

COMUNICACIÓN CON UN SERVIDOR TELNET.

Índice de contenido

1.COMUNICACIÓN CON UN SERVIDOR TELNET	2
1.1.INSTALACIÓN Y USO DE UN SERVIDOR TELNET	2
1.2.COMUNICACIÓN CON UN SERVIDOR TELNET CON JAVA.....	5

1. COMUNICACIÓN CON UN SERVIDOR TELNET

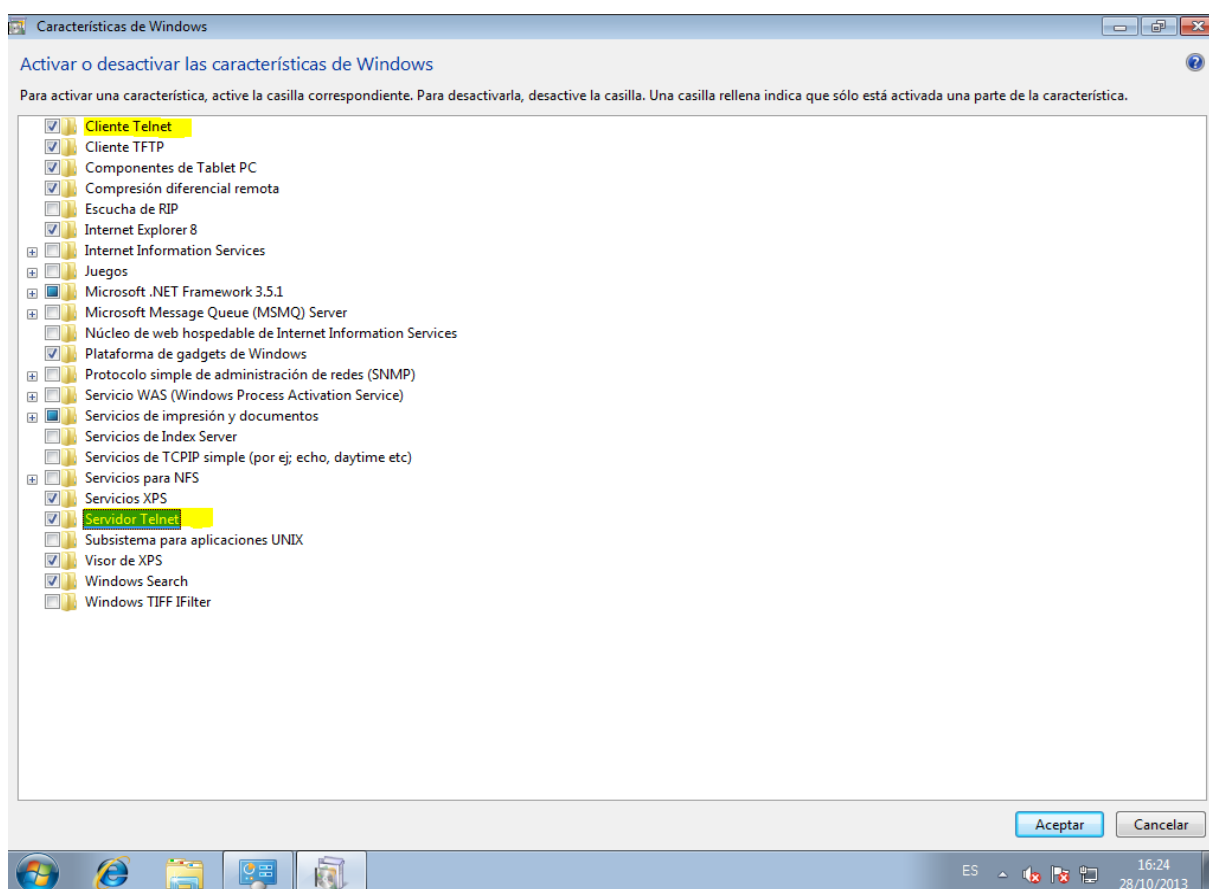
Telnet es un protocolo de red que pertenece a la familia de protocolos de Internet. Permite a los usuarios acceder a un ordenador remoto y realizar tareas como si estuviesen trabajando directamente delante de él. El acceso al ordenador remoto se realiza en modo terminal, es decir, no se muestra una pantalla gráfica (como el escritorio de Windows) para realizar las tareas; se trabaja desde la línea de comandos. Es útil para arreglar fallos de forma remota y para consultar información. Se utiliza bastante en sistemas UNIX-LINUX y en equipos de comunicaciones para la configuración de routers. Por defecto, utiliza el puerto TCP 23.

