Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет» Кафедра ИИТ

Лабораторная работа №10

Выполнил:

Студент группы ПО-5

Пищик А. В.

Проверил:

Крощенко А. А.

Брест 2022

**Вариант 11**

**Цель**: приобрести практические навыки разработки многооконных приложений на JavaFX для работы с базами данных.

**Общее задание**

* На основе БД, разработанной в лабораторной работе No9, реализовать многооконное приложение-
* клиент, позволяющее выполнять основные операции над таблицей в БД (добавление, удаление, мо-
* дификацию данных).
* Основные требования к приложению:
* • Для отображения выбирать таблицу с внешними ключами;
* • Осуществлять вывод основных данных в табличном представлении;
* • При выводе краткого представления записи в таблице (т.е. если выводятся не все поля), по
* щелчку мышкой на запись осуществлять вывод всех полей в подготовленные компоненты на
* форме;
* • Для всех полей, представленных внешними ключами, выводить их текстовое представление
* из связанных таблиц (например, таблица-справочник «Времена года» содержит два поля –
* идентификатор и название сезона, в связанной таблице «Месяц года» есть внешний ключ на
* таблицу «Времена года»; в этом случае при выводе таблицы «Месяц года» нужно выводить
* название сезона, а не его идентификатор);
* • При выводе предусмотреть упорядочивание по столбцу;
* • Реализовать простейший фильтр данных по одному-двум полям;
* • При добавлении новых данных в таблицу использовать дополнительное окно для ввода;
* • При модификации данных можно использовать ту же форму, что и для добавления, но с вне-
* сенными актуальными значениями полей;
* • При добавлении/модификации выводить варианты значений полей с внешним ключом с помо-
* щью выпадающего списка;
* • При удалении данных осуществлять удаление записи, на которой в данных момент находится
* фокус.

**Info**

package com.example.lab4.Controllers;

import com.example.lab4.Connector.DBConnector;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.fxml.FXML;

import javafx.scene.text.Text;

public class Info {

public Text idGUI;

public Text companyGUI;

public Text nameGUI;

public Text branchesGUI;

public Text countcompGUI;

public Text addressGUI;

public Text nameofficeGUI;

public Text fnameGUI;

public static int id;

public static String company;

public static String name;

public static String branches;

public static String countcomp;

public static String address;

public static String nameoffice;

public static String fname;

public static void getData(int ID, String Company, String Name, String Branches, String Countcomp, String Adress, String Nameoffice, String Fname) {

id = ID;

company = Company;

name = Name;

branches = Branches;

countcomp = Countcomp;

address = Adress;

nameoffice = Nameoffice;

fname = Fname;

}

@FXML

void initialize() throws ClassNotFoundException {

try {

idGUI.setText(String.valueOf(id));

companyGUI.setText(company);

nameGUI.setText(name);

branchesGUI.setText(branches);

countcompGUI.setText(countcomp);

addressGUI.setText(address);

nameofficeGUI.setText(nameoffice);

fnameGUI.setText(fname);

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

}

**UpDateController**

package com.example.lab4.Controllers;

import com.example.lab4.Connector.DBConnector;

import com.example.lab4.Model.Partments;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.scene.control.Button;

import javafx.scene.control.ComboBox;

import javafx.scene.control.TextField;

import javafx.stage.Stage;

public class UpDateController {

@FXML

public Button bottonOk;

@FXML

public TextField Name;

@FXML

public ComboBox<String> Company;

@FXML

public ComboBox<String> LastName;

@FXML

public ComboBox<String> Number;

@FXML

public ComboBox<String> Adress;

public static int id;

public static String company;

public static String name;

public static String branches;

public static String countcomp;

public static String adress;

Partments partments;

public static void setPartments(int ID, String Company, String Name, String Branches, String Countcomp, String Adress) {

id = ID;

company = Company;

name = Name;

branches = Branches;

countcomp = Countcomp;

adress = Adress;

}

@FXML

void initialize() {

try {

Name.setText(name);

ObservableList<String> CompanyLIST = FXCollections.observableArrayList(DBConnector.getCompany());

Company.setItems(CompanyLIST);

Company.setValue(company);

ObservableList<String> LastNameLIST = FXCollections.observableArrayList(DBConnector.getLastName());

LastName.setItems(LastNameLIST);

