Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Лабораторная работа №6 по дисциплине  
«Технология программирования» на тему

**Клиент-Сервер**

Факультет: АВТФ Преподаватель:

Группа: АВТ-810 Михайленко Д. А.

Студенты: Степаненко Владислав

Новосибирск

2020

#### Практические задания

# Цель работы:

1. Изучить особенности реализации сетевых приложений в Java.

2. Доработать программу, созданную в лабораторной работе № 5:

**Вариант 7**

Реализовать возможность скопировать и добавить в симуляцию все объекты одного из подключенных клиентов.

# Задание

1) создать отдельное консольное приложение, которое будет играть роль TCP-сервера. Клиентом будет приложение, которое было создано в предыдущих работах;

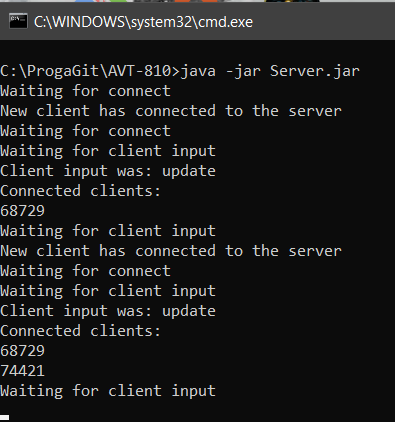
2) TCP-сервер должен ожидать подключения клиентов и выдавать вновь подключенному клиенту список уже подключенных. Необходимо также внести изменения в интерфейс клиента, так чтобы в панели управления отображался список всех подключенных к серверу клиентов. При отсоединении клиентов или при подключении новых список должен обновляться;

3) запрограммировать специальное взаимодействие по TCP с другими клиентами через сервер по варианту;

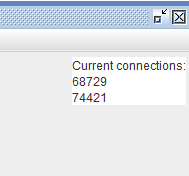
Создано отдельное консольное приложение Server. Создан вектор хранящий все соединения в виде отдельных потоков.

При присоединение нового клиента или отключения старого автоматически обновляется список клиентов и рассылается по всем клиентам.

Вся информация сервера выводится в консоль



Отображение информации о подключенных клиентах



Копирование объектов из другого клиента осуществляется при помощи консоли клиента созданной ранее. (формат команды – get id), где id – номер клиента в списке.

Связи между клиентом и сервером осуществляется через потоки ввода и вывода ObjectStream.

Сервер воспринимает команды:

gettome id – получить объекты у номера id и вернуть текущему клиенту

setto id – передать объекты номеру id

update – обновить информацию о подключениях у всех клиентов

**Код сервера:**

import objects.\*;  
import java.io.\*;  
import java.net.Socket;  
import java.util.Vector;  
  
  
public class Server extends Thread  
{  
 public int id;  
 private Socket socket;  
 public ObjectInputStream objectInputStream;  
 public ObjectOutputStream objectOutputStream;  
  
 public Server(Socket socket) throws IOException {  
 this.socket = socket;  
 this.id = (int) (Math.*random*() \* 100000);  
 objectOutputStream= new ObjectOutputStream(socket.getOutputStream());  
 objectInputStream = new ObjectInputStream(socket.getInputStream());  
 start();  
 }  
  
 @Override  
 public void run() {  
 super.run();  
 try {  
 while(true)  
 {  
 System.*out*.println("Waiting for client input");  
 String message = (String) objectInputStream.readObject();  
 System.*out*.println("Client input was: " + message);  
 if (message.equals("update"))  
 {  
 Main.*updateClientList*();  
 }  
  
 String[] res = message.split(" ");  
 if (res[0].equals("gettome"))  
 {  
 int destId = Integer.*parseInt*(res[1]);  
 for (Server item: Main.*serverList*)  
 if (item.id == destId)  
 {  
 item.objectOutputStream.writeObject("getto " + this.id);  
 }  
 }  
 if (res[0].equals("setto"))  
 {  
 int destId = Integer.*parseInt*(res[1]);  
 Vector<House> data = (Vector<House>) objectInputStream.readObject();  
 for (Server item: Main.*serverList*)  
 if (item.id == destId)  
 {  
 item.objectOutputStream.writeObject("set");  
 item.objectOutputStream.flush();  
 item.objectOutputStream.writeObject(data);  
 }  
 }  
  
 }  
 }  
 catch (IOException | ClassNotFoundException e)  
 {  
 //e.printStackTrace();  
 delete();  
 System.*out*.println("Disconnected");  
 }  
 }  
 private void delete() {  
 Main.*serverList*.remove(this);  
 try {  
 this.objectInputStream.close();  
 this.objectOutputStream.close();  
 this.socket.close();  
 Main.*updateClientList*();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

import java.io.IOException;  
import java.net.ServerSocket;  
import java.net.Socket;  
import java.util.ArrayList;  
import java.util.LinkedList;  
  
class Main  
{  
 public static final int PORT =48655;  
 public static LinkedList<Server> serverList = new LinkedList<>();  
  
 public static void main(String[] args) throws IOException  
 {  
 ServerSocket serverSocket = new ServerSocket(PORT);  
 try {  
 while (true) {  
 System.*out*.println("Waiting for connect");  
 Socket socket = serverSocket.accept();  
 System.*out*.println("New client has connected to the server");  
 *serverList*.add(new Server(socket));  
  
 }  
 } finally {  
 serverSocket.close();  
 }  
 }  
 static void updateClientList() {  
 ArrayList<Integer> idList = new ArrayList<>(30);  
 for (Server item: *serverList*)  
 idList.add(item.id);  
 System.out.println("Connected clients:");  
 for (Server item: serverList)  
 {  
 System.out.println(item.id);  
 try {  
 item.objectOutputStream.writeObject("update");  
 item.objectOutputStream.flush();  
 item.objectOutputStream.writeObject(idList);  
 item.objectOutputStream.flush();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}