import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.stage.Stage;

public class Main extends Application {

@Override

public void start(Stage primaryStage) throws Exception {

Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));

primaryStage.setTitle(";)");

primaryStage.setScene(new Scene(root, 855, 535));

// primaryStage.setResizable(false);

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

}

Class Controller:

import java.net.URL;

import java.sql.\*;

import java.util.ResourceBundle;

import javafx.application.Platform;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.\*;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.web.WebEngine;

import javafx.scene.web.WebHistory;

import javafx.scene.web.WebView;

public class Controller implements Initializable {

//Button

@FXML

public Button backButton;// публічні змінні типу Button, які

@FXML

public Button forwardButton;

@FXML

private Button addButton;

@FXML

private Button refreshButton;

@FXML

private Button deleteButton;

@FXML

public Button goButton;

//Button

// TextField

@FXML

public TextField searchField;

@FXML

public TextField nameField;

@FXML

public TextField linkField;

// TextField

// TableView

@FXML

public TableColumn<Data, String> tableViewColumId;

@FXML

public TableColumn<Data, String> tableViewColumName;

@FXML

public TableColumn<Data, String> tableViewColumLink;

@FXML

public TableView<Data> tableView;

@FXML

public ObservableList<Data> data;

@FXML

public DataBaseConnector dc;

Data dta = new Data();

//TableView

//Browser

public String http = " http://";

@FXML

public TextField addresBar;

@FXML

public WebView webView;

@FXML

public WebEngine engine;

@FXML

public String adrsLink;

//Browser /////////////////////////////////////////////\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/////////////////////////////////////////////////

// methods

// Главный метод инициализации

@Override

public void initialize(URL location, ResourceBundle resources) {

//Brawser

engine = webView.getEngine();//получаем объект WebEngine из WebView используя метод getEngine()

engine.load(http + "www.google.com");

//Brawser

dc = (new DataBaseConnector());

//Tooltip

addButton.setTooltip(toolTipe("This button add elements in table"));

deleteButton.setTooltip(toolTipe("This button delete elements from table"));

refreshButton.setTooltip(toolTipe("Press this button for load data in table"));

goButton.setTooltip(toolTipe("Go to you address"));

tableView.setTooltip(toolTipe("Here located youre data"));

backButton.setTooltip(toolTipe("Back on previously page"));

forwardButton.setTooltip(toolTipe("Back on next page"));

addresBar.setTooltip(toolTipe("Input here you address"));

tableView.setTooltip(toolTipe("Here located youre data"));

addButton.setDisable(true);

deleteButton.setDisable(true);

//Tooltip

//Listiners

addButton.setOnAction(event -> {

addAction();

loadDataFromDatabaseToTableView();

});

deleteButton.setOnAction(event -> {

deleteAction();

});

refreshButton.setOnAction(event -> {

addButton.setDisable(false);

deleteButton.setDisable(false);

refreshButton.setDisable(true);

loadDataFromDatabaseToTableView();

});

goButton.setOnAction(event -> {

go();

});

tableView.setOnMouseClicked(event -> {

getAddres();

});

backButton.setOnAction(event -> {

goBack();

});

forwardButton.setOnAction(event -> {

goForward();

});

//Listiners

}

public void deleteAction() {

Data d = tableView.getSelectionModel().getSelectedItem();

if (d == null) {

warningDialogs("No object selected", "Choose object for deleting ");

} else {

try {

deleteFromDb(d.getId());

///deletefromTable

int selectR = tableView.getSelectionModel().getSelectedIndex();

tableView.getItems().remove(selectR);

addresBar.clear();

///deletefromTable

} catch (Exception e) {

e.printStackTrace();

}

}

}

public void deleteFromDb(String id) {

try {

String sql = "DELETE FROM mydatabase WHERE idmydatabase = " + id + "";

sqlExecute(sql);

} catch (Exception e) {

} }

public void addAction() {

if (nameField.getLength() != 0 && linkField.getLength() != 0) {

DataBaseConnector dataBaseConnector = new DataBaseConnector();

dataBaseConnector.addElementToDB(nameField.getText(), linkField.getText());

nameField.clear();

linkField.clear();

} else {

warningDialogs("Empty textfield", "Information in textfield is empty\nPlease, fill textfield");

}

}

// Метод загрузки инфи в таблицу из БД

public void loadDataFromDatabaseToTableView() {

try {

Connection conn = dc.getDbConnection();

data = FXCollections.observableArrayList();