LastName.setValue(branches);

ObservableList<String> AdressLIST = FXCollections.observableArrayList(DBConnector.getAdress());

Adress.setItems(AdressLIST);

Adress.setValue(adress);

ObservableList<String> NumberLIST = FXCollections.observableArrayList(DBConnector.getNumbers());

Number.setItems(NumberLIST);

Number.setValue(countcomp);

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

public void onClickUpdate(ActionEvent actionEvent) throws ClassNotFoundException {

if(Company.getValue() == null || Name.getText() == null ||

LastName.getValue() == null || Number.getValue() == null || Adress.getValue() == null) return;

partments = new Partments(id,Company.getValue(), Name.getText(), LastName.getValue(), Number.getValue(), Adress.getValue());

DBConnector.updatePartments(partments);

Stage stage = (Stage) bottonOk.getScene().getWindow();

stage.close();

}

}

**HelloController**

package com.example.lab4.Controllers;

import com.example.lab4.Connector.DBConnector;

import com.example.lab4.Model.Partments;

import javafx.beans.value.ChangeListener;

import javafx.beans.value.ObservableValue;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.stage.Stage;

import java.io.IOException;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

public class HelloController {

public HelloController() throws SQLException, ClassNotFoundException {

bdController = new DBConnector(this);

}

public TableView<Partments> table;

@FXML

private TableColumn<Partments, String> idColumn;

@FXML

private TableColumn<Partments, String> nameColumn;

@FXML

private TableColumn<Partments, String> nameCompanyColumn;

@FXML

private TableColumn<Partments, String> lNameColumn;

@FXML

private TableColumn<Partments, String> NumberColumn;

@FXML

private TableColumn<Partments, String> AdressColumn;

public ObservableList<Partments> listview = FXCollections.observableArrayList();

public int ID;

public String company;

public String name;

public String branches;

public String countcomp;

public String adress;

private final DBConnector bdController;

@FXML

public Button Add;

@FXML

public Button Change;

@FXML

public Button Delete;

@FXML

public Button Refresh;

@FXML

public Button AllInfo;

@FXML

void initialize() {

try {

listview.addAll(bdController.connection());

} catch (ClassNotFoundException | SQLException e) {

throw new RuntimeException(e);

}

idColumn.setCellValueFactory(new PropertyValueFactory<Partments, String>("id"));

nameCompanyColumn.setCellValueFactory(new PropertyValueFactory<Partments, String>("CompanyName"));

nameColumn.setCellValueFactory(new PropertyValueFactory<Partments, String>("Name"));

lNameColumn.setCellValueFactory(new PropertyValueFactory<Partments, String>("LName"));

NumberColumn.setCellValueFactory(new PropertyValueFactory<Partments, String>("Number"));

AdressColumn.setCellValueFactory(new PropertyValueFactory<Partments, String>("Adress"));

table.setItems(listview);

table.getSortOrder().add(idColumn);

TableView.TableViewSelectionModel<Partments> selectionModel = table.getSelectionModel();

selectionModel.selectedItemProperty().addListener(new ChangeListener<Partments>(){

@Override

public void changed(ObservableValue<? extends Partments> observableValue, Partments oldval, Partments newVal) {

if(newVal != null) {

ID = newVal.getId();

company = newVal.getCompanyName();

name = newVal.getName();

branches = newVal.getLName();

countcomp = newVal.getNumber();

adress = newVal.getAdress();

}

}

});

}

public void onClickRefresh() throws SQLException, ClassNotFoundException {

listview.clear();

listview.addAll(bdController.connection());

table.setItems(listview);

}

public void onClickAdd(ActionEvent actionEvent) {

try {

FXMLLoader fxmlLoader = new FXMLLoader();

fxmlLoader.setLocation(getClass().getResource("/com/example/lab4/insert.fxml"));

Scene scene = new Scene(fxmlLoader.load(), 400, 400);

Stage stage = new Stage();

stage.setTitle("Add Element");

stage.setScene(scene);

stage.show();

} catch (IOException e) {

System.out.println(e.getMessage());

Logger logger = Logger.getLogger(getClass().getName());

logger.log(Level.SEVERE, "Failed to create new Window.", e.getMessage());