El principal problema de Telnet es la seguridad, ya que los datos necesarios para conectarse a máquinas remotas (nombre de usuario y clave), no son cifrados y viajan por la red como texto plano, facilitando a cualquiera que espíe la red mediante un programa *sniffer* obtener estos datos.

Telnet sigue un modelo cliente-servidor, para poder utilizarlo necesitamos tener instalado en una máquina un servidor Telnet y en otra máquina el cliente Telnet para poder acceder a ella.

1.1. INSTALACIÓN Y USO DE UN SERVIDOR TELNET

Para instalar un servidor Telnet en Windows 7 pulsamos en *Inicio-> Panel de control-> Programas y características-> Activar y desactivar las características de Windows*; se abre una ventana desde la que podremos activar o desactivar características de Windows. Marcamos las casillas *Servidor Telnet* para **instalar el servidor** y *Cliente Telnet* para **instalar el cliente**. Después se pulsa *Aceptar*, un mensaje nos indica que hay que esperar unos minutos hasta que la instalación finalice, al finalizar puede que nos pida reiniciar el equipo.



También se puede acceder desde: *Inicio-> Panel de control-> Programas->Programas y*

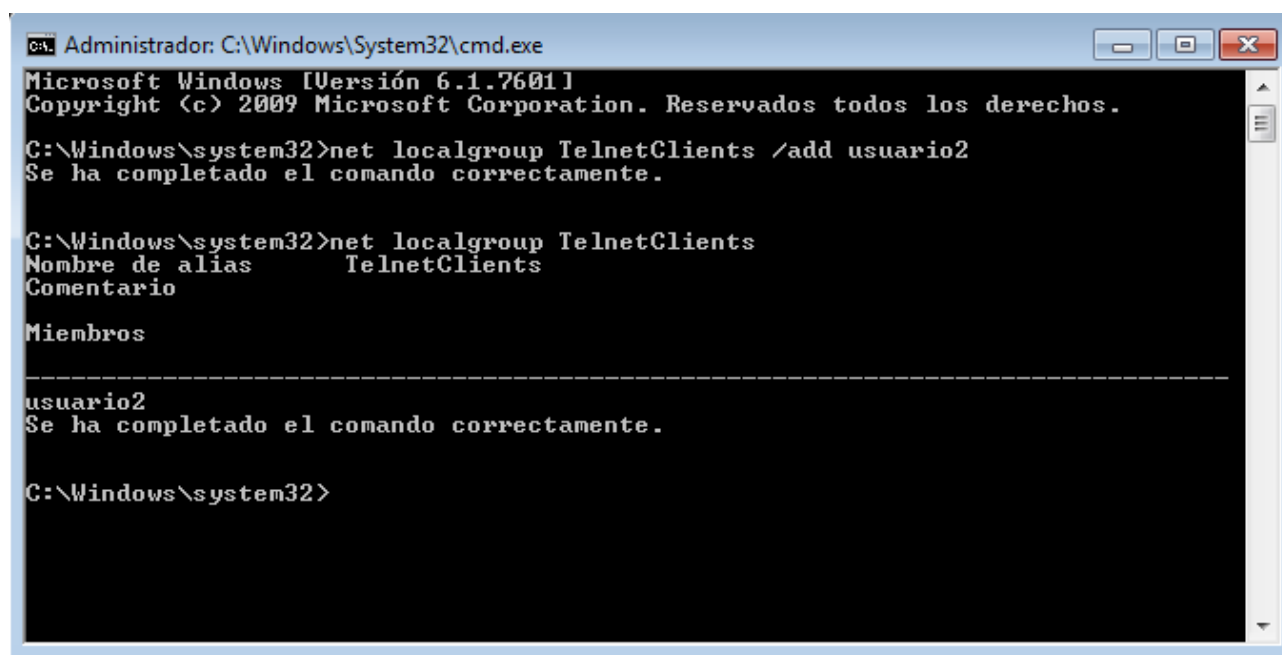
características -> Activar y desactivar las características de Windows.

El servidor Telnet se instala como un **servicio de Windows**, será necesario iniciarlo para poder trabajar con él. Accediendo a los servicios de Windows (*Inicio -> Panel de control -> Sistema y Seguridad -> Herramientas administrativas-> Servicios*) se puede indicar el tipo de inicio: manual, automático, etc. Situados sobre él, botón derecho -> General -> Tipo de Inicio : automático. Después sobre Telnet, botón derecho -> Iniciar.