ResultSet rs = conn.createStatement().executeQuery("SELECT \* FROM mydatabase;");

while (rs.next()) {

data.add(new Data(rs.getString("Name"), rs.getString("Link"), rs.getString("idmydatabase")));

}

tableViewColumName.setCellValueFactory(new PropertyValueFactory<>("name"));

tableViewColumLink.setCellValueFactory(new PropertyValueFactory<>("link"));

tableView.setItems(null);

tableView.setItems(data);

} catch (SQLException e) {

e.printStackTrace();

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

/// Go Button

public void go() {

if (addresBar.getLength() != 0) {

adrsLink = addresBar.getText().toString();

engine.load(http + adrsLink);

} else {

warningDialogs("Empty Address Bar", "Input address");

}

}

public void sqlExecute(String sql) {

try {

Connection connection = dc.getDbConnection();

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.executeUpdate();

preparedStatement.close();

connection.close();

} catch (SQLException e) {

e.printStackTrace();

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

public void getAddres() {

Data links = tableView.getSelectionModel().getSelectedItem();

String lin = links.getLink();

addresBar.setText(lin);

}

public void warningDialogs(String title, String description) {

Alert alert = new Alert(Alert.AlertType.WARNING);

alert.setTitle(title);

alert.setContentText(description);

alert.showAndWait();

}

public void goBack() {

final WebHistory history = engine.getHistory();

ObservableList<WebHistory.Entry> entryList = history.getEntries();

int currentIndex = history.getCurrentIndex();

Platform.runLater(() ->

{

history.go(entryList.size() > 1 && currentIndex > 0 ? -1 : 0);

});

}

public void goForward() {

final WebHistory history = engine.getHistory();

ObservableList<WebHistory.Entry> entryList = history.getEntries();

int currentIndex = history.getCurrentIndex();

Platform.runLater(() ->

{

history.go(entryList.size() > 1 && currentIndex < entryList.size() - 1 ? 1 : 0);

});

}

public Tooltip toolTipe(String text) {

Tooltip tooltip = new Tooltip(text);

return tooltip;

}

}

Class CONST:

public class Const {

public static final String MYTABLE ="mydatabase";

public static final String MYTABLE\_ID ="idmydatabase";

public static final String MYTABLE\_NAME ="Name";

public static final String MYTABLE\_LINK ="Link";

}

Class DataBaseConnector:

import java.sql.\*;

public class DataBaseConnector {

public Connection dbConnection;

/// Подключение к БД

public Connection getDbConnection()

throws ClassNotFoundException, SQLException {

String dbName = "root";

String dbPass = "mwg3936";

String connectionUrl = "jdbc:mysql://localhost:3306/mydatabase?serverTimezone=UTC&useSSL=false";

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

dbConnection = DriverManager.getConnection(connectionUrl, dbName, dbPass);

return dbConnection;

}

/// метод добавления инф. в БД

public void addElementToDB(String name, String link) {

String sql = "INSERT INTO " + Const.MYTABLE + "(" + Const.MYTABLE\_NAME + "," + Const.MYTABLE\_LINK + ")" + "VALUES (?,?)";//SQL

try {

PreparedStatement preparedStatement = getDbConnection().prepareStatement(sql);

preparedStatement.setString(1, name);

preparedStatement.setString(2, link);

preparedStatement.execute();

} catch (SQLException e) {

e.printStackTrace();

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

}

Class Data:

import javafx.beans.property.SimpleStringProperty;

public class Data {

private SimpleStringProperty name;

private SimpleStringProperty link;

private SimpleStringProperty id;

// Конструктор

public Data(String name, String link, String id) {

this.name = new SimpleStringProperty(name);

this.link = new SimpleStringProperty(link);

this.id = new SimpleStringProperty(id);

}

// Конструктор

public Data() {

}

public String getName() {

return name.get();

}

public SimpleStringProperty nameProperty() {

return name;

}

public void setName(String name) {

this.name.set(name);

}

public String getLink() {

return link.get();

}

public SimpleStringProperty linkProperty() {

return link; }

public void setLink(String link) {

this.link.set(link);

}

public String getId() {

return id.get();

}

public SimpleStringProperty idProperty() {

return id;

}

public void setId(String id) {

this.id.set(id);

}

}