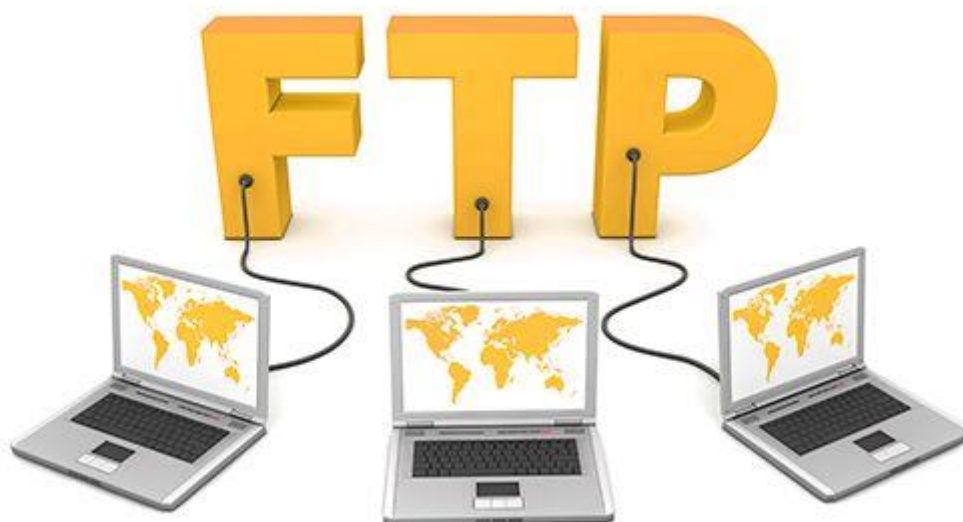


PSP



PROGRAMACIÓN DE
SERVICIOS Y PROCESOS

13/03/2020

DAVID BORREGO ASENCIO

ÍNDICE

CAPTURAS DE PANTALLA

1

CÓDIGO JAVA

5

CAPTURAS DE PANTALLA

Arrancamos nuestra aplicación, por defecto está configurada para que se conecte a nuestro servidor FTP con los siguientes valores:

- Servidor = "127.0.0.1";
- Usuario = "david";
- Contraseña = "Stodium2019";

La pantalla principal de nuestro programa es la siguiente:






Nuestra aplicación cuenta con las siguientes funcionalidades implementadas:

- Entrar en directorios remoto
- Crear directorios remotos
- Eliminar directorios remotos
- Renombrar directorios remotos
- Subir ficheros al directorio remoto
- Bajar ficheros del directorio remoto al directorio local
- Establecer ruta de bajada (carpeta del equipo local)
- Renombrar ficheros remotos
- Borrar ficheros remotos

En la lista, se pueden observar todos las carpetas y ficheros alojados en nuestro servidor FTP. En la zona inferior podemos ver la ruta actual en la que se encuentra nuestro programa. Podremos ir interactuando con nuestra carpeta raíz para realizar las distintas funciones.

Directorio:

A continuación, vamos a explicar las distintas funcionalidades de los botones de navegación.

