Національний Технічний Університет України

Київський Політехнічний Інститут імені Ігоря Сікорського

Інститут Прикладного Системного Аналізу

Кафедра Системного Проектування

Паралельні обчислення

Лабораторна робота №5

Виконав:

Дєрюгін Є.О.

Група: ДА-81

Перевірив: Яременко В.С.

Київ – 2021

ЗМІСТ

[Мета роботи 3](#_Toc70287781)

[Завдання 3](#_Toc70287782)

[Лістинг програми мовою Java 4](#_Toc70287783)

[Dealer.java – сервер 4](#_Toc70287784)

[Client.java 7](#_Toc70287785)

[Deck.java – колода, рука гравця та дилера 8](#_Toc70287786)

[Card.java – карта 10](#_Toc70287787)

[Suit.java – масть 11](#_Toc70287788)

[Value.java – значення 11](#_Toc70287789)

[Результати роботи програми 12](#_Toc70287790)

[Опис роботи програми 13](#_Toc70287791)

[Висновки 14](#_Toc70287792)

Мета роботи

Дослідження мережевих можливостей розподіленних систем та реалізація клієнт-серверного додатку.

Завдання

Розробити клієнт-серверний додаток, використовуючи Java RMI, Sockets або аналогічну технологію.

**Варіант 7**

Чат. Сервер розсилає всім клієнтам інформацію про клієнтів, які увійшли в чат і покинули його.

Лістинг програми мовою Java

Server.java – сервер

import java.io.\*;

import java.net.\*;

public class Server {

    // необхідні змінні для роботи серверу

    private static ServerSocket server;

    private static BufferedReader in1;

    private static BufferedWriter out1;

    private static BufferedReader in2;

    private static BufferedWriter out2;

    public static void main(String[] args){

        try{

            try{

                // прив'язую серверний сокет до порта

                server = new ServerSocket(7000);

                System.out.println("Server is started!");

                // прослуховую під'єднання першого клієнта до сокета

                Socket client1 = server.accept();

                System.out.println("Client 1 connected!");

                // прослуховую під'єднання другого клієнта до сокета

                Socket client2 = server.accept();

                System.out.println("Client 2 connected!");

                try{

                    // створюю необхідні змінні для вводу/виводу інформації на клієнти через InputStream та OutputStream

                    in1 = new BufferedReader(new InputStreamReader(client1.getInputStream()));

                    out1 = new BufferedWriter(new OutputStreamWriter(client1.getOutputStream()));

                    in2 = new BufferedReader(new InputStreamReader(client2.getInputStream()));

                    out2 = new BufferedWriter(new OutputStreamWriter(client2.getOutputStream()));

                    out1.write("Welcome to the Chat! \n");

                    out1.flush();

                    out2.write("Welcome to the Chat! \n");

                    out2.flush();

                    out1.write("Please, write your name and age:) \n");

                    out1.flush();

                    out2.write("Please, write your name and age:) \n");

                    out2.flush();

                    // запитую у першого клієнта ім'я та вік

                    String cliWord1 = in1.readLine();

                    System.out.println("Client 1: " + cliWord1);

                    // запитую у другого клієнта ім'я та вік

                    String cliWord2 = in2.readLine();

                    System.out.println("Client 2: " + cliWord2);

                    // вивожу другому клієнту ім'я та вік першого, якщо отриманий рядок не пустий

                    if (cliWord1 != null){

                        out2.write("Your collocutor on client 1 is " + cliWord1 + " \n");

                    }

                    // якщо отриманий рядок пустий, то вивожу другому клієнту інформацію

                    // що перший клієнт нічого не написав

                    else {

                        out2.write("Your collocutor on client 1 wrote nothing \n");

                    }

                    out2.flush();

                    // вивожу першому клієнту ім'я та вік другого, якщо отриманий рядок не пустий

                    if (cliWord2 != null){

                        out1.write("Your collocutor on client 2 is " + cliWord2 + " \n");

                    }

                    // якщо отриманий рядок пустий, то вивожу першому клієнту інформацію

                    // що другий клієнт нічого не написав

                    else {

                        out1.write("Your collocutor on client 2 wrote nothing \n");

                    }

                    out1.flush();

                } finally {

                    // закриваю клієнт, Input/OutputStreams

                    client1.close();

                    client2.close();

                    in1.close();

                    out1.close();

                    in2.close();

                    out2.close();

                }

            } finally{

                // закриваю сервер

                System.out.println("Server is closed!");

                server.close();

            }

        } catch (IOException e) {

            System.err.println(e);

        }

    }

}

Client1.java

import java.io.\*;

import java.net.\*;

public class Client1 {

    // необхідні змінні для роботи клієнту

    private static Socket client1;

    private static BufferedReader reader;

    private static BufferedReader in;

    private static BufferedWriter out;

    public static void main(String[] args){

        try{

            try{

                // спроба підключення клієнта до сервера

                client1 = new Socket("localhost", 7000);

                // створюю необхідні Input/Output Streams для передачі інформації між клієнтом та сервером

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

                in = new BufferedReader(new InputStreamReader(client1.getInputStream()));

                out = new BufferedWriter(new OutputStreamWriter(client1.getOutputStream()));

                // зчитую інформацію з сервера

                String serverWord = in.readLine(); // welcome to the chat

                System.out.println(serverWord);

                serverWord = in.readLine(); // name age

                System.out.println(serverWord);

                // зчитую інформацію на з консолі для передачі на сервер

                String clientWord = reader.readLine();

                // передаю інформацію на сервер

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

                out.flush();

                serverWord = in.readLine(); // name age of cli 2

                System.out.println(serverWord);

            } finally {

                // закриваю клієнт та стріми

                System.out.println("Client 1 was closed");

                client1.close();

                in.close();

                out.close();

                reader.close();

            }

        } catch (IOException e){

            System.err.println(e);

        }

    }

}

Client2.java

import java.io.\*;

import java.net.\*;

public class Client1 {

    // необхідні змінні для роботи клієнту

    private static Socket client1;

    private static BufferedReader reader;

    private static BufferedReader in;

    private static BufferedWriter out;

    public static void main(String[] args){

        try{

            try{

                // спроба підключення клієнта до сервера

                client1 = new Socket("localhost", 7000);

                // створюю необхідні Input/Output Streams для передачі інформації між клієнтом та сервером

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

                in = new BufferedReader(new InputStreamReader(client1.getInputStream()));

                out = new BufferedWriter(new OutputStreamWriter(client1.getOutputStream()));

                // зчитую інформацію з сервера

                String serverWord = in.readLine(); // welcome to the chat

                System.out.println(serverWord);

                serverWord = in.readLine(); // name age

                System.out.println(serverWord);

                // зчитую інформацію на з консолі для передачі на сервер

                String clientWord = reader.readLine();

                // передаю інформацію на сервер

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

                out.flush();

                serverWord = in.readLine(); // name age of cli 2

                System.out.println(serverWord);

            } finally {

                // закриваю клієнт та стріми

                System.out.println("Client 1 was closed");

                client1.close();

                in.close();

                out.close();

                reader.close();

            }

        } catch (IOException e){

            System.err.println(e);

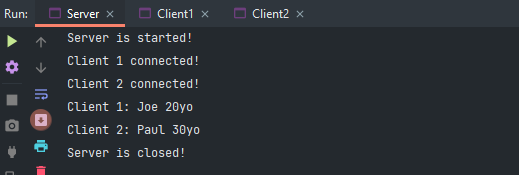
        }

    }

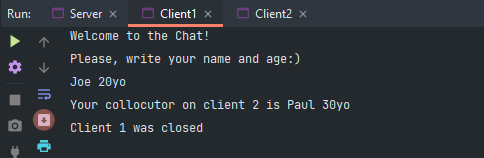
}

Результати роботи програми

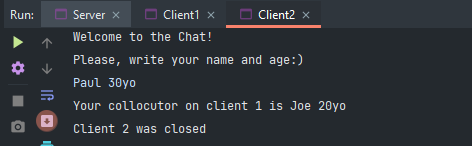
Консоль сервера:



Консоль першого клієнта:



Консоль другого клієнта:



Опис роботи програми

Спочатку запускаємо програму серверу – Server.java, далі по черзі запускаємо Client1.java та Client2.java. Після запуску клієнтів, вони під’єднуються до сервера, про що в консолі сервера виводиться «Client 1 connected» або «Client 2 connected». Після успішного під’єднання до сервера клієнтам виводиться вітальне повідомлення та прохання ввести ім’я та вік клієнта, що передається з клієнтів на сервер. На сервері введені дані відображаються у консолі для перевірки правильності виконання програми.

Далі на сервері відбувається перевірка введених даних на те, щоб вони не були пусті, і якщо вони не пусті, то клієнтам виводиться ім’я та вік їх співрозмовників на інших клієнтах. Якщо введені дані будуть пустими, на клієнти буде виводитися, що співрозмовник нічого не написав.

Висновки

Під час виконання лабораторної роботи було створено 3 програми (сервер та два клієнта) за умовами варіанту, а саме чат, який виводить клієнтам інформацію про інших клієнтів. Програми підключаються один до одного за допомогою сокетів. Спілкуються програми через Input/Output стріми.

Під час виконання лабораторної роботи я ознайомився із концепцією сокетів та особливостями їх використання. Проблем під час виконання лабораторної роботи не виникало.