UF3-M09
PRÁCTICA 1
IMPLEMENTAR UN CHAT BIDIRECCIONAL ENTRE UN CLIENTE Y UN SERVIDOR, PARTIENDO DE UN CHAT ECHO

EDUARD LARA

EJEMPLO: CHAT CON ECHO

- Clase Server

```java
public class Server {
    private ServerSocket servidor;
    private Socket cliente;
    private ObjectOutputStream salida;
    private ObjectInputStream entrada;
    private String mensaje;
    private boolean fin;

    public Server() throws IOException{
        int puerto = 5000;
        servidor = new ServerSocket(puerto);
        fin = false;
        System.out.println("Servidor establecido");
        System.out.println("Esperando conexion en port:" +
                                                puerto);
    }
    public static void main(String args[]) throws Exception {
        Server miServer = new Server();
        miServer.iniciar();
    }
```

```java
    private void iniciar() throws Exception{
        try {
            cliente = servidor.accept();
            System.out.println("Conexion aceptada de: "+
                                        cliente.getInetAddress());
            salida = new ObjectOutputStream(cliente.getOutputStream());
            entrada = new ObjectInputStream(cliente.getInputStream());

            do {
                mensaje = (String)entrada.readObject();
                if(mensaje.equals("fin\n")) fin=true;
                System.out.println("CLIENTE >>> "+ mensaje);
                salida.writeObject((String)mensaje);
                salida.flush();
                System.out.println("SERVIDOR <<< " + mensaje);
            }while(!fin);
            entrada.close();
            salida.close();
            cliente.close();
            servidor.close();
        }catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```

EJEMPLO: CHAT CON ECHO

- Clase Cliente

```
public class Cliente {
    private ObjectOutputStream salida;
    private ObjectInputStream entrada;
    private String mensaje = "";
    private String server = "localhost";
    private Socket cliente;
    private boolean fin = false;
    Scanner reader = new Scanner(System.in);

    public static void main(String[] args) {
        try {
            new Cliente();
        }catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
    public Cliente() throws Exception {
        System.out. println("Intentando realizar conexion");
        cliente = new Socket(server, 5000);
        System.out.println("Conectado a: "+cliente.getInetAddress().
                                                getHostName());
        salida = new ObjectOutputStream(cliente.getOutputStream());
        entrada = new ObjectInputStream(cliente.getInputStream());
        do {
            System.out.println("Introduce mensaje a enviar: ");
            mensaje = reader.nextLine();
            salida.writeObject(mensaje + "\n");
            salida.flush(); System.out.println("ClIENTE >» "+ mensaje);
            mensaje = (String) entrada.readObject();
            System.out.println("SERVIDOR «< " + mensaje);
            if (mensaje.equals("fin\n")) fin = true;
        }while(!fin);
        salida.close();
        entrada. close();
        cliente.close();
    }
}
```

ENUNCIADO PRACTICA 1

- Cada grupo de 2 personas ha de implementar un chat entre dos equipos de forma directa, donde uno ejercerá de servidor y el otro de cliente
- La comunicación entre los dos equipos se puede realizar a través de la Wifi del Instituto o a través de un móvil que proporcione wifi a los dos equipos.
- Se deben de realizar previamente un estudio de las direcciones IP proporcionadas para enlazar los sockets contra la máquina correcta
- **NOTA: Tanto la clase Cliente como la Server deben de heredar de Thread y realizar en su void run la lectura de las cadenas enviadas para mostrarlas en la consola**