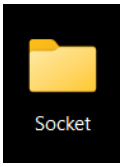
He sacado la información del siguiente link:
https://www.geeksforgeeks.org/socket-programming-in-java/

1.1Creamos una carpeta en el escritorio y la llamamos Socket:



1.2Abrimos cualquier editor de texto, introducimos el siguiente código:

Client.txt | Server.txt

Archivo Editar Ver

```
// A Java program for a Client
import java.io.*;
import java.net.*;

public class Client {
    // initialize socket and input output streams
    private Socket socket = null;
    private DataInputStream input = null;
    private DataOutputStream out = null;

    // constructor to put ip address and port
    public Client(String address, int port)
    {
        // establish a connection
        try {
            socket = new Socket(address, port);
            System.out.println("Connected");

            // takes input from terminal
            input = new DataInputStream(System.in);

            // sends output to the socket
            out = new DataOutputStream(
                socket.getOutputStream());
        }
        catch (UnknownHostException u) {
            System.out.println(u);
            return;
        }
        catch (IOException i) {
            System.out.println(i);
            return;
        }
```

Ln 64, Col 2 | 1.635 caracteres.

```
        // string to read message from input
        String line = "";

        // keep reading until "Over" is input
        while (!line.equals("Over")) {
            try {
                line = input.readLine();
                out.writeUTF(line);
            }
            catch (IOException i) {
                System.out.println(i);
            }
        }

        // close the connection
        try {
            input.close();
            out.close();
            socket.close();
        }
        catch (IOException i) {
            System.out.println(i);
        }
    }

    public static void main(String args[])
    {
        Client client = new Client("127.0.0.1", 5000);
    }
}
```

Y lo guardamos como ‘Cliente.java’.

1.3Luego, abrimos otro archivo de texto en la misma carpeta, e introducimos el siguiente código:

```java
// A Java program for a Server
import java.net.*;
import java.io.*;

public class Server
{
    //initialize socket and input stream
    private Socket          socket   = null;
    private ServerSocket    server   = null;
    private DataInputStream in       =  null;

    // constructor with port
    public Server(int port)
    {
        // starts server and waits for a connection
        try
        {
            server = new ServerSocket(port);
            System.out.println("Server started");

            System.out.println("Waiting for a client ...");

            socket = server.accept();
            System.out.println("Client accepted");

            // takes input from the client socket
            in = new DataInputStream(
                new BufferedInputStream(socket.getInputStream()));

            String line = "";

            // reads message from client until "Over" is sent
```

```
            while (!line.equals("Over"))
            {
                try
                {
                    line = in.readUTF();
                    System.out.println(line);

                }
                catch(IOException i)
                {
                    System.out.println(i);
                }
            }
            System.out.println("Closing connection");

            // close connection
            socket.close();
            in.close();
        }
        catch(IOException i)
        {
            System.out.println(i);
        }
    }

    public static void main(String args[])
    {
        Server server = new Server(5000);
    }
}
```

Y lo guardamos como ‘Server.java’.

2.1: Ahora, cambiamos dos líneas de código en el archivo ‘Client.java’:

```
    public static void main(String args[])
    {
        Client client = new Client("127.0.0.1", 5000);
    }
```

pasa a ser:

```
    public static void main(String args[])
    {
        Client client = new Client("127.0.0.1", 5555);
    }
```

Y la siguiente línea de código:

```
        while (!line.equals("Over")) {
            try {
                line = input.readLine();
                out.writeUTF(line);
            }
```

Pasa a ser:

```
// keep reading until "Over" is input
while (!line.equals("Final")) {
    try {
```

Y en el Server.java cambiamos el puerto de 5000 a 5555:

```
    public static void main(String args[])
    {
        Server server = new Server(5555);
    }
}
```

Una vez esté guardado, abrimos el powershell en la carpeta Socket , y para compilar nuestros archivos introducimos los siguientes comandos:

```
PS C:\Users\[redacted]\OneDrive\Escritorio\Socket> javac Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
PS C:\Users\[redacted]\OneDrive\Escritorio\Socket> javac Server.java
PS C:\Users\[redacted]\OneDrive\Escritorio\Socket>
```

Ahora, en este terminal abrimos el 'Server.java' con el siguiente comando:

java Server.java

```
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Server.java
Server started
Waiting for a client ...
```

Y en otro terminal, abierto simultáneamente, abrimos el 'Client.java' con el siguiente comando:

```
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
```

Comprobamos que los dos terminales están bien conectados con el socket, escribiendo algo en el terminal del cliente, y al darle al enter, vemos que se escribe en el lado del servidor:

```
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
Hola!
Hecho por Jorge Varela Zamora
```

```
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Server.java
Server started
Waiting for a client ...
Client accepted
Hola!
Hecho por Jorge Varela Zamora
```

Podemos salir escribiendo ‘Final’ en el lado del cliente:

```
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
Hola!
Hecho por Jorge Varela Zamora
Final
PS C:\Users\jotaa\OneDrive\Escritorio\Socket>
```

```
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
```

Ahora, vamos a hacer unos experimentos.
Para que nos confirme que hemos mandado un mensaje por el socket, aumentando la legibilidad de los resultados en la terminal, vamos a hacer que en la terminal escriba ‘Enviado: ’ y justo después el mensaje, de la siguiente manera:

```
while (!line.equals("Final")) {
    try {
        line = input.readLine();
        out.writeUTF(line);
        System.out.println("Enviado: "+line);
    }
```

Queremos que también nos confirme que ha llegado un mensaje al servidor, y lo podemos cambiar de la siguiente manera.:

```
while (!line.equals("Over"))
{
    try
    {
        line = in.readUTF();
        System.out.println("Recibido: "+line);

    }
```

Volvemos a compilar los archivos ‘Client.java’ y ‘Server.java’ y los volvemos a ejecutar para comprobar los cambios:

Explicación código:

—----------

127.0.0.1 es localhost
Nos podemos conectar al servidor de otro compañero cambiando la línea de código a su dirección de IP, por ejemplo:

10.227.128.198

Como yo estoy en un portátil, conectado por wifi y no por cable, sólo podré conectarme a servidores de compañeros que hayan creado el suyo en un portátil, y estén conectados por wifi.