}

}

public void onClickChange(ActionEvent actionEvent) {

try {

FXMLLoader fxmlLoader = new FXMLLoader();

UpDateController.setPartments(ID, company, name, branches, countcomp, adress);

fxmlLoader.setLocation(getClass().getResource("/com/example/lab4/update.fxml"));

Scene scene = new Scene(fxmlLoader.load(), 400, 400);

Stage stage = new Stage();

stage.setTitle("Update Element");

stage.setScene(scene);

stage.show();

} catch (IOException e) {

System.out.println(e.getMessage());

Logger logger = Logger.getLogger(getClass().getName());

logger.log(Level.SEVERE, "Failed to create new Window.", e.getMessage());

}

}

public void onClickDelete(ActionEvent actionEvent) throws SQLException, ClassNotFoundException {

if(ID < 1) return;

DBConnector.deletePartments(ID);

onClickRefresh();

ID = 0;

}

public void onClickAllInfo(ActionEvent actionEvent) throws SQLException, ClassNotFoundException {

try {

FXMLLoader fxmlLoader = new FXMLLoader();

if(ID < 1) return;

System.out.println(ID);

DBConnector.getAllInfo(ID);

fxmlLoader.setLocation(getClass().getResource("/com/example/lab4/Info.fxml"));

Scene scene = new Scene(fxmlLoader.load(), 600, 400);

Stage stage = new Stage();

stage.setTitle("All Info Element");

stage.setScene(scene);

stage.show();

} catch (IOException e) {

System.out.println(e.getMessage());

Logger logger = Logger.getLogger(getClass().getName());

logger.log(Level.SEVERE, "Failed to create new Window.", e.getMessage());

}

}

}

**FromInsertDBController**

package com.example.lab4.Controllers;

import com.example.lab4.Connector.DBConnector;

import com.example.lab4.Model.Partments;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.scene.control.Button;

import javafx.scene.control.ComboBox;

import javafx.scene.control.TextField;

import javafx.stage.Stage;

import java.sql.SQLException;

public class FromInsertDBController {

@FXML

public Button bottonOk;

@FXML

public TextField Name;

@FXML

public ComboBox<String> Company;

@FXML

public ComboBox<String> LastName;

@FXML

public ComboBox<String> Number;

@FXML

public ComboBox<String> Adress;

Partments partments;

@FXML

void initialize() throws ClassNotFoundException {

try {

ObservableList<String> CompanyLIST = FXCollections.observableArrayList(DBConnector.getCompany());

Company.setItems(CompanyLIST);

ObservableList<String> LastNameLIST = FXCollections.observableArrayList(DBConnector.getLastName());

LastName.setItems(LastNameLIST);

ObservableList<String> AdressLIST = FXCollections.observableArrayList(DBConnector.getAdress());

Adress.setItems(AdressLIST);

ObservableList<String> NumberLIST = FXCollections.observableArrayList(DBConnector.getNumbers());

Number.setItems(NumberLIST);

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

public void onClickInsert(ActionEvent actionEvent) throws SQLException, ClassNotFoundException {

if(Company.getValue() == null || Name.getText() == null ||

LastName.getValue() == null || Number.getValue() == null || Adress.getValue() == null) return;

partments = new Partments(3,Company.getValue(), Name.getText(), LastName.getValue(), Number.getValue(), Adress.getValue());

DBConnector.insertPartments(partments);

Stage stage = (Stage) bottonOk.getScene().getWindow();

stage.close();

}

}

**DBConnector**

package com.example.lab4.Connector;

import com.example.lab4.Controllers.HelloController;

import com.example.lab4.Controllers.InfoController;

import com.example.lab4.Model.Partments;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class DBConnector {

public static HelloController hellContr;

public DBConnector(HelloController helloController) throws SQLException, ClassNotFoundException {

hellContr = helloController;

}

public List<Partments> connection() throws ClassNotFoundException, SQLException {

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

System.out.println("Completed!");

Statement statement = connection.createStatement();

try {

List<Partments> list = new ArrayList<>();

ResultSet result;

result = statement.executeQuery("SELECT Partments.id, Partments.name, Company.name as COMPANY, Branches.lname as Last, countcomp.count as Count, Adress.adress as ADRESS FROM Partments INNER JOIN " +