Para hacer que los usuarios se puedan conectar a la máquina que tiene instalado el servidor Telnet hay que **crear usuarios**. Esto se puede hacer desde *Inicio -> Panel de control-> Cuentas de usuario*; se crea por ejemplo un usuario estándar de nombre *usuario2* y clave *usu2*.

Para permitir que los usuarios obtengan acceso al servidor Telnet, es necesario agregarlos al grupo **TelnetClients**, seguiremos estos pasos:

- Ejecutamos el comando cmd.exe (ubicado en *C:\Windows\System32*) en el modo "**Ejecutar como administrador**". Se abre una ventana DOS.
- Escribimos el siguiente comando para añadir el usuario creado al grupo **TelnetClients**: `net localgroup TelnetClients /add usuario2`. Se debe visualizar el mensaje *Se ha completado el comando correctamente*.
- Para comprobar si se ha añadido el usuario ejecutamos esta orden: `net localgroup TelnetClients`. Se debe ver el usuario.



```
Administrador: C:\Windows\System32\cmd.exe
Microsoft Windows [Versión 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Windows\system32>net localgroup TelnetClients /add usuario2
Se ha completado el comando correctamente.

C:\Windows\system32>net localgroup TelnetClients
Nombre de alias      TelnetClients
Comentario

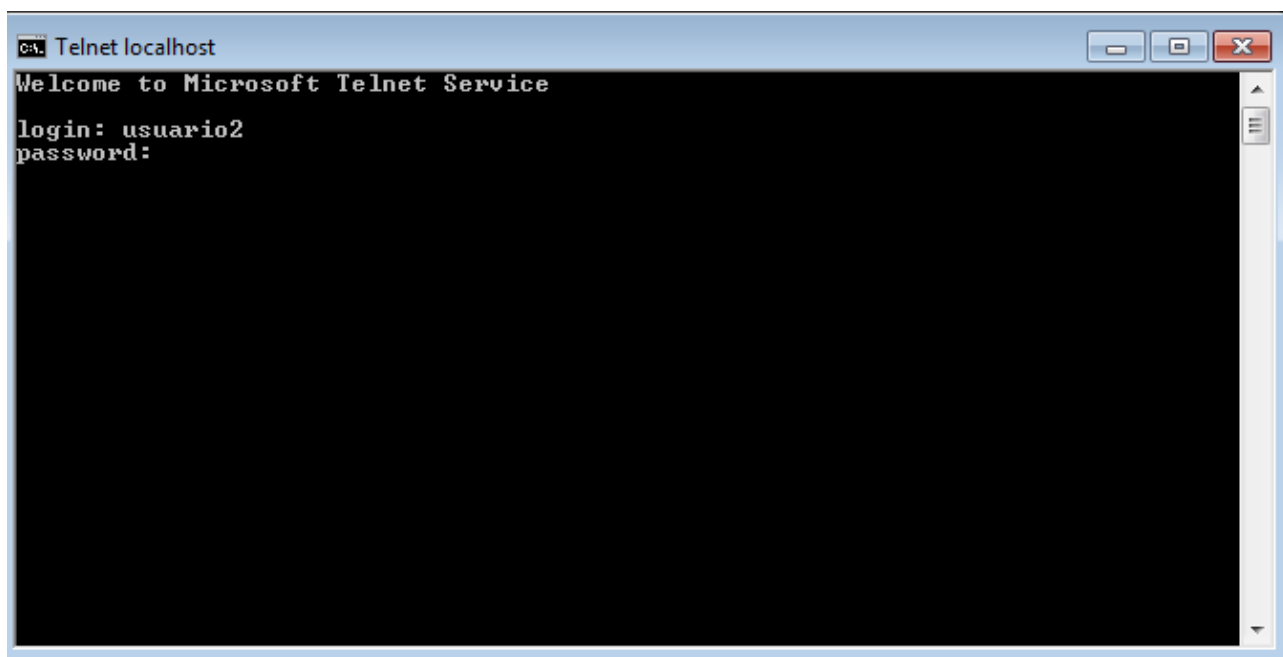
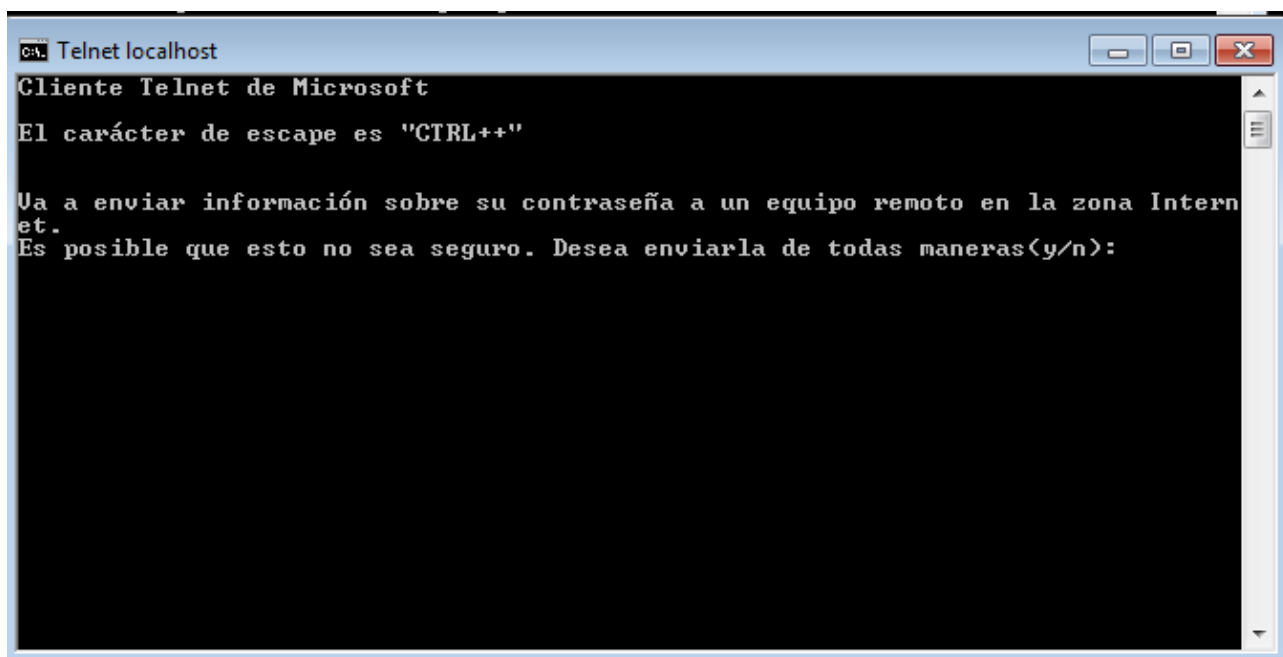
Miembros
-----
usuario2
Se ha completado el comando correctamente.

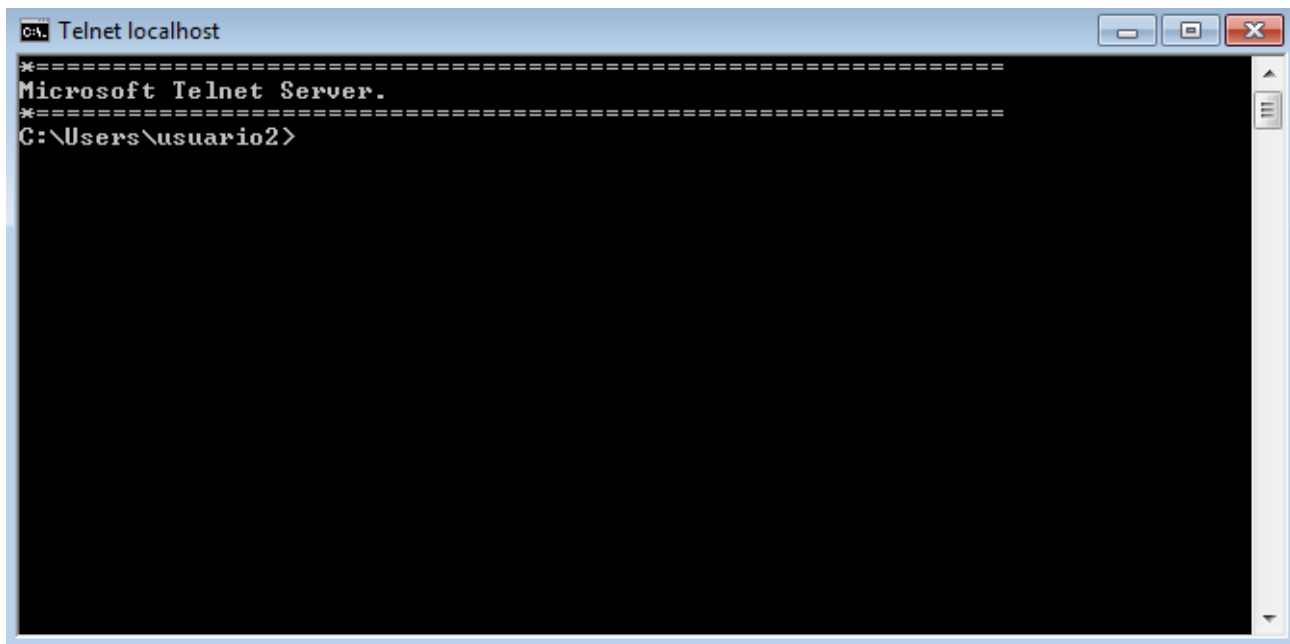
C:\Windows\system32>
```

Para probar la conexión (en modo local) con el servidor Telnet ejecutamos desde la línea de comandos la orden `telnet localhost` (tiene que estar el cliente Telnet instalado).

Se muestra la pantalla del cliente Telnet, que muestra un mensaje que nos pregunta si deseamos enviar la contraseña aunque no sea seguro (ya que no viajará encriptada por la red), escribimos s para aceptar (si pulsamos y, en lugar de s se desconecta). Después muestra una nueva pantalla con el mensaje de bienvenida y nos pide el nombre de usuario y la contraseña, los escribimos (*usuario2* y *usu2*); una vez comprobados se muestra la consola desde la que podemos escribir comandos e interactuar con el equipo remoto (que en este caso es el mismo equipo donde está el servidor).

También podemos entrar en el sistema con el usuario usuario2, ejecutamos telnet localhost desde el cmd y ya nos conectamos.





1.2. COMUNICACIÓN CON UN SERVIDOR TELNET CON JAVA

La librería **Apache Commons Net** proporciona la clase **TelnetClient** (extiende **SocketClient**) que implementa el sencillo terminal virtual de red (NVT) para el protocolo **Telnet** según RFC 854.

Presenta dos tipos de constructores:

CONSTRUCTOR	MISIÓN
TelnetClient()	Constructor por defecto, establece como tipo de terminal VT100
TelnetClient(String termType)	Se establece como tipo de terminal el especificado en el String <i>termtype</i> (XTerm, VT100, VT200, etc.)

La clase utiliza el método *connect()* de la clase **SocketClient** para conectarse al servidor. Una vez conectado se obtienen un **InputStream** y un **OutputStream** para leer y enviar datos a través de la conexión que se pueden obtener mediante los métodos *getInputStream()* y *getOutputStream()*.

Xterm es un emulador de terminal para el sistema de ventanas *X Window System*.

Algunos métodos de la clase **TelnetClient** son:

MÉTODOS	MISIÓN
void disconnect()	Desconecta la sesión Telnet, cierra los input y output stream, así como el socket
InputStream getInputStream()	Devuelve el stream de entrada de la conexión Telnet, se usa para leer las respuestas que envía el servidor Telnet
OutputStream getOutputStream()	Devuelve el stream de salida de la conexión Telnet, se usa para enviar los comandos al servidor Telnet.

En el siguiente ejemplo nos conectamos al servidor **Telnet** local con nombre de usuario *usuario2* y

clave *usu2* y le enviamos el comando **DIR** para que nos muestre el contenido del directorio. Vamos a ver paso a paso las operaciones de lectura mostrando lo que nos envía el servidor y de escritura enviando lo que nos pide (login, password, comandos,...)

En primer lugar creamos el cliente Telnet mediante el segundo constructor estableciendo como tipo de terminal XTerm. A continuación se realiza la conexión con el servidor mediante el método *connect()*. Una vez conectados se crean el **OutputStream** o flujo de salida para enviar datos al servidor y el **InputStream** o flujo de entrada para leer los datos que el servidor envía:

```
import java.io.*;
import org.apache.commons.net.telnet.*;

public class ClienteTelnet1 {
    static InputStream in;
    static PrintStream out;
    public static void main(String[] args) {
        TelnetClient telnet = new TelnetClient("xterm");
        String server= "localhost";
        String login= "usuario2";
        String password="usu2";
        try {
            telnet.connect(server) ;//conexión con el servidor
            //Se definen los flujos para enviar y recibir datos
            out = new PrintStream(telnet.getOutputStream());
            in = telnet.getInputStream();
```