La línea de ‘catch (IOException i) {
System.out.println(i);
}’

corta directamente cuando haya una error dentro de las llaves donde estén, y printee en la terminal el error que ha dado, con su correspondiente código.
Lo más normal es que devuelva un error cuando se corte la señal con el cliente, porque el servidor volvería a intentar la conexión con el mismo cliente.

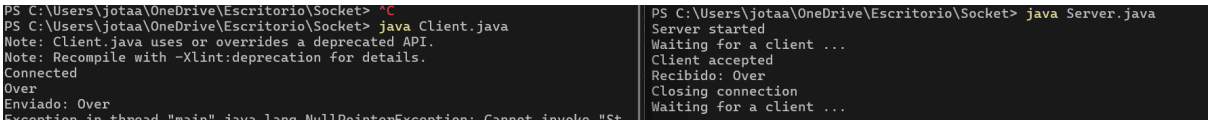
SÓLO se puede conectar un cliente a un servidor a la vez. Si intento conectarme a un servidor de un compañero en el que ya haya otro compañero conectado, no va a funcionar.

EXPERIMENTO para que no cierre el servidor cuando cierra el client, sino que vuelva a esperar una conexión:

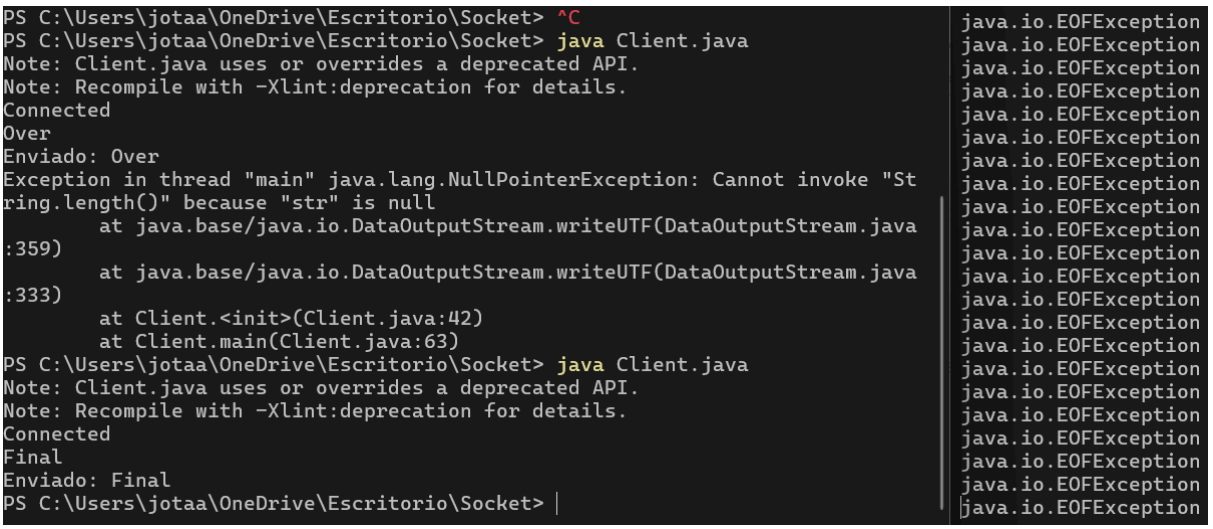
Simplemente lo metemos en un bucle, para que al terminar de detectar el cliente, no de un error, sino que se vuelva a abrir para conectarse con otro:

```java
// A Java program for a Server
import java.net.*;
import java.io.*;

public class Server
{
    //initialize socket and input stream
    private Socket          socket   = null;
    private ServerSocket    server   = null;
    private DataInputStream in       =  null;

    // constructor with port
    public Server(int port)
    {
        // starts server and waits for a connection
        try
        {
            server = new ServerSocket(port);
            System.out.println("Server started");

            while (true) {
                System.out.println("Waiting for a client ...");

                socket = server.accept();
                System.out.println("Client accepted");

                // takes input from the client socket
                in = new DataInputStream(
```

Daría el siguiente resultado:



No obstante, si escribimos ‘Final’, sí que cierra tambien el servidor.
Éste método lo podemos usar para elegir cuál de los dos cerramos, o para ponerle una contraseña al cierre del servidor.


PS C:\Users\jotaa\OneDrive\Escritorio\Socket> ^C
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
Over
Enviado: Over
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "St
ring.length()" because "str" is null
at java.base/java.io.DataOutputStream.writeUTF(DataOutputStream.java
:359)
at java.base/java.io.DataOutputStream.writeUTF(DataOutputStream.java
:333)
at Client.<init>(Client.java:42)
at Client.main(Client.java:63)
PS C:\Users\jotaa\OneDrive\Escritorio\Socket> java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
Final
Enviado: Final
PS C:\Users\jotaa\OneDrive\Escritorio\Socket>
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException
java.io.EOFException