"Company ON Partments.company = Company.id INNER JOIN " +

"Branches ON Partments.branches = Branches.id INNER JOIN " +

"countcomp ON Partments.countcomp = countcomp.id INNER JOIN " +

"Adress ON Partments.adress = Adress.id");

while (result.next()) {

list.add(new Partments(

result.getInt("Partments.id"),

result.getString("COMPANY"),

result.getString("Partments.name"),

result.getString("Last"),

result.getString("Count"),

result.getString("ADRESS")));

}

return list;

}

catch (Exception exception) {

System.out.println(exception.getMessage());

}

}

return null;

}

public static List<String> getCompany() throws ClassNotFoundException {

List<String> list = new ArrayList<>();

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "SELECT \* FROM Company";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(sql);

while (resultSet.next()){

list.add(resultSet.getString("name"));

}

} catch (SQLException throwables) {

throwables.printStackTrace();

}

return list;

}

public static List<String> getAdress() throws ClassNotFoundException {

List<String> list = new ArrayList<>();

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "SELECT \* FROM Adress";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(sql);

while (resultSet.next()){

list.add(resultSet.getString("adress"));

}

} catch (SQLException throwables) {

throwables.printStackTrace();

}

return list;

}

public static List<String> getLastName() throws ClassNotFoundException {

List<String> list = new ArrayList<>();

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "SELECT \* FROM Branches";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(sql);

while (resultSet.next()){

list.add(resultSet.getString("lname"));

}

} catch (SQLException throwables) {

throwables.printStackTrace();

}

return list;

}

public static List<String> getNumbers() throws ClassNotFoundException {

List<String> list = new ArrayList<>();

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "SELECT \* FROM countcomp";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(sql);

while (resultSet.next()){

list.add(resultSet.getString("count"));

}

} catch (SQLException throwables) {

throwables.printStackTrace();

}

return list;

}

public static void insertPartments(Partments partments) throws ClassNotFoundException {

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "insert into Partments (company, name, branches, countcomp, adress) " +

"values (" +

"(SELECT id FROM Company where name = ?)" +

",(?)," +

"(SELECT id FROM Branches where lname = ?)," +

"(SELECT id FROM countcomp where count = ?)," +

"(SELECT id FROM Adress where adress = ?))";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, partments.getCompanyName());

preparedStatement.setString(2, partments.getName());

preparedStatement.setString(3, partments.getLName());

preparedStatement.setString(4, partments.getNumber());

preparedStatement.setString(5, partments.getAdress());

preparedStatement.executeUpdate();

hellContr.onClickRefresh();

} catch (SQLException throwables) {

throwables.printStackTrace();

}

}

public static void updatePartments(Partments partments) throws ClassNotFoundException {

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "UPDATE Partments SET company = (SELECT id FROM Company where name = ?), " +

"name = ?, " +

"branches = (SELECT id FROM Branches where lname = ?), " +

"countcomp = (SELECT id FROM countcomp where count = ?), " +

"adress = (SELECT id FROM Adress where adress = ?) " +

"WHERE id = ?";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, partments.getCompanyName());

preparedStatement.setString(2, partments.getName());

preparedStatement.setString(3, partments.getLName());

preparedStatement.setString(4, partments.getNumber());

preparedStatement.setString(5, partments.getAdress());

preparedStatement.setInt(6, partments.getId());

preparedStatement.executeUpdate();

hellContr.onClickRefresh();

} catch (SQLException throwables) {

throwables.printStackTrace();

}

}

public static void getAllInfo(int ID) throws ClassNotFoundException, SQLException {

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

System.out.println("Completed!");

try {

ResultSet result;

String sql = "SELECT Partments.id, Partments.name, Company.name as COMPANY, Branches.nameoffice as nameoffice, Branches.lname as Last, Branches.fname as First, countcomp.count as Count, Adress.adress as ADRESS FROM Partments INNER JOIN " +

"Company ON Partments.company = Company.id INNER JOIN " +

"Branches ON Partments.branches = Branches.id INNER JOIN " +

"countcomp ON Partments.countcomp = countcomp.id INNER JOIN " +

"Adress ON Partments.adress = Adress.id WHERE Partments.id = ?";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, String.valueOf(ID));

result = preparedStatement.executeQuery();

while (result.next()) {

InfoController.getData(

result.getInt("Partments.id"),

result.getString("COMPANY"),

result.getString("Partments.name"),

result.getString("Last"),

result.getString("Count"),

result.getString("ADRESS"),

result.getString("nameoffice"),

result.getString("First"));