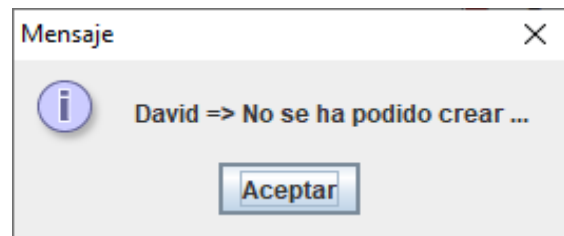
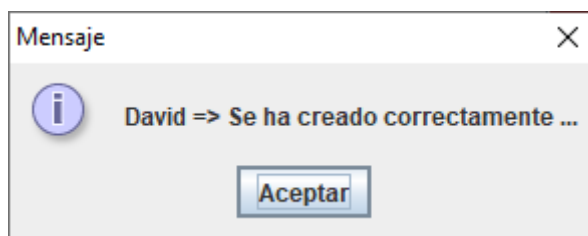
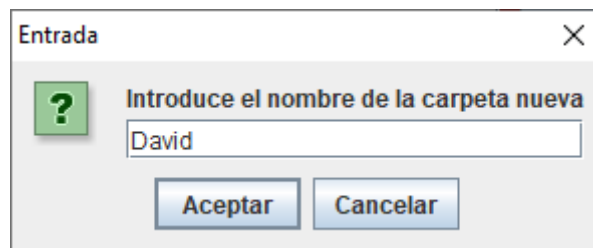
-  Botones que permiten la navegación del usuario por los distintos directorios que haya visitado.
-  Botón Refrescar: Este botón permite refrescar los archivos disponibles en el directorio que nos encontramos.
-  Botón Inicio: Este botón nos lleva directamente a la carpeta raíz de nuestro servidor FTP.

Destacar que las carpetas son fácilmente localizables en nuestro servidor FTP ya que vienen precedidas por el prefijo “(DIR)”.

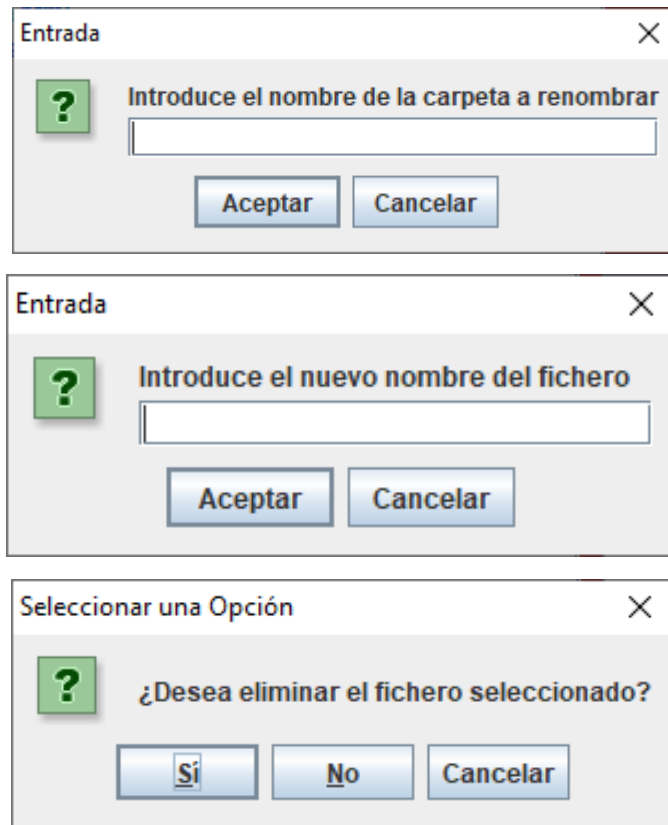


Para navegar por los distintos directorios basta con hacer clic dos veces sobre la carpeta que deseemos en la lista, y nos abrirá la subcarpeta dentro de nuestro servidor FTP.

