Министерство образования Республики Беларусь

Министерство науки и высшего образования Российской Федерации

МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»»

**Лабораторная работа № 4. Сетевое программирование**

**с сокетами и каналами**

Выполнил: студент гр.ИСИТ-191

Харкевич А. В.

Проверил: Денисевич Д. А.

Могилев, 2021

**Цель работы:** приобретение навыков проектирования и разработки приложений в архитектуре клиент-сервер.

**Задание:** Сервер рассылает сообщения только тем клиентам, которые в настоящий момент находятся в on-line.

**Проект Lab4Server:**

**Main.java**

package com.company;

import javax.imageio.IIOException;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

public class Main {

private static int port = 8081;

private static ServerSocket serverSocket;

private static Socket socket;

public static void main(String[] args) {

try {

serverSocket = new ServerSocket(port);

} catch (IOException e) {

e.printStackTrace();

}

while (true) {

try {

socket = serverSocket.accept();

} catch (IOException e) {

System.out.println("error: " + e);

}

new EchoThread(socket).start();

}

}

}

**EchoThread.java**

package com.company;

import java.io.\*;

import java.net.Socket;

public class EchoThread extends Thread {

private Socket socket;

private BufferedReader in;

private PrintWriter out;

private String username;

private Status status;

public EchoThread(Socket clientSocket) {

this.socket = clientSocket;

try {

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

out = new PrintWriter(new OutputStreamWriter(socket.getOutputStream()), true);

} catch (IOException e) {

return;

}

status = new Status(clientSocket);

}

public void run() {

try {

username = in.readLine();

out.println(username + " в чате");

} catch (IOException e) {

e.printStackTrace();

}

new Thread(new Status(socket)).start();

while (true) {

try {

String text = in.readLine();

if (text.equals("quit")) {

socket.close();

return;

}

System.out.println("Echoing: " + text);

out.println(text);

} catch (IOException e) {

out.println(username + " вышел");

System.out.println("disconnected");

return;

}

}

}

}

**Status.java**

package com.company;

import java.io.IOException;

import java.io.OutputStreamWriter;

import java.io.PrintWriter;

import java.net.Socket;

public class Status extends Thread{

private Socket socket;

private PrintWriter out;

public Status(Socket clientSocket) {

this.socket = clientSocket;

try {

out = new PrintWriter(new OutputStreamWriter(socket.getOutputStream()), true);

} catch (IOException e) {

e.printStackTrace();

}

}

public void run(){

while(true){

try {

Thread.sleep(5000);

} catch (InterruptedException e) {

e.printStackTrace();

}

out.println("Your status is online");

}

}

}

**Проект lab4Client:**

**Main.java**

package com.company;

import java.io.\*;

import java.net.Socket;

public class Main {

private static Socket socket;

private static BufferedReader reader;

private static BufferedReader in;

private static PrintWriter out;

public static void main(String[] args) throws IOException {

socket = new Socket("localhost", 8081);

reader = new BufferedReader(new InputStreamReader(System.in));

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

out = new PrintWriter(new OutputStreamWriter(socket.getOutputStream()), true);

System.out.println("Соединение установлено:");

System.out.println("Введите имя: ");

String username = reader.readLine();

out.write(username + "\n");

out.flush();

String welcomeUser = in.readLine();

System.out.println(welcomeUser);

new Thread(new Reader(socket)).start();

while(true){

System.out.println("->");

String word = reader.readLine();

out.write(word + "\n");

out.flush();

}

}

}

**Reader.java**

package com.company;

import java.io.\*;

import java.net.Socket;

public class Reader extends Thread{

private Socket socket;

private BufferedReader in;

public Reader(Socket clientSocket) {

this.socket = clientSocket;

try {

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

} catch (IOException e) {

e.printStackTrace();

}

}

public void run(){

while(true){

try {

String serverWord = in.readLine();

System.out.println(serverWord);

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