}

}

catch (Exception exception) {

System.out.println(exception.getMessage());

}

}

}

public static void deletePartments(int ID) throws ClassNotFoundException {

String url = "jdbc:mysql://localhost:3306/java", user = "root", passwd = "root";

Class.forName("com.mysql.cj.jdbc.Driver");

String sql = "DELETE FROM `Partments` WHERE id = ?";

try(Connection connection = DriverManager.getConnection(url, user, passwd)) {

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setInt(1, ID);

preparedStatement.executeUpdate();

hellContr.onClickRefresh();

} catch (SQLException throwables) {

throwables.printStackTrace();

}

}

}

**Partments**

package com.example.lab4.Model;

public class Partments {

int id;

String CompanyName;

String Name;

String LName;

String Number;

String Adress;

public Partments(int id, String companyName, String name, String LName, String number, String adress) {

this.id = id;

this.CompanyName = companyName;

this.Name = name;

this.LName = LName;

this.Number = number;

this.Adress = adress;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCompanyName() {

return CompanyName;

}

public void setCompanyName(String companyName) {

CompanyName = companyName;

}

public String getName() {

return Name;

}

public void setName(String name) {

Name = name;

}

public String getLName() {

return LName;

}

public void setLName(String LName) {

this.LName = LName;

}

public String getNumber() {

return Number;

}

public void setNumber(String countComp) {

Number = countComp;

}

public String getAdress() {

return Adress;

}

public void setAdress(String adress) {

Adress = adress;

}

}

**HelloApplication**

package com.example.lab4;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.stage.Stage;

import java.io.IOException;

public class HelloApplication extends Application {

@Override

public void start(Stage stage) throws IOException {

FXMLLoader fxmlLoader = new FXMLLoader(HelloApplication.class.getResource("hello-view.fxml"));

Scene scene = new Scene(fxmlLoader.load(), 1000, 540);

stage.setTitle("Lab4");

stage.setScene(scene);

stage.show();

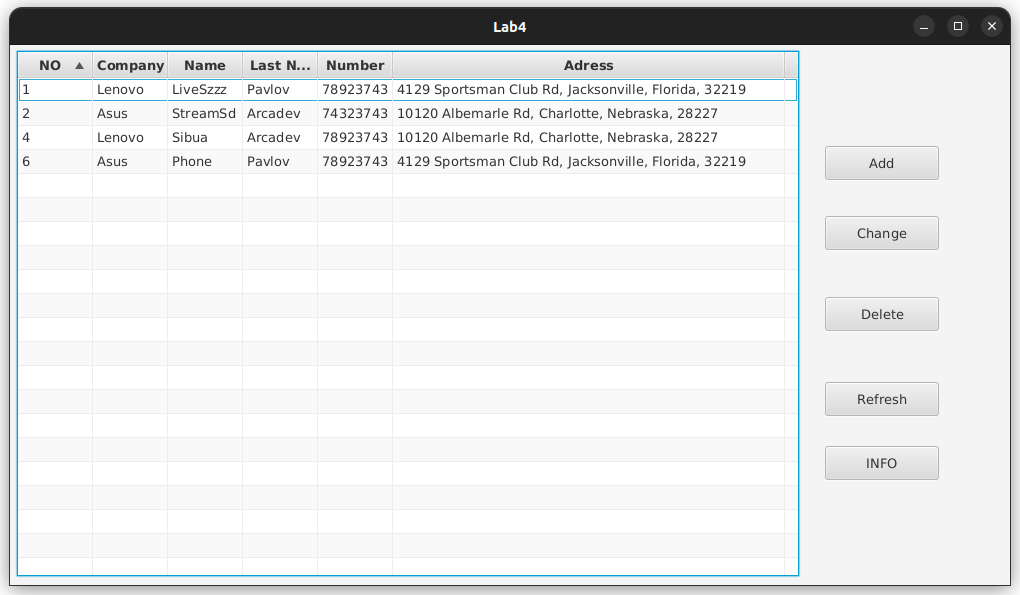
}

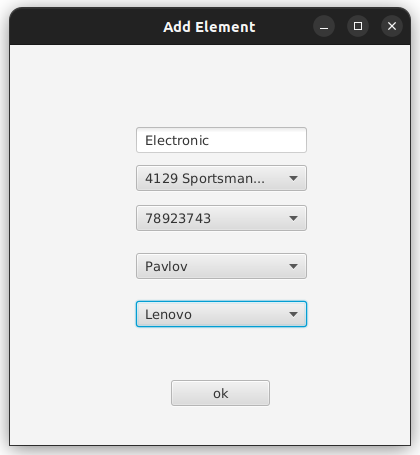
public static void main(String[] args) {