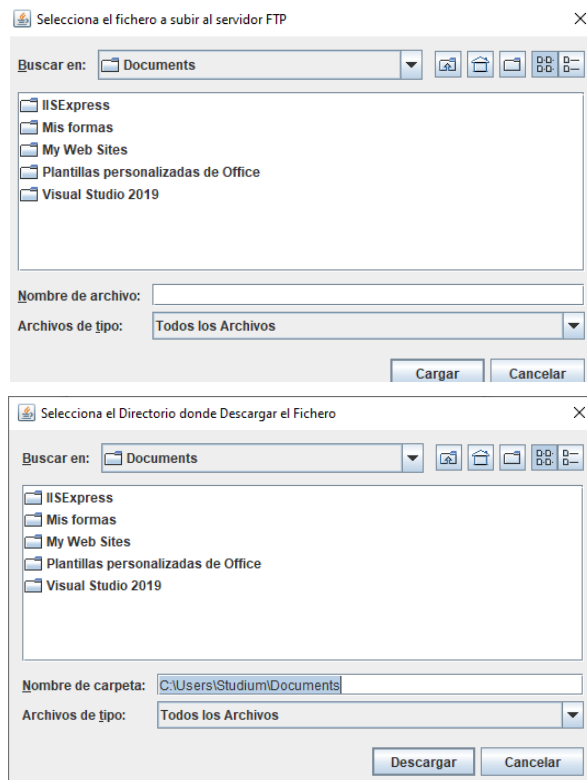
Para crear una nueva carpeta se nos abrirá un cuadro de diálogo pidiéndonos que indiquemos el nombre de la nueva carpeta. Nos saldrá los respectivos mensajes, dependiendo de si la operación se haya completado exitosamente o no.



La misma pantalla de confirmación la podemos encontrar en distintas partes de nuestro programa, controlando así las distintas acciones realizadas por el usuario:



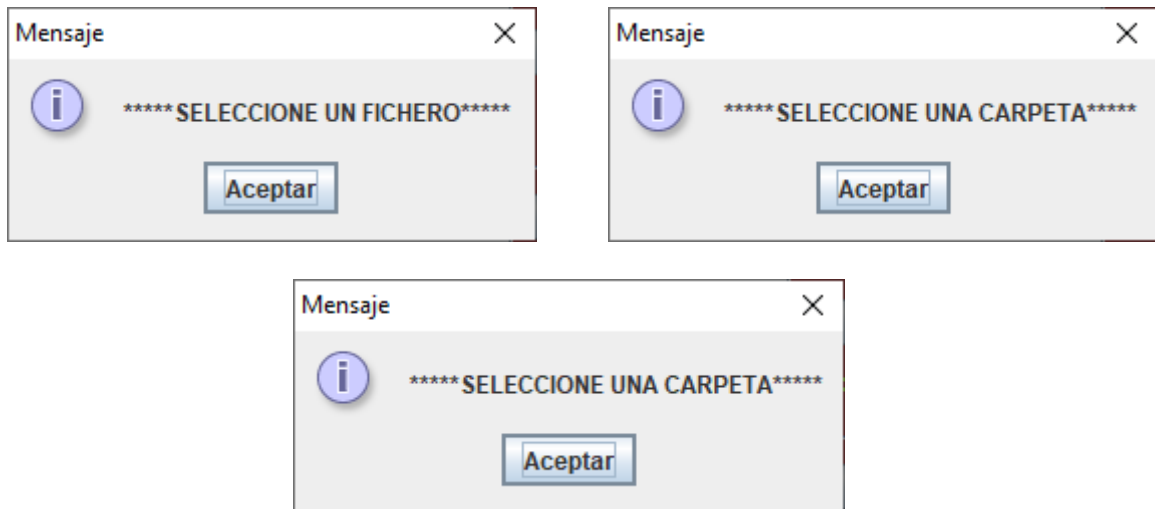
En caso de que queramos subir o bajar un fichero a nuestro servidor FTP se nos abrirá una nueva ventana para que podamos navegar a través de los distintas carpetas alojadas en nuestro equipo:



La aplicación también recoge los distintos errores en las acciones que puedan ser accionados por el usuario, en concreto:

- Renombrar una carpeta pulsando el botón de renombrar en el apartado ficheros.
- Renombrar un fichero pulsando el botón de renombrar en el apartado carpetas.
- Eliminar una carpeta pulsando el botón de eliminar en el apartado ficheros.
- Eliminar un fichero pulsando el botón de eliminar en el apartado carpetas.
- Pulsar botones de eliminar o renombrar sin seleccionar ningún archivo de la lista.

