

```

/*
 * Auszüge aus GUI.java
 *
 * Created on 20. September 2008, 14:24
 */

package datenbankverbindungsgui;
import java.sql.*;          //Verende SQL Package um SQL Funktionen nutzen zu können

/**
 *
 * @author Bre@ker
 */
public class GUI extends javax.swing.JFrame {
String dbQuery;

    //#####
    //#####Definiere DB VerbindungsDaten#####
    public final static String DBUSERNAME      ="admin";
    public final static String DBPASSWORT     ="root";
    public final static String DBCONNECTION   ="jdbc:derby://localhost:1527";
    public final static String DBNAME        ="Kontaktverwaltung";
    public final static String TABLENAME     ="Contacts";
    //#####

    /** Creates new form GUI */
    public GUI() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    public void dbAbfrage()                                // Wird immer dann Aufgerufen,
                                                         // wenn aus der Db gelesen werden soll

        String tempString="Keine Daten vorhanden";        // Definiere Hilfsvariablen, um
                                                         // die Erhaltenen Daten zu speichern

        String puffer="none";
        try {
            //Try Catch um bei fehlerhafter
            // Verbindung keinen Programmfehler zu erhalten
            // Lädt und Registriert den SQL Server Treiber
            Class.forName("org.apache.derby.jdbc.ClientDriver");
            // Baut die Verbindung auf (DB Server
            // Adresse:Port, Nutzernamen und Passwort
            // müssen angegeben werden)

            Connection conn =
DriverManager.getConnection(DBCONNECTION+"/"+DBNAME,DBUSERNAME,DBPASSWORT);
            //Erzeugt ein Objekt vom Typ Statement
            Statement sql_stmt = conn.createStatement();
            // Erzeugt ein Objekt vom Typ ResultSet
            // und führt die Abfrage aus (Als Ergebnis
            // wird ein ResultSet (Ergebnis Satz geliefert)
            // (Um Daten zu Lesen wird ein Query vom Typ
            // Execute benötigt)
            ResultSet rset = sql_stmt.executeQuery(dbQuery);

            boolean firstRun=true;                          // Bool Hilfsvariable (Kleine Schummellei, um
                                                         // beim ersten Durchlauf der While Schleife
                                                         //keine leere Zeile zu erhalten)

            while (rset.next()){                             //In unserer Ergebnismenge sind
            alle Datensätze (ID,Vorname,Name) gespeichert die die DB zu unserem Query gefunden hat

```

```

//Die While Schleife läuft so oft
durch wie Datensätze vorhanden sind. Mit jedem neuen Durchlauf wird ein neuer Datensatz angesprochen
// (z.B. 1ter Durchlauf ID=1
Vorname=Ulrich Nachname=Barrot || 2ter Durchlauf ID=2 Vorname=Ralf Nachname=Trierweiler uws.)
int nummer = rset.getInt("ID"); //Bei jedem Schleifendurchlauf
erfassen diese Variablen die Einzelnen Spalten des aktuellen Datensatzes
String name_field = rset.getString("FIRST_NAME");
String vorname_field = rset.getString("LAST_NAME");

if (firstRun) {tempString=nummer+"\t"+name_field+"\t"+vorname_field; firstRun=false;}
//Beim ersten Durchlauf, erfasse nur neuen Datensatz
else tempString=puffer+"\n"+nummer+"\t"+name_field+"\t"+vorname_field;
//Bei allen Anderen Schreibe die vorherigen vorne drann und trenne durch einen Umbruch

puffer=tempString;
//Hilfvariable speichert alle bisher erfassten Daten in einem String
}
txtMsg.setText(tempString);
//Schreibe den erzeugten String (enthält alle Infos der Datensätze) ins Ausgabefeld

//Schließe die Ergebnismenge und das SQL Statement
rset.close();
sql_stmt.close();
//Schließe die Datenbankverbindung
conn.close();

}
catch(Exception e) { // Falls oben ein Fehler aufgetreten ist,
// schreibe Error MSG ins Ergebnisfeld

txtMsg.setText("Verbindungsfehler!");
e.printStackTrace();
}
}

public void dbAenderung() // Wird immer dann Aufgerufen, wenn
// es an der DB etwas zu Schreiben/Löschen gibt
{
try {
//vgl Oben
Class.forName("org.apache.derby.jdbc.ClientDriver");
//vgl Oben
Connection conn =
DriverManager.getConnection(DBCONNECTION+"/"+DBNAME,DBUSERNAME,DBPASSWORT);
// vgl Oben

Statement sql_stmt = conn.createStatement();

// Um Daten zu schreiben/löschen
// wird ein Query vom Typ Update benötigt

sql_stmt.executeUpdate(dbQuery);
sql_stmt.close();
//vgl Oben
conn.close();

}
catch(Exception e) { // vgl Oben
txtMsg.setText("Verbindungsfehler!");
e.printStackTrace();
}
}
}

```

```

private void searchActionPerformed(java.awt.event.ActionEvent evt) { //GEN-
FIRST:event_searchActionPerformed
String searchString=searchField.getText(); //Lade Such String aus dem Textfeld
int searchByID=0; //Definiere Hilfsvariable
try //Try Catch Block (Wenn der Nutzer einen
String eingibt und keine Zahl, würde beim Parsen immer ein Fehler entstehen. Dies wird hier
abgefangen, die ID bleibt 0)
{
searchByID=Integer.parseInt(searchString); //Versuche String in Integer zu parsen
}
catch(Exception e){} //Exception muss hier nicht ausgewertet
dbQuery="SELECT * from "+TABLENAME+" WHERE FIRST_NAME='"+searchString+"' OR
LAST_NAME='"+searchString+"'OR ID="+searchByID; //Formuliere Query Suche nach Vorname, Nachname
oder ID durch logisches ODER verknüpft
dbAbfrage(); //Rufe dbAbfrage auf
searchField.setText(""); //Lösche eingabe im Suchfeld
} //GEN-LAST:event_searchActionPerformed

private void printAllActionPerformed(java.awt.event.ActionEvent evt) { //GEN-
FIRST:event_printAllActionPerformed
dbQuery="SELECT * from "+TABLENAME; //Formuliere Standart Query (frage/suche)
dbAbfrage(); //Führe Abfrage aus
} //GEN-LAST:event_printAllActionPerformed

private void enterButtonActionPerformed(java.awt.event.ActionEvent evt) { //GEN-
FIRST:event_enterButtonActionPerformed
if (!vorname.getText().equals("") && !name.getText().equals("")) //Prüfe ob ein Vorname
und ein Nachname eingegeben wurden
{
dbQuery="INSERT INTO "+TABLENAME+" VALUES (Default, '"+vorname.getText()+"', '"+name.getText()+"')";
//Formuliere Query (Verwende DEFAULT da ID per AUTO INCR. erhöht wird)
dbAenderung(); //Methode abAenderung
(Achreibt die neuen Daten in die DB)
dbQuery="SELECT * from "+TABLENAME; //Abfrage Query
dbAbfrage(); //Frage neue
Datenbanksituation ab
}
vorname.setText("");
name.setText("");
} //GEN-LAST:event_enterButtonActionPerformed

private void deleteButtonActionPerformed(java.awt.event.ActionEvent evt) { //GEN-
FIRST:event_deleteButtonActionPerformed
if (!deleteField.getText().equals("")) //Prüfe ob der Nutzer eine ID eingegeben
hat
{
int deleteID=Integer.parseInt(deleteField.getText()); //Parse String ID in Integer
dbQuery="DELETE FROM "+TABLENAME+" WHERE ID="+deleteID; //Formuliere Query
dbAenderung(); //Rufe Methode abAenderung auf
dbQuery="SELECT * from "+TABLENAME; //Formuliere neues Query
dbAbfrage(); //Frage die neue Datenbanksituation ab
deleteField.setText(""); //Lösche ID aus dem Textfeld
}
}

```