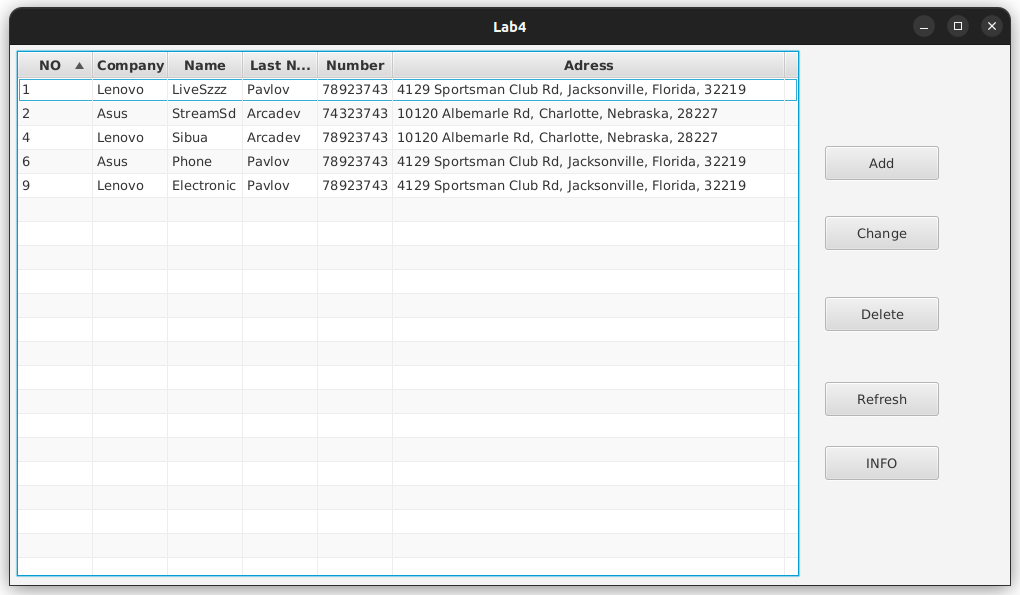
launch();