Dependiendo de cada caso se nos mostrarán las siguientes ventanas:



CÓDIGO JAVA

```
import javax.swing.*;
import javax.swing.border.*;
import javax.swing.event.*;
import org.apache.commons.net.PrintCommandListener;
import org.apache.commons.net.ftp.FTP;
import org.apache.commons.net.ftp.FTPClient;
import org.apache.commons.net.ftp.FTPFile;
import java.awt.*;
import java.awt.event.*;
import java.io.*;

public class ClienteFTP extends JFrame implements
ListSelectionListener, MouseListener, ActionListener {

    private String servidor;
    private String user;
    private String pasw;
    private static final long serialVersionUID = 1L;
```



```

    private JPanel contentPane;;
    private static JList<String> lista;
    private static FTPClient cliente = new FTPClient();//
cliente FTP
    private static FTPFile[] files;
    private static String directorioInicial = "/"; //DIRECTORIO
CUANDO ARRANCAMOS (ES LA RAIZ DE LA CARPETA COMPARTIDA)
    private static String directorioActual =
directorioInicial;//DIRECTORIO SELECCIONADO EN CADA MOMENTO
    private JButton boton_adelantar;
    private JButton boton_refrescar;
    private JButton boton_volver;
    private JButton boton_home;
    private static int contador;
    private static String acumulador_directorios[]= new
String[50];
    private JLabel laber_directorio;
    private JButton boton_crear_carpeta;
    private JButton boton_eliminar_carpeta;
    private JButton boton_renombrar_carpeta;
    private JButton boton_renombrar_fichero;
    private JButton boton_bajar_fichero;
    private JButton boton_subir_fichero;
    private JButton boton_borrar_fichero;
    /**
     * Create the frame.
     */
    public ClienteFTP() throws IOException
    {
        acumulador_directorios[0] = "/";
        contador = 0;
        servidor = "127.0.0.1";
        user = "david";
        pasw = "Studium2019";

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 489, 488);
        contentPane = new JPanel();
        contentPane.setBackground(new Color(165, 42, 42));
        contentPane.setForeground(SystemColor.info);
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        lista = new JList<String>();
        lista.setForeground(new Color(0, 102, 255));

```

```

        lista.setSelectionMode(ListSelectionModel.SINGLE_SELECTION)
;
        JScrollPane scrollPane = new JScrollPane(lista);
        scrollPane.setPreferredSize(new Dimension(335, 420));
        scrollPane.setBounds(new Rectangle(5, 65, 335, 420));
        scrollPane.setBounds(10, 47, 335, 384);
        contentPane.add(scrollPane);

        boton_crear_carpeta = new JButton("Crear");
        boton_crear_carpeta.setBounds(355, 99, 102, 23);
        contentPane.add(boton_crear_carpeta);

        boton_eliminar_carpeta = new JButton("Eliminar");
        boton_eliminar_carpeta.setBounds(355, 133, 102, 23);
        contentPane.add(boton_eliminar_carpeta);

        boton_renombrar_carpeta = new JButton("Renombrar");
        boton_renombrar_carpeta.setBounds(355, 167, 102, 23);
        contentPane.add(boton_renombrar_carpeta);

        boton_volver = new JButton("");
        boton_volver.setBounds(10, 9, 30, 30);
        boton_volver.setIcon(new
javax.swing.ImageIcon(ClienteFTP.class.getResource("/ClienteFTP/
volver.png")));
        contentPane.add(boton_volver);

        boton_adelantar = new JButton("");
        boton_adelantar.setBounds(43, 9, 30, 30);
        boton_adelantar.setIcon(new
javax.swing.ImageIcon(ClienteFTP.class.getResource("/ClienteFTP/
adelantar.png")));
        contentPane.add(boton_adelantar);

        boton_refrescar = new JButton("");
        boton_refrescar.setBounds(105, 9, 30, 30);
        boton_refrescar.setIcon(new
javax.swing.ImageIcon(ClienteFTP.class.getResource("/ClienteFTP/
actualizar-removebg-preview.png")));
        contentPane.add(boton_refrescar);

        boton_home = new JButton("");
        boton_home.setIcon(new
javax.swing.ImageIcon(ClienteFTP.class.getResource("/ClienteFTP/
home.png")));
        boton_home.setBounds(145, 9, 30, 30);
        contentPane.add(boton_home);

