



JAVAFX FUSSBALLVEREIN



Ibrahim Keles 11.06.2016
[FIRMENNAME] [Firmenadresse]

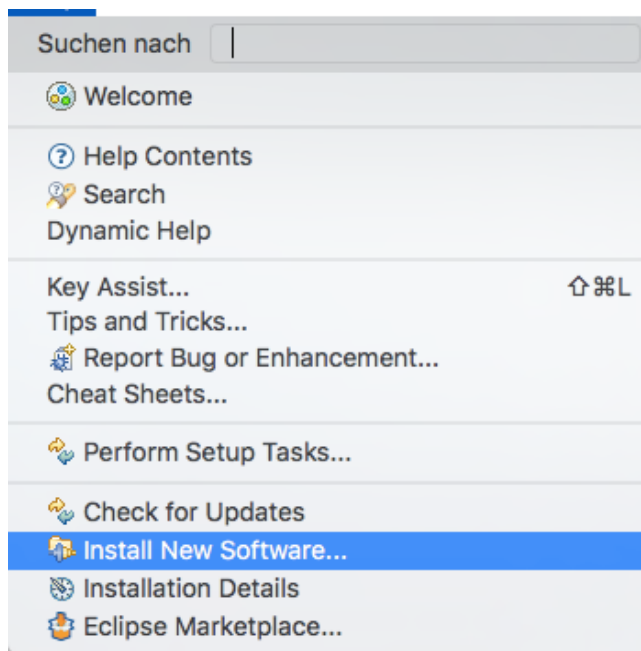
Inhalt

1.Configuration	2
2.Gui.....	3
2.1Login.....	3
2.2Select.....	3
2.3Create.....	4
2.4 Delete	5
2.5 Update	5
3.Codesnippets.....	6
3.1Main	6
3.2Fussballcontroller	7
4.SpieltThreads	11
4.1Threads Main	12
5.Zeitaufwand	12
6.Quellen	13

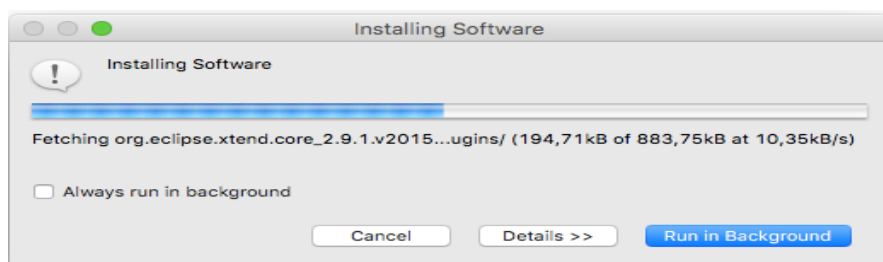
1.Configuration

Um die Aufgabe in JavaFx muss das Plugin für JavaFx heruntergeladen werden unter dem Link findet man die Installation :<http://download.eclipse.org/efxclipse/updates-released/2.3.0/site>

Nächster Schritt ist auf Help-> Installation neuer Software gehen



Nach der Betätigung und der Eingabe des Linkes kommt man zu einem Ladeschein wo die IDE Eclipse die benötigten Software herunterlasset und installiert



Nach der Installation sieht man das man beim erstellen eines Projekte einfach auf Other geht und somit kann man ein JavaFx Projekt starten.

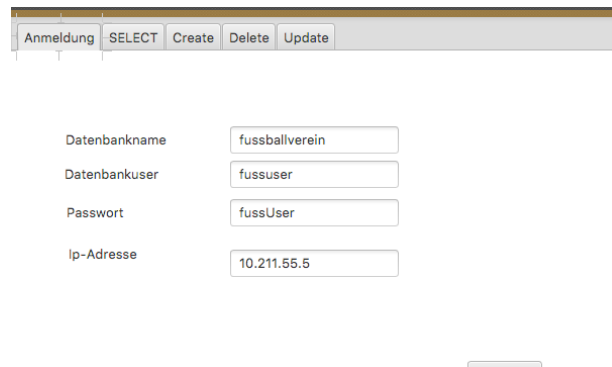
Nach dieser Installation lade ich mir den Java Fx Scene Builder damit ich die Gui besser gestalten kann .Link:<http://www.oracle.com/technetwork/java/javafxscenebuilder-1x-archive-2199384.html>

2.Gui

In meine Giu habe ich mehrere Tabs die jeder ein Crud Befehl beinhaltet zb . Read,Create,Update und Delete.

Diese sieht zurzeit so aus:

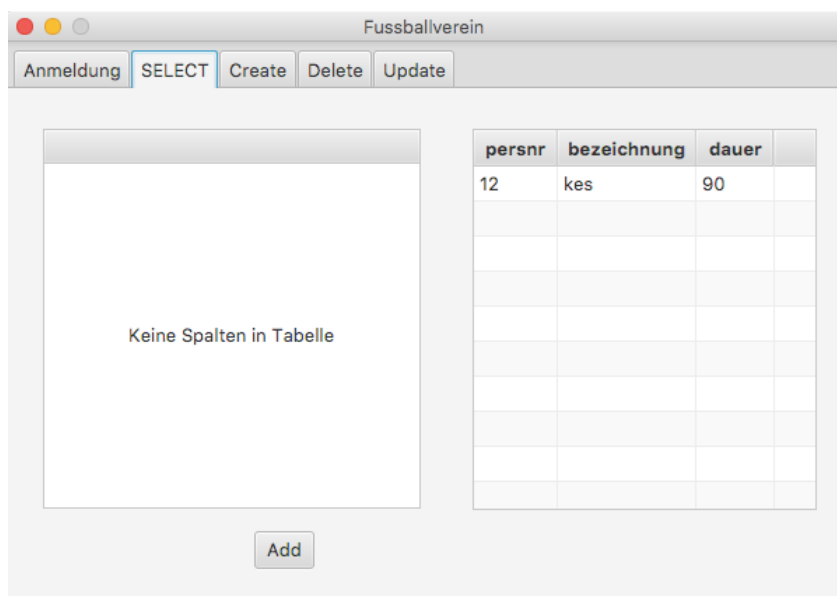
2.1Login



The screenshot shows a GUI window with a tabbed interface. The tabs are labeled 'Anmeldung', 'SELECT', 'Create', 'Delete', and 'Update'. The 'Anmeldung' tab is currently selected. Below the tabs, there are four input fields with labels: 'Datenbankname' (containing 'fussballverein'), 'Datenbankuser' (containing 'fussuser'), 'Passwort' (containing 'fussUser'), and 'Ip-Adresse' (containing '10.211.55.5').

2.2Select

Innerhalb des Selectes habe ich zwei Tabellen gesetzt damit ich die die Information von den beiden Tabellen der Datenbank habe falls sich was geändert hat und damit ich diese auch mit dem Thread den ich für die beiden „Attribute“ zur Verfügung gestellt habe.