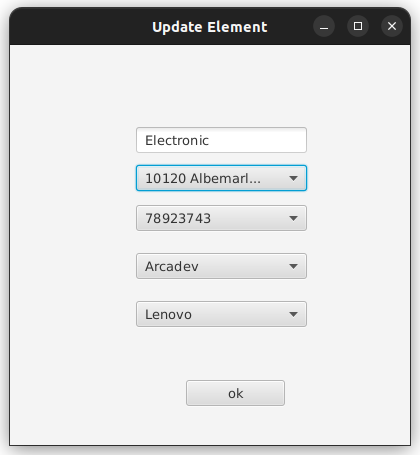
}

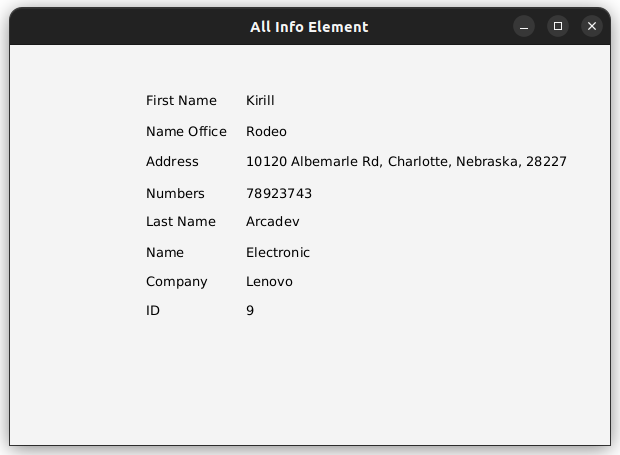
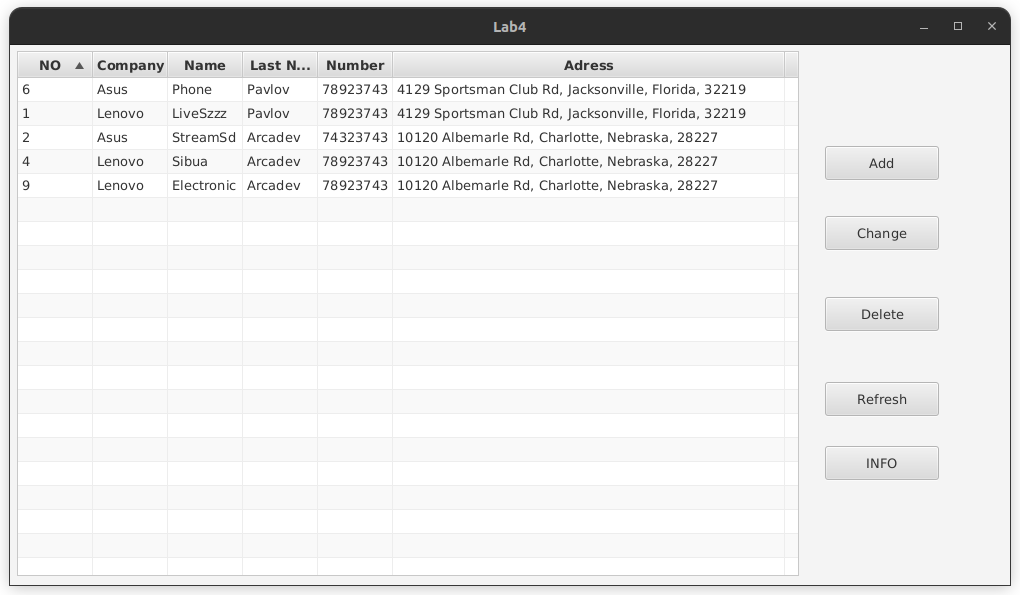
}

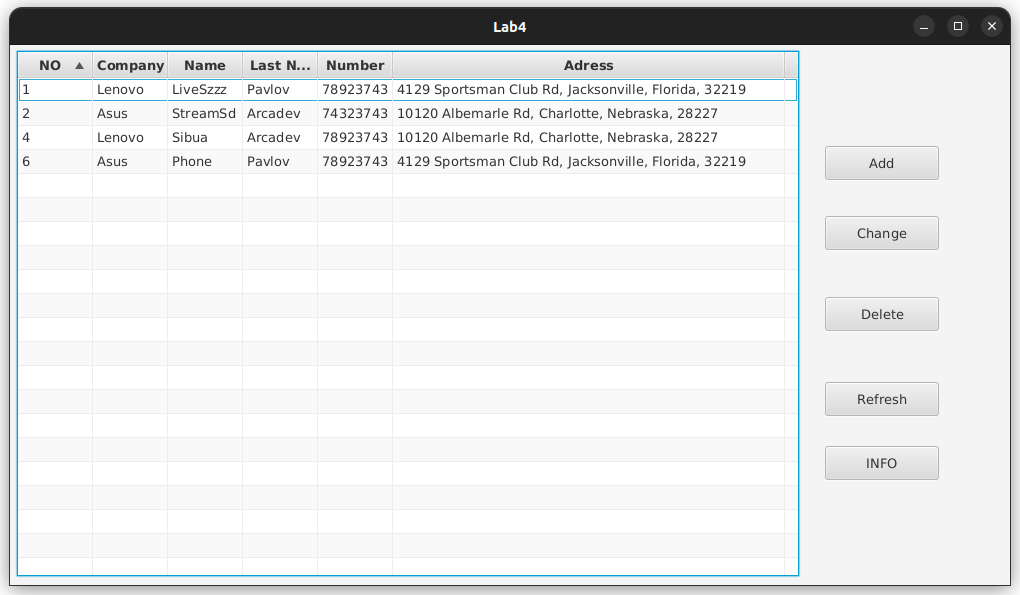












**Вывод:** приобрел практические навыки разработки многооконных приложений на JavaFX для работы с базами данных.