```

```

JLabel lblNewLabel = new JLabel("CARPETAS");

lblNewLabel.setHorizontalAlignment(SwingConstants.CENTER);
lblNewLabel.setForeground(new Color(173, 255, 47));
lblNewLabel.setFont(new Font("Arial", Font.BOLD,
17));

lblNewLabel.setBounds(355, 47, 102, 23);
contentPane.add(lblNewLabel);

JSeparator separator = new JSeparator();
separator.setBounds(355, 81, 102, 2);
contentPane.add(separator);

JLabel lblFicheros = new JLabel("FICHEROS");

lblFicheros.setHorizontalAlignment(SwingConstants.CENTER);
lblFicheros.setForeground(new Color(173, 255, 47));
lblFicheros.setFont(new Font("Arial", Font.BOLD,
17));

lblFicheros.setBounds(355, 201, 102, 23);
contentPane.add(lblFicheros);

JSeparator separator_1 = new JSeparator();
separator_1.setBounds(355, 235, 102, 2);
contentPane.add(separator_1);

boton_renombrar_fichero = new JButton("Renombrar");
boton_renombrar_fichero.setBounds(355, 317, 102, 23);
contentPane.add(boton_renombrar_fichero);

boton_bajar_fichero = new JButton("Bajar");
boton_bajar_fichero.setBounds(355, 283, 102, 23);
contentPane.add(boton_bajar_fichero);

boton_subir_fichero = new JButton("Subir");
boton_subir_fichero.setBounds(355, 249, 102, 23);
contentPane.add(boton_subir_fichero);

boton_borrar_fichero = new JButton("Borrar");
boton_borrar_fichero.setBounds(355, 351, 102, 23);
contentPane.add(boton_borrar_fichero);

laber_directorio = new JLabel("");
laber_directorio.setForeground(new Color(173, 255,
47));
laber_directorio.setFont(new Font("Arial", Font.BOLD
| Font.ITALIC, 14));

```

```

        laber_directorio.setBounds(91, 432, 368, 27);
        contentPane.add(laber_directorio);

        JLabel lblDirectorio = new JLabel("Directorio:");
        lblDirectorio.setForeground(new Color(173, 255, 47));

        lblDirectorio.setBackground(SystemColor.textHighlight);
        lblDirectorio.setFont(new Font("Arial", Font.BOLD,
14));
        lblDirectorio.setBounds(10, 432, 101, 27);
        contentPane.add(lblDirectorio);

        // --- AÑADIMOS LOS LISTENER---
        boton_renombrar_fichero.addActionListener(this);
        boton_borrar_fichero.addActionListener(this);
        boton_subir_fichero.addActionListener(this);
        boton_bajar_fichero.addActionListener(this);
        boton_renombrar_carpeta.addActionListener(this);
        boton_eliminar_carpeta.addActionListener(this);
        boton_crear_carpeta.addActionListener(this);
        boton_home.addActionListener(this);
        boton_volver.addActionListener(this);
        boton_adelantar.addActionListener(this);
        boton_refrescar.addActionListener(this);
        lista.addMouseListener(this);
        lista.addListSelectionListener(this);
        setResizable(false);
        setLocationRelativeTo(null);
        setVisible(true);

        //Para ver los comandos que se originan
        cliente.addProtocolCommandListener(new
PrintCommandListener(new PrintWriter (System.out)));

        //-----

        realiza_conexion();

        // --- RELLENAMOS LA JLIST POR PRIMERA VEZ---

        //Se establece el directorio de trabajo actual
        cliente.changeWorkingDirectory(directorioInicial);
        //Obteniendo ficheros y directorios del directorio
actual
        files = cliente.listFiles();
        llenarLista(files);