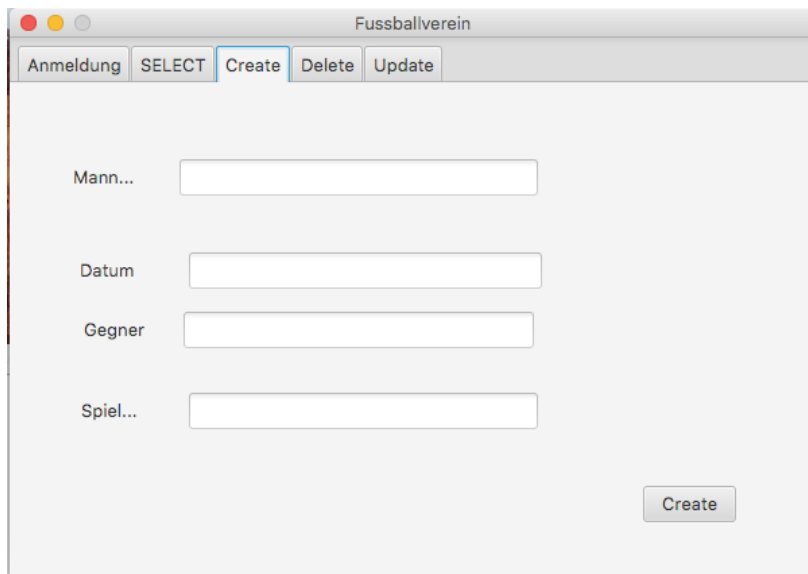
The screenshot shows a GUI window titled 'Fussballverein' with a tabbed interface. The tabs are labeled 'Anmeldung', 'SELECT', 'Create', 'Delete', and 'Update'. The 'SELECT' tab is currently selected. On the left side of the 'SELECT' tab, there is a large empty box with the text 'Keine Spalten in Tabelle' at the bottom. On the right side, there is a table with the following structure:

persnr	bezeichnung	dauer	
12	kes	90	

At the bottom of the window, there is an 'Add' button.

2.3 Create

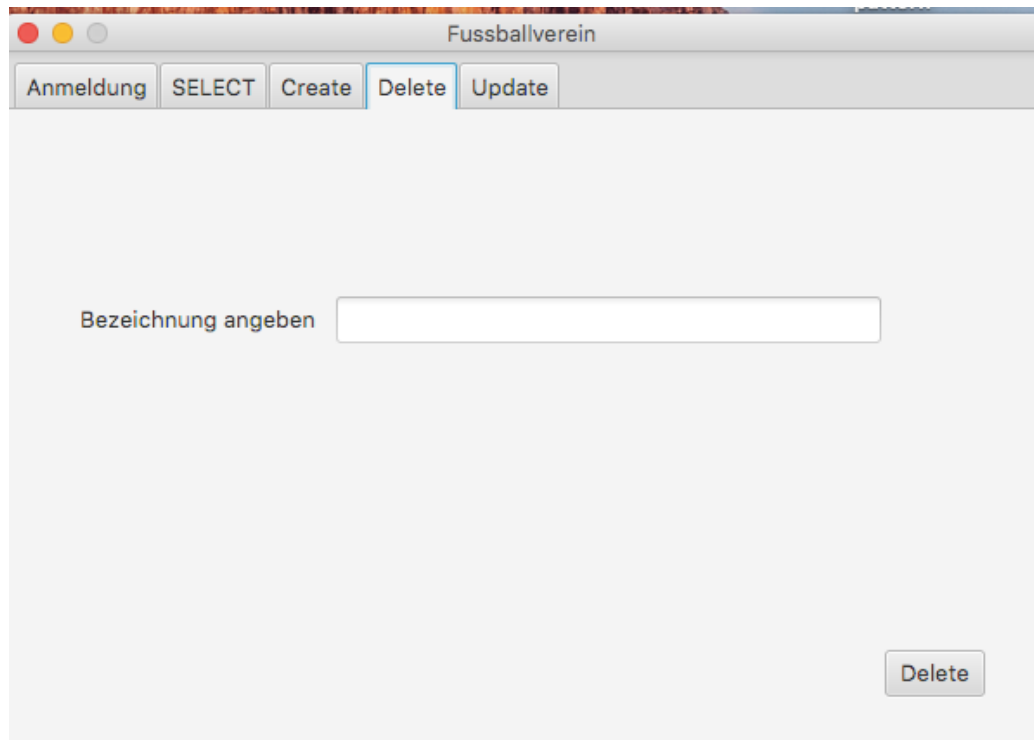
In diesem Tab habe ich die Inserts erstellt für die Tabelle Spiel damit diese dann wieder in die Datenbank geschrieben wird und wieder in die Tableview.



The screenshot shows a window titled "Fussballverein" with a tabbed interface. The "Create" tab is selected, showing four input fields for data entry: "Mann...", "Datum", "Gegner", and "Spiel...". Each field is represented by a text label followed by an empty rectangular input box. At the bottom right of the form area, there is a button labeled "Create". Above the input fields, a row of tabs is visible: "Anmeldung", "SELECT", "Create" (which is highlighted with a blue border), "Delete", and "Update".

2.4 Delete

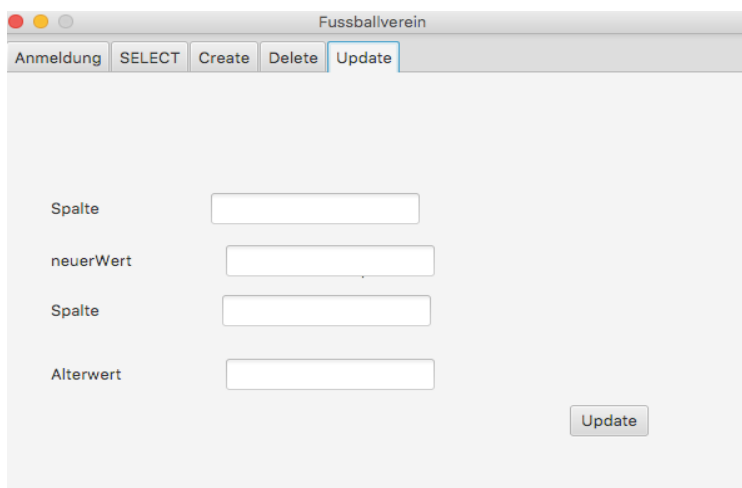
In diesem Tab wird das Sql Statement Delete ausgeführt. Dies habe ich mit der Tabelle Spiel gemacht. In der habe ich die Bezeichnung genommen weil sie etwas eindeutiges ist in dieser Tabelle.



The screenshot shows a window titled 'Fussballverein' with a tabbed interface. The tabs are 'Anmeldung', 'SELECT', 'Create', 'Delete', and 'Update'. The 'Delete' tab is currently selected and highlighted with a blue border. Inside the 'Delete' tab, there is a label 'Bezeichnung angeben' followed by a text input field. At the bottom right of the window, there is a button labeled 'Delete'.

2.5 Update

Als nächstes habe ich das Update Statement benutzt die auf die Tabelle Spiel zugreift und je nach will Attribute in der Tabelle ändert.



The screenshot shows the same 'Fussballverein' window, but now the 'Update' tab is selected and highlighted with a blue border. The 'Delete' tab is no longer visible. The 'Update' tab contains four labels with corresponding text input fields: 'Spalte', 'neuerWert', 'Spalte', and 'Alterwert'. At the bottom right of the window, there is a button labeled 'Update'.

3.Codesnippets

3.1Main

Die Aufgabe der Main-Methode ist das alle Element also View ,Controller usw. harmonisieren und somit eine Ausgabe betätigt. Also kommen alle Komponenten dieser Applikation in diese Klasse. In dieser Klasse werden die Threads laufengelassen damit sie ausgeführt werden.Zu den Threads komme ich später

```
//FussballvereinController fu =new FussballvereinController();
//SpielThreads T1 ,T2;
try {

    Parent root =
FXMLLoader.load(getClass().getResource("/application/fussballverein.fxml
1"));

    Scene scene = new Scene(root);

    scene.getStylesheets().add(getClass().getResource("application.css
").toExternalForm());
    primaryStage.setScene(scene);

    primaryStage.setTitle("Fussballverein");
    primaryStage.setResizable(false);
    primaryStage.show();
} catch (Exception e) {
    e.printStackTrace();
}

}

public static void main(String[] args) {

//    Thread T1 = new SpieltThreads(wert, fU);
//    Thread spielt =new SpieltThreads(wert, fU);
//    T1.start();
//    T2.start();

    // wait for threads to end
    //try {
//        T1.join();
//        T2.join();
//    } catch (Exception e) {
//        System.out.println("Interrupted");
//    }
```

3.2 Fussballcontroller

```
@FXML
private Label dbinformation, dauerLabel;
@FXML
private TextField
dbname, dbuser, dbpasswd, dbip, dbport, createneTab, datumInsert, gegnerTab, sp
ieleTab, valueHin, updateTab, alttabel, newColm, newValue;

@FXML
private Button connect ;
```

In dieser Klasse werden alle Gut Elemente initialisiert und werden dann ausgegeben.

In diesem Beispiel wird die Datenbankverbindung mittels JDBC gemacht. Um diese zu Verbinden muss das jar file der jeweilige Datenbank . In dem Beispiel war das Postgres. Außerdem sollte man beachten wie die Postgres.conf Datei geschrieben ist . Listen.dress =*; sollte reingeschrieben werden und pg_hba.conf diese 2 Dateien sich in Verzeichnis /etc/postgresql/9.5/.

```
public void connection(ActionEvent e )
{
    // Datenquelle erzeugen und konfigurieren
    ds = new PGSimpleDataSource();
    ds.setServerName(dbip.getText());
    ds.setDatabaseName(dbname.getText());
    ds.setUser(dbuser.getText());
    ds.setPassword(dbpasswd.getText());
}
```

Ich hab ActionEvent benutzt das ich dan bei betätigen des Button von der Anmeldung sich etwas tut .-> Verbindung der Datenbank.

In der Klasse table() und talbespielt werden die Daten von Tabellen spiel und spielt ausgelesen und in eine Tabelview ausgegeben . Hier lag die Schwierigkeit die einzelne Spalten auszulesen . table habe ich in diesem Beispiel habe ich mit einem button gemacht damit man die Erneuerung bekommt die in der Datenbank ausgeführt sind.

Außerdem braucht man bei jeder Einzelne Methode die geschrieben wurde und die eine Verbindung zur Datenbank hat ein neues connect.

```
data = FXCollections.observableArrayList();
con=ds.getConnection();
String SQL = "SELECT * from spielt";
```

```
for(int i=0 ; i<rs.getMetaData().getColumnCount(); i++){
    //We are using non property style for making
dynamic table
        final int j = i;
        TableColumn col = new
TableColumn(rs.getMetaData().getColumn(i+1));
        col.setCellValueFactory(new
Callback<CellDataFeatures<ObservableList,String>,ObservableValue<String>
>>(){
            public ObservableValue<String>
call(CellDataFeatures<ObservableList, String> param) {
                return new
SimpleStringProperty(param.getValue().get(j).toString());
            }
        });
```

Das Problem für die Sql wurden so gelöst , das hier PreaperdStatments genutzt wurde . Diese ersetzen im Sql Statement Frage zeichnen im String mit den Statement die sie aus der Gui haben. pst.setString(gibtfragezeigen an zb(1),gibt den string an)

```

PreparedStatement pst;
String statat=datumInsert.getText();
String insertString = "INSERT INTO Spiel VALUES (?, '"
+statat+"', ?, ?)";
pst = con.prepareStatement(insertString);
con.setAutoCommit(false);

// named Person] which in this set of statements we
shall call 'p' within the list
// previously defined and named 'mylist' ... or "For
each Person 'p' in 'mylist'"
String mannschaft = createneTab.getText(); // get the
name which corresponds to the Person in this object of 'mylist'

String gegner =gegnerTab.getText();// ditto, phone. Did
as integer here to show how to add to pst below
String ergebnis=spieleTab.getText();

pst.setString(1, mannschaft); // replace question mark
1 with value of 'name
pst.setString(2, gegner); // ditto, 3 and 'phone'
pst.setString(3, ergebnis);

```

Außerdem wurde in dem ein weiteres Programm geschrieben damit die Table geleert und erneuert werden kann.

```

public void aktual()
{
    tableZeichen.setColumns().clear();
    data.clear();
    data = FXCollections.observableArrayList();

}

```

Eine weitere Aufgabe ist das für Spieldauer, Spielerzeit in einem Thread ausgegeben muss. Dazu wurden 2 weitere Methode mit den ResultSet erzeugt und in einer while schleife ausgegeben in dem man genau angibt welches Attribut will. rs.getString(gewünschte String im Statement mit einer Zahl zb (2)),

```
con=ds.getConnection();
Statement st=con.createStatement();
ResultSet rs =st.executeQuery("select
Spieler.persnr,dauer from spielt inner join Spieler on Spieler.persnr =
spielt.persnr;");
while(rs.next()){

    String wert=rs.getString(2);
```

4.SpieltThreads

In der Klasse wurde versucht die beiden Methoden in zwei verschiedene threads einzuspeichern und somit sie auszugeben. In diesem Fall wurde extends Threads benutzt. Es wurde ein Objekt von FussballController erzeugt und somit die Methoden zuthreaden. In der Methode run wird synchronized() benutzt . Synchronized ist dazu da damit sich nicht 2 Threads nicht überschreiben . Außerdem wurde Thread.sleep(50) benutzt damit eine Aktualisierung stattfindet innerhalb von ca 50 Millisekunden.

```
@Override
public void run() {
    try
    {
        synchronized (fU) {
            fU.abrufenderKlasse();
            Thread.sleep(5);
        }
    } catch (Exception e)
    {
        System.out.println(e);
    }

}

public void start()
{
    if(t==null)
    {

        t=new Thread();
        t.start();
    }
}
```

4.1 Threads Main

In der Main Methode wird die Threads initialisiert und damit ausgerufen -> Objekt erzeugt. Mit T1.start(); beginnt der Thread und mit T1.join wird der Thread beendet. Leider funktioniert diese nicht und ich hab sie kommentiert damit sich die Gui starten lässt .

```
// Thread T1 = new SpieltThreads(wert, fU);  
    //Thread spielt =new SpieltThreads(wert, fU);  
    // T1.start();  
    // T2.start();  
  
    // wait for threads to end  
    //try {  
    //    T1.join();  
    //    T2.join();  
    // } catch( Exception e) {  
    //    System.out.println("Interrupted");  
    // }
```

5. Zeitaufwand

JavaFx	13 Stunden
--------	------------

6.Quellen

Tableview : <http://stackoverflow.com/questions/18941093/how-to-fill-up-a-tableview-with-database-data>

Threads :

https://www.dpunkt.de/java/Programmieren_mit_Java/Multithreading/3.html

<https://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html>

<http://stackoverflow.com/questions/2531938/java-thread-example>

https://docs.oracle.com/cd/F49540_01/DOC/java.815/a64685/tips1.htm

<http://stackoverflow.com/questions/10697311/runnable-interface-example>

<http://stackoverflow.com/questions/7313657/should-you-synchronize-the-run-method-why-or-why-not>

<http://stackoverflow.com/questions/13238618/set-text-in-label-during-an-action-event>

<http://stackoverflow.com/questions/17341223/retrieve-from-database-into-jlabel>

http://www.tutorialspoint.com/java/java_thread_synchronization.htm

http://www.tutorialspoint.com/java/java_multithreading.htm

<https://docs.oracle.com/javase/tutorial/essential/concurrency/sleep.html>

<http://stackoverflow.com/questions/10560167/putting-runnable-to-sleep>

http://www.tutorialspoint.com/java/lang/thread_sleep_millis.htm

PreparedStatement :

<http://stackoverflow.com/questions/419021/java-prepared-statement-arguments>