Министерство образования Республики Беларусь

Учреждение образования

“БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ”

Факультет компьютерного проектирования

Дисциплина: СиТАиРИС

**ЛАБОРАТОРНАЯ РАБОТА № 1**

**Объектно-ориентированный анализ и моделирование,**

**разработка статических моделей**

Выполнил:

Студент гр. 814301

Аблажевич В.В.

Проверил:

Лыщик А.П.

Минск 2020

1. **Задание:** а). Проектирование моделей в стандарте UML.

б). Программная реализация на JAVA в отдельном java-пакете.

в). Использование и полученного пакета для разработки программной системы.

г). Придумать для данных схем (моделей) описание (легенду) задачи реального мира.

1. **Решение:**

Программа реализации интенет-магазина в виде веб приложения с использованием баз данных.

**MainServer (в этом файле мы производим подключение клиента к серверу):**

package server.sample;

import javafx.application.Application;

import javafx.application.Platform;

import javafx.event.EventHandler;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.stage.Stage;

import javafx.stage.WindowEvent;

public class MainServer extends Application {

@Override

public void start(Stage primaryStage) throws Exception {

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

primaryStage.setTitle("Сервер");

primaryStage.setScene(new Scene(root, 500, 500));

primaryStage.setOnCloseRequest(new EventHandler<WindowEvent>() {

@Override

public void handle(WindowEvent t) {

Platform.exit();

System.exit(0);

}

});

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

}

**ServerController(в этом файле мы запускаем работу сервера):**

package server.sample;

import javafx.fxml.FXML;

import javafx.scene.control.Button;

import javafx.scene.control.TextField;

import server.Database.Database;

import server.server.ServerInstance;

public class ServerController {

@FXML

private TextField database;

@FXML

private TextField login;

@FXML

private TextField psswrd;

@FXML

private TextField prt;

@FXML

private Button srvOn;

@FXML

private Button srvOff;

public void connectDB() {

Database.connect(database.getText(), login.getText(), psswrd.getText(), prt.getText());

}

public void startServer() {

new Thread(ServerInstance.INSTANCE.getInstance()).start();

srvOn.setVisible(false);

srvOff.setVisible(true);

}

public void stopServer() {

ServerInstance.INSTANCE.getInstance().stop();

srvOn.setVisible(true);

srvOff.setVisible(false);

}

}

**MainClient(в этом файле мы считываем и запоминаем данные для подключения к серверу):**

package server.sample;

import javafx.fxml.FXML;

import javafx.scene.control.Button;

import javafx.scene.control.TextField;

import server.Database.Database;

import server.server.ServerInstance;

public class ServerController {

@FXML

private TextField database;

@FXML

private TextField login;

@FXML

private TextField psswrd;

@FXML

private TextField prt;

@FXML

private Button srvOn;

@FXML

private Button srvOff;

public void connectDB() {

Database.connect(database.getText(), login.getText(), psswrd.getText(), prt.getText());

}

public void startServer() {

new Thread(ServerInstance.INSTANCE.getInstance()).start();

srvOn.setVisible(false);

srvOff.setVisible(true);

}

public void stopServer() {

ServerInstance.INSTANCE.getInstance().stop();

srvOn.setVisible(true);

srvOff.setVisible(false);

}

}

**Client(в этом файле мы подключаемся к серверу):**

package client.sample;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.\*;

import java.util.ArrayList;

public class Client {

private Socket clientSocket;

private ObjectInputStream clientInpStream;

private ObjectOutputStream clientOutpStream;

public void connect() {

try {

clientSocket = new Socket("localhost", 2525);

clientOutpStream = new ObjectOutputStream(clientSocket.getOutputStream());

clientInpStream = new ObjectInputStream(clientSocket.getInputStream());

} catch (Exception e) {

e.printStackTrace();

}

}

public void send(Object obj) {

try {

clientOutpStream.writeObject(obj);

} catch (Exception e) {

e.printStackTrace();

}

}

public boolean receiveResultBool() {

boolean result = true;

try {

result = (boolean) clientInpStream.readObject();

} catch (Exception e) {

e.printStackTrace();

}

return result;

}

public ArrayList<String> receiveResultList() {

ArrayList<String> list = null;

try {

list = (ArrayList<String>)clientInpStream.readObject();

} catch (Exception e) {

}

return list;

}

public String receiveResultString() {

String result = null;

try {

result = (String) clientInpStream.readObject();

} catch (Exception e) {

}

return result;

}

}

**Catalog(в этом файле мы заполняем каталог):**

package client.entityClass;

import java.util.Objects;

public class Catalog {

private int id\_catalog;

private String number;

private String clothes;

private String material;

private String brand;

private String size;

private double cost;

private String status;

public Catalog() {

}

public Catalog(int id\_catalog, String number, String clothes, String material, String brand, String size, double cost, String status) {

this.id\_catalog = id\_catalog;

this.number = number;

this.clothes = clothes;

this.material = material;

this.brand = brand;

this.size = size;

this.cost = cost;

this.status = status;

}

public int getId\_catalog() {

return id\_catalog;

}

public void setId\_catalog(int id\_catalog) {

this.id\_catalog = id\_catalog;

}

public String getNumber() {

return number;

}

public void setNumber(String number) {

this.number = number;

}

public String getClothes() {

return clothes;

}

public void setClothes(String clothes) {

this.clothes = clothes;

}

public String getMaterial() {

return material;

}

public void setMaterial(String material) {

this.material = material;

}

public String getBrand() {

return brand;

}

public void setBrand(String brand) {

this.brand = brand;

}

public String getSize() {

return size;

}

public void setSize(String size) {

this.size = size;

}

public double getCost() {

return cost;

}

public void setCost(double cost) {

this.cost = cost;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

Catalog catalog = (Catalog) o;

return id\_catalog == catalog.id\_catalog &&

Double.compare(catalog.cost, cost) == 0 &&

Objects.equals(number, catalog.number) &&

Objects.equals(clothes, catalog.clothes) &&

Objects.equals(material, catalog.material) &&

Objects.equals(brand, catalog.brand) &&

Objects.equals(size, catalog.size) &&

Objects.equals(status, catalog.status);

}

@Override

public int hashCode() {

return Objects.hash(id\_catalog, number, clothes, material, brand, size, cost, status);

}

@Override

public String toString() {

return "Catalog{" +

"id\_catalog=" + id\_catalog +

", number='" + number + '\'' +

", clothes='" + clothes + '\'' +

", material='" + material + '\'' +

", brand='" + brand + '\'' +

", size='" + size + '\'' +

", cost=" + cost +

", status='" + status + '\'' +

'}';

}

}

1. **Результат программы:**



