UF3. [PAC01] SOLUCIÓN

Actividades

Parte práctica

1. Escribe un programa que cuente el número de conexiones que vaya recibiendo. Este programa dispondrá de un socket stream servidor. Cada vez que un socket cliente se conecte, este le enviará un mensaje con el número de clientes conectados hasta ahora. Así pues, el primer cliente que se conecte recibirá un 1, el segundo un 2, el tercero un 3, etc.

SERVIDOR

```
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
/*
 * CLASE SERVIDOR___
 *
 * Esta clase es la encargada de recibir las conexiones
 * de los clientes y devolverles un entero indicando
 * el numero de cliente.
 *
 */

public class Server {
     public static void main(String[] args) throws
InterruptedException {
          Server myServer= new Server(); //Creamos el objeto
servidor
          myServer.run(); //Lo ejecutamos.
     }
```

```
	//Definimos todos los objetos que nos harán falta para hacer funcionar nuestro servidor
	static int clientCounter; //Entero para contar el número de clientes
	static ServerSocket serverSock; //Objeto que recibira las peticiones de los clientes
	static Socket clientSock;
	static DataOutputStream dataOutputStram; //Flujo de salida para enviar datos al cliente

	//Inicializamos las variables del servidor.
	public Server(){
		try {
			serverSock = new ServerSocket(6060); //Establecemos como puerto el 6060
		} catch (IOException e) {
			e.printStackTrace();
		}
		clientCounter=0;
		clientSock=null;
	}

	//Funcion encargada de la ejecución.
	public void run(){
		System.out.println("The server is listening on port 6060");
		while(true){
			try {
				clientSock = serverSock.accept(); //Esperando a un cliente
				System.out.println("New client accepted");

				//Inicializamos el flujo de salida con el cliente
```

```
                    dataOutputStram = new
DataOutputStream(clientSock.getOutputStream());
                    dataOutputStram.writeInt(++clientCounter);
//Enviamos el entero al cliente
                    dataOutputStram.close(); //Cerramos el
flujo de salida
                    clientSock.close();       //Cerramos el
cliente
               } catch (IOException e) {}
          }
     }

     public void closeServer(){
          try {
               serverSock.close();
          } catch (IOException e) {

               e.printStackTrace();
          }
     }
}
```

CLIENTE

```
import java.io.DataInputStream;

import java.io.IOException;

import java.net.Socket;

import java.net.UnknownHostException;

/*

 * CLASE CLIENTE

 *  Esta clase esta diseñada para hacer peticiones al servidor.
```

```
 */
public class Client {

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

          Client myClient= new Client();

          myClient.run();

     }

     Socket clientSocket; //Socket que se conectara al servidor

     DataInputStream inputStraeam; //Flujo de entrada para
recibir los datos del server

     public Client(){

          try {

               clientSocket = new Socket("localhost",6060);
//Nos conectamos al server

               ////Creamos el flujo de entrada a partir del
socket



     inputStraeam=new
DataInputStream(clientSocket.getInputStream());

          } catch (UnknownHostException e) {

               e.printStackTrace();

          } catch (IOException e) {

               e.printStackTrace();

          }
```

```
	}

	public void run(){

		try {

			//Mostramos el dato proporcionado por el servidor

			System.out.println("Your client number is "   +
inputStraeam.readInt());

			//Cerramos el flujo y el socket

			inputStraeam.close();

			clientSocket.close();

		} catch (UnknownHostException e) {

			e.printStackTrace();

		} catch (IOException e) {

			e.printStackTrace();

		}

	}

}
```