```

```

//-----

}

public static void main(String[] args) throws IOException
{
    new ClienteFTP();
}

public void DescargarFichero(String NombreCompleto, String
nombreFichero)
{
    File file;
    String archivoyCarpetaDestino = "";
    String carpetaDestino = "";
    JFileChooser f = new JFileChooser();
    //solo se pueden seleccionar directorios

    f.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
    //título de la ventana
    f.setDialogTitle("Selecciona el Directorio donde
Descargar el Fichero");
    int returnVal = f.showDialog(null, "Descargar");
    if (returnVal == JFileChooser.APPROVE_OPTION)
    {
        file = f.getSelectedFile();
        //obtener carpeta de destino
        carpetaDestino =
(file.getAbsolutePath()).toString();
        //construimos el nombre completo que se creará
en nuestro disco
        archivoyCarpetaDestino = carpetaDestino +
File.separator + nombreFichero;
        try
        {
            cliente.setFileType(FTP.BINARY_FILE_TYPE);
            BufferedOutputStream out = new
BufferedOutputStream(new
FileOutputStream(archivoyCarpetaDestino));
            if (cliente.retrieveFile(NombreCompleto,
out))
                JOptionPane.showMessageDialog(null,
nombreFichero + " => Se ha descargado correctamente ...");

```

```

        else
            JOptionPane.showMessageDialog(null,
nombreFichero + " => No se ha podido descargar ...");
            out.close();
        }
        catch (IOException e1)
        {
            e1.printStackTrace();
        }
    }
} // Final de DescargarFichero

public static void llenarLista(FTPFile[] files)
{
    if (files == null)
        return;

    //Se crea un objeto DefaultListModel
    DefaultListModel<String> modeloLista = new
DefaultListModel<String>();
    modeloLista = new DefaultListModel<String>();

    //Se eliminan los elementos de la lista
    lista.removeAll();

    //Se recorre el array con los ficheros y directorios
que se cargaran en la lista
    for (int i = 0; i < files.length; i++)
    {
        if (!(files[i].getName()).equals(".")) &&
!(files[i].getName()).equals(".."))
        {
            //Nos saltamos los directorios . y ..
            //Se obtiene el nombre del fichero o
directorio

            String f = files[i].getName();

            try
            {
                String z = new
String(f.getBytes("ISO-8859-1"), "UTF-8"); //CAMBIAMOS EL
FORMATO DEL STRING PARA QUE SE VEAN LAS TILDES

                //Si es directorio se añade al nombre
(DIR)

                if (files[i].isDirectory()) z =
"(DIR) " + z;

```

```

        //Se añade el nombre del fichero o
        directorio al listmodel
        modeloLista.addElement(z);

        } catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    try
    {
        //Se asigna el modelo al JList
        //Se muestra en pantalla la lista de ficheros

        lista.setModel(modeloLista);

        //System.out.println("El directorioSeleccionado
vare: " + directorioActual);
    }
    catch (Exception e )
    {
        System.out.println(e.getMessage());
    }
}

public boolean comprueba_si_es_directorio(String cadena)
{
    String[] partes = cadena.split(" ");

    if (partes[0].equals("(DIR)"))
    {
        //System.out.println(partes[0]);
        //System.out.println(partes[1]);
        return true;
    }
    else
    {
        return false;
    }
}

public void BorrarFichero(String NombreCompleto, String
nombreFichero)

```

```

    {
        //Pide confirmación
        int seleccion = JOptionPane.showConfirmDialog(null,
            "¿Desea eliminar el fichero seleccionado?");
        if (seleccion == JOptionPane.OK_OPTION)
        {
            try
            {
                if (cliente.deleteFile(NombreCompleto))
                {
                    String m = nombreFichero + " =>
Eliminado correctamente... ";
                    JOptionPane.showMessageDialog(null,
m);

                    //llenar la lista con los ficheros
del directorio actual
                    //Obteniendo ficheros y directorios
del directorio actual

                    //RECARGAR LA LISTA
files = cliente.listFiles();
llenarLista(files);

                }
                else
                    JOptionPane.showMessageDialog(null,
nombreFichero + " => No se ha podido eliminar ...");
            }
            catch (IOException e1)
            {
                e1.printStackTrace();
            }
        }
    } // Final de BorrarFichero

    public boolean SubirFichero(String archivo, String
soloNombre) throws IOException
    {
        cliente.setFileType(FTP.BINARY_FILE_TYPE);
        BufferedInputStream in = new BufferedInputStream(new
FileInputStream(archivo));
        boolean ok = false;
        //directorio de trabajo actual
        cliente.changeWorkingDirectory(directorioActual);
        if (cliente.storeFile(soloNombre, in))
        {

```



```

        String s = " " + soloNombre + " => Subido
correctamente...";
        JOptionPane.showMessageDialog(null, s);

        //llenar la lista con los ficheros del
directorio actual
        //Obteniendo ficheros y directorios del
directorio actual
        files = cliente.listFiles();
        llenarLista(files);

    }
    else
        JOptionPane.showMessageDialog(null, "=> Error al
subir el archivo...");
        return ok;
    } // final de SubirFichero

    public void realiza_conexion()
    {
        // -- CONECTAMOS AL SERVIDOR --
        try
        {
            cliente.enterLocalPassiveMode();//IMPORTANTE
            cliente.connect(servidor);
            cliente.login(user, pasw); //Si el boolean login
devuelve true, significa que se ha conectado exitosamente.
            System.out.println("Conexión realizada con
éxito.");
        } catch (Exception e)
        {
            System.out.println(e.getMessage());
        }
    }

    public void valueChanged(ListSelectionEvent le)
    {
        if (le.getValueIsAdjusting()) //CADA VEZ QUE
PINCHAMOS O CAMBIA EL VALOR SELECCIONADO EN LA LISTA
        {
            System.out.println("Fichero Seleccionado:
"+lista.getSelectedValue().toString());
        }
    }

    @Override
    public void mouseClicked(MouseEvent me)

```

```

    {
        try
        {
            if (me.getClickCount() == 2) //SI EL USUARIO
HACE CLIC DOS VECES
            {

                String elementoSeleccionadoEnLaLista =
lista.getSelectedValue().toString();

                if
(comprueba_si_es_directorio(elementoSeleccionadoEnLaLista))
//COMPROBAMOS SI LO SELECCIONADO ES DE TIPO(DIR)
                {

                    //VA ACUMULANDO LAS RUTAS RECORRIDAS
POR EL USUARIO

                    directorioActual = directorioActual
+ elementoSeleccionadoEnLaLista.substring(6) + "/"; //EL
SUBSTING ES PARA QUITAR EL DIR DE DELANTE DE LAS CARPETAS
                    //System.out.println("El directorio
actuar es: " + directorioActual);

                    laber_directorio.setText(directorioActual);

                    contador++;

                    //LO GUARDAMOS EN UN ARRAY PARA
LLEVAR EL SEGUIMIENTO DE LOS DIRECTORIOS RECORRIDOS POR EL
USUARIO

                    acumulador_directorios[contador] =
directorioActual;

                    try
                    {

                        //SE ESTABLECE EL DIRECTORIO
ACTUAL DE TRABAJO

                        cliente.changeWorkingDirectory(acumulador_directorios[conta
dor]);

                        //OBTENER DIRECTORIOS Y FICHEROS
files = cliente.listFiles();
llenarLista(files);

                        //-----
-----

```

```

        } catch (Exception e)
        {

System.out.println(e.getMessage());
        }
    }
}
    }
    catch (Exception e)

    {
        System.out.println(e.getMessage());
    }

}

@Override
public void mousePressed(MouseEvent e) {}

@Override
public void mouseReleased(MouseEvent e) {}

@Override
public void mouseEntered(MouseEvent e) {}

@Override
public void mouseExited(MouseEvent e) {}

@Override
public void actionPerformed(ActionEvent ae)
{

    // -----BOTONES DE ACCIÓN-----

    if (ae.getSource().equals(boton_crear_carpeta))
    {

        String nombreCarpeta =
JOptionPane.showInputDialog(null, "Introduce el nombre de la
carpeta nueva", "Nueva Carpeta");

        if (!(nombreCarpeta==null))
        {
            String directorio_a_crear =
directorioActual;

```

```

        directorio_a_crear +=
nombreCarpeta.trim()); //AÑADE EL NOMBRE DE LA CARPETA QUE ESTÁ
CREANDO A LA RUTA

        try
        {
            if
(nombreCarpeta.makeDirectory(directorio_a_crear))
            {
                String m = nombreCarpeta.trim()+
" => Se ha creado correctamente ...";

                JOptionPane.showMessageDialog(null, m);

                //REFRESCAR LISTA CON LA NUEVA
CARPETA

                files = cliente.listFiles();
                llenarLista(files);
                //-----
            }
            else

                JOptionPane.showMessageDialog(null, nombreCarpeta.trim() +
" => No se ha podido crear ...");
        }
        catch (IOException e1)
        {
            e1.printStackTrace();
        }
    }

    if (ae.getSource().equals(boton_eliminar_carpeta))
    {

        if (!lista.isSelectionEmpty())
        {
            String nombreCarpeta =
lista.getSelectedValue().toString();

            if
(nombreCarpeta.makeDirectory(nombreCarpeta)) //SI ENTRA AQUÍ
SIGNIFICA QUE ESTAMOS QUERIENDO ELIMINAR UNA CARPETA
            {

```

```

        String directorio_a_eliminar =
directorioActual + nombreCarpeta.substring(6); //EL SUBSTING ES
        PARA QUITAR EL DIR DE DELANTE DE LAS CARPETAS

        //System.out.println(directorio_a_eliminar);

        try
        {

            if(cliente.removeDirectory(directorio_a_eliminar))
            {
                String m =
                nombreCarpeta.trim()+" => Se ha eliminado correctamente ...";

                JOptionPane.showMessageDialog(null, m);

                //REFRESCAR LISTA CON LA
                NUEVA CARPETA

                files =
                cliente.listFiles();

                llenarLista(files);
                //-----

                //-----BORRAR
                TODAS LAS RUTAS A PARTIR DE LA CARPETA BORRADA-----
                //ESTO SE HACE PARA QUE NO
                DEN ERROR EL BOTON DE ADELANTE, UNA VEZ QUE SE BORRA UNA
                CARPETA

                for (int i = (contador +
                1); i < acumulador_directorios.length; i++)
                {

                    acumulador_directorios[i] = null;
                }

            }
            else
            {

                JOptionPane.showMessageDialog(null, nombreCarpeta.trim() +
                " => No se ha podido eliminar ...");
            }
        } catch (Exception e)
        {

```

```

        System.out.println(e.getMessage());
    }
    else
    {
        JOptionPane.showMessageDialog(this,
"*****OPCIÓN NO VÁLIDA*****");
    }
    else
    {
        JOptionPane.showMessageDialog(this,
"*****SELECCIONE UNA CARPETA*****");
    }
}

if (ae.getSource().equals(boton_renombrar_carpeta))
{
    if (!lista.isEmpty())
    {
        if
(comprueba_si_es_directorio(lista.getSelectedValue().toString())
)
        {
            String nombreNuevoCarpeta =
JOptionPane.showInputDialog(null,"Introduce el nombre de la
carpeta a renombrar","");

            String directorio_a_renombrar =
directorioActual +
lista.getSelectedValue().toString().substