22)
Raab Mario 6176919
Mustermann Max 815
Eberhardt Fritz 1234567
```

Abbildung 13: Ausgabe

3 Forum

Link zum Git Repository

- https://github.com/mraab89/Tutorial_Java_Database_Connectivity_JDBC

Link zum Forum stackoverflow

- <http://stackoverflow.com/questions/17426052/error-2003-hy000-cant-connect-to-mysql-server-on-localhost-10061>