Recordemos que lo primero que hace el servidor Telnet es mostrarnos en pantalla el mensaje de bienvenida y pedirnos el login del usuario. Por ello hay que hacer una primera lectura de lo que nos envía, utilizando el **InputStream**, y mostrarla en pantalla. Eso se hace en el método *LecturaDatos()*. A continuación el servidor nos pide el login del usuario, para mandárselo necesitaremos escribir en el **OutputStream**; se usará el método *EnvioDatos(cadena)* para enviar datos al servidor:

```
//Recibe las primeras líneas y pide login:
LecturaDatos();
//Se envia el nombre de usuario
EnvioDatos(login);
```

Una vez que hemos enviado el login del usuario el servidor responde pidiéndonos el password, hay que hacer de nuevo una lectura de lo que nos envía mediante el método *LecturaDatos()* seguido de una escritura enviando el password:

```
//Ahora pide el password:
LecturaDatos();
//Se envia el password
EnvioDatos(password);
```

Una vez identificados se hace una nueva lectura de lo que envía el servidor, en este caso unas líneas de presentación y el prompt desde el que se pueden introducir los comandos; usamos de nuevo el método *LecturaDatos()*. A continuación le enviamos

el comando DIR mediante el método de envío de datos. Para poder recoger la salida del comando será necesario realizar una nueva lectura:

```
//Se muestra la presentacion y el prompt
LecturaDatos();
//se manda el comando DIR
EnvioDatos("DIR");
//lectura visualizacion de la salida del comando DIR
LecturaDatos();
```

Por último nos desconectamos del servidor Telnet:

```
//desconectar
telnet.disconnect();
}catch (Exception e) {
    e.printStackTrace();
}
}
}///fin main
```

El método para leer los datos que el servidor nos envía (lectura del **InputStream**) es el siguiente:

```
//Lectura y visualización de lo que envia el servidor Telnet
static void LecturaDatos() throws IOException {
    byte[] buff = new byte[1024];
    int nbytes ;
    nbytes = in.read(buff); //lee los bytes
    System.out.print(new String(buff, 0, nbytes)); //los visualiza
} //lectura
```

El método para que el usuario envíe datos al servidor (escritura en el **OutputStream**) que recibe la cadena a enviar es el siguiente:

```
//Envío de datos al servidor Telnet
private static void EnvioDatos(String cad) {
    out.println(cad) ;
    out.flush();
} //escritura

} // .. ClienteTelnet1
```

La ejecución del programa muestra la siguiente salida:

```
CA: C:\Windows\system32\cmd.exe

C:\Java\UD4>javac ClienteTelnet1.java

C:\Java\UD4>java ClienteTelnet1
Welcome to Microsoft Telnet Service

login: usuario2
password:

=====
Microsoft Telnet Server.
=====
C:\Users\usuario2>DIR
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 4863-C309

Directorio de C:\Users\usuario2

28/10/2013  16:51    <DIR>          .
28/10/2013  16:51    <DIR>          ..
28/10/2013  16:51    <DIR>          Contacts
28/10/2013  16:51    <DIR>          Desktop
28/10/2013  16:51    <DIR>          Documents
28/10/2013  16:51    <DIR>          Downloads
28/10/2013  16:51    <DIR>          Favorites
28/10/2013  16:51    <DIR>          Links
28/10/2013  16:51    <DIR>          Music
28/10/2013  16:51    <DIR>          Pictures
28/10/2013  16:51    <DIR>          Saved Games
28/10/2013  16:51    <DIR>          Searches
28/10/2013  16:51    <DIR>          Videos
                0 archivos                0 bytes
                13 dirs 11.054.817.280 bytes libres

C:\Users\usuario2>
C:\Java\UD4>
```

En el ejemplo que acabamos de ver, se hace una lectura (llamada al método *LecturaDatos()*) cada vez que se van a recibir datos del servidor.

Podemos hacer que el **proceso de lectura** se realice en un **hilo**, una vez conectados al servidor Telnet se inicia el hilo y se leen los datos o respuestas que el servidor va enviando desde el hilo. El hilo recibe en su constructor el objeto *TelnetClient*. Vamos a verlo en este otro ejemplo:


```
import java.io.*;
import org.apache.commons.net.telnet.*;

//Lectura de las respuestas que envia el servidor
class LecturaRespuestas extends Thread {
    TelnetClient tc = null;
    //constructor
    public LecturaRespuestas(TelnetClient tcli) {tc=tcli;}
    //
    public void run() {
        InputStream in = tc.getInputStream() ;
        byte[] buff = new byte[1024];
        int nbytes = 0;
        try{
            nbytes = in.read(buff) ;//lectura de bytes
            while (nbytes > 0) {
                System.out.print(new String(buff, 0, nbytes));
                nbytes = in.read (buff) ;
            }
        }catch (IOException e) {
            System.err.println("ERROR AL LEER EL InputStream "+ e.getMessage());
        }
    }
}

// run
// fin del hilo
```

Lo más normal es que el usuario escriba por teclado las órdenes a enviar al servidor. Una vez conectados e iniciado el hilo de lectura de respuestas del servidor hacemos un bucle desde el que se introducen datos por teclado y se envían al servidor. El proceso termina cuando el usuario escriba *exit*. El código es el siguiente:

```

public class ClienteTelnet2 {
    static TelnetClient telnet = new TelnetClient("xterm");
    public static void main(String[] args) {
        String server = "localhost";
        byte[] buff = new byte[1024];
        int nbytes = 0;
        boolean repetir = true;
        OutputStream out;
        try {
            telnet.connect(server); //conexión al servidor
            System.out.println("Cliente Telnet conectado .... ");
        } catch (IOException e) {
            System.err.println("ERROR AL CONECTAR EL CLIENTE" + e.getMessage());
        }
        // LANZAR HILO
        LecturaRespuestas hilo = new LecturaRespuestas(telnet);
        hilo.start();

        //stream donde escribo las órdenes que mando al servidor
        out = telnet.getOutputStream();

        while (repetir == true) {
            try {
                //lectura de los comandos por teclado
                //se leen bytes
                nbytes = System.in.read(buff);
            } catch (IOException e) {
                System.err.println("ERROR AL LEER POR TECLADO" + e.getMessage());
            }
            if (nbytes > 0) {
                //convierte bytes a cadena
                String cadena = new String(buff, 0, nbytes);
                try {
                    //enviar los comandos al servidor
                    out.write(buff, 0, nbytes);
                    out.flush();
                } catch (IOException e) {
                    System.err.println("ERROR AL ESCRIBIR EN EL OutputStream " + e.getMessage());
                }
                //el proceso termina cuando se escribe exit
                if (cadena.trim().equalsIgnoreCase("exit"))
                    repetir = false;
            }
        } //fin while
        try {
            telnet.disconnect();
            System.out.println("Fin de la conexión .... ");
        } catch (IOException e) {
            System.err.println("ERROR AL CERRAR EL CLIENTE TELNET" + e.getMessage());
        }
        System.exit(0); //salir
    } //main
} // .. ClienteTelnet2

```

Al ejecutarlo introducimos por teclado el login del usuario y luego su password, nos muestra las líneas de presentación y a continuación el prompt desde el que podemos ir introduciendo comandos DOS hasta que introducimos exit. Un ejemplo de ejecución se muestra a continuación:

```
Símbolo del sistema - java ClienteTelnet2

C:\>cd java/ud4

C:\Java\UD4>java ClienteTelnet2
Cliente Telnet conectado ....
Welcome to Microsoft Telnet Service

login: usuario2
usuario2
password: usu2

=====
Microsoft Telnet Server.
=====
C:\Users\usuario2>dir
dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 4863-C309

Directorio de C:\Users\usuario2

28/10/2013  16:51    <DIR>          .
28/10/2013  16:51    <DIR>          ..
28/10/2013  16:51    <DIR>          Contacts
29/10/2013  16:43    <DIR>          Desktop
28/10/2013  16:51    <DIR>          Documents
28/10/2013  16:51    <DIR>          Downloads
28/10/2013  16:51    <DIR>          Favorites
28/10/2013  16:51    <DIR>          Links
28/10/2013  16:51    <DIR>          Music
28/10/2013  16:51    <DIR>          Pictures
28/10/2013  16:51    <DIR>          Saved Games
28/10/2013  16:51    <DIR>          Searches
28/10/2013  16:51    <DIR>          Videos
                0 archivos                0 bytes
                13 dirs 10.836.021.248 bytes libres

C:\Users\usuario2>_
```