TP 2: Socket TCP/IP et Threads

Application 1: Serveur de Calcul MultiClients

Ecrire une application Client/Serveur TCP/IP MultiClients basée sur les sockets permettant au client d’envoyer deux entiers et un opérateur au Serveur. Le serveur fait alors le calcul nécessaire et renvoie le résultat au client.

NB : Chaque client peut envoyer plusieurs demandes de calcul jusqu’à l’envoi de la valeur 0 comme première opérande.

/\*

Classe Serveur

\*/

**import** java.io.IOException;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** Serveur {

**public** **static** **void** main(String []args)

{**int** i=0;

**try** {

ServerSocket ss=**new** ServerSocket(2020);

**while**(**true**)

{Socket s=ss.accept();

i++;

ThreadCalcul c=**new** ThreadCalcul(s);

c.start();

System.***out***.println("Client" +i+ "connecté au Thread"+Thread.*currentThread*().getName());

}

} **catch** (IOException e) {

e.printStackTrace();

}

}

/\*

Classe ThreadCalcul

\*/

**import** java.io.\*;

**import** java.net.Socket;

**public** **class** ThreadCalcul **extends** Thread{

Socket s;

BufferedReader in;

PrintWriter out;

**public** ThreadCalcul(Socket s) {

**this**.s=s;

**try** {

in= **new** BufferedReader (**new** InputStreamReader(s.getInputStream()));

out =**new** PrintWriter(s.getOutputStream());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**public** **void** run() {

**while** (**true**) {

**try** {

String N1 = in.readLine();

**if**(N1.equals("0"))

{s.close();

**break**;}

String N2 = in.readLine();

**int** n1=Integer.*parseInt*(N1);

**int** n2=Integer.*parseInt*(N2);

String op = in.readLine();

**switch** (op){

**case** "+": out.println(N1+" + "+N2+" = "+(n1+n2));**break**;

**case** "-": out.println(N1+" - "+N2+" = "+(n1-n2));**break**;

**case** "\*": out.println(N1+" \* "+N2+" = "+n1\*n2);**break**;

**case** "/": out.println(N1+" / "+N2+" = "+n1/n2);**break**;

**default**: out.println("opératur non valide!");

}

System.***out***.println("opération effectuée au "+Thread.*currentThread*().getName());

out.flush();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}}

}

}

/\*

Classe Client

\*/

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.PrintWriter;

**import** java.net.InetAddress;

**import** java.net.Socket;

**import** java.util.Scanner;

**public** **class** Client {

**public** **static** **void** main(String []args)

{

**try** {

Socket s=**new** Socket(InetAddress.*getLocalHost*(),2020);

BufferedReader in= **new** BufferedReader (**new** InputStreamReader(s.getInputStream()));

PrintWriter out =**new** PrintWriter(s.getOutputStream());

Scanner sc=**new** Scanner(System.***in***);

**while**(**true**) {

System.***out***.println("donner N1");

**int** N1=sc.nextInt();

out.println(N1);

out.flush();

**if**(N1==0) {s.close(); **break**;}

System.***out***.println("donner N2");

**int** N2=sc.nextInt();

sc=**new** Scanner(System.***in***);

System.***out***.println("donner op");

String op=sc.nextLine();

out.println(N2);

out.println(op);

out.flush();

System.***out***.println(in.readLine());

}

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

Application 2 : Serveur MultiServices

Soit la classe Etudiant ayant comme attributs : nom (String), prenom (String), note1 (int), note2 (int). Notre objectif est d’implémenter une application Client/Serveur basée sur les Sockets TCP permettant au serveur de fournir deux services:

-Le premier permet de recevoir deux chaînes et deux entiers pour en créer une instance Etudiant et l’envoyer au client.

-Le deuxième permet de recevoir une instance de type Etudiant et de renvoyer sa moyenne au client.

Implémenter la classe Etudiant, le serveur et un client pour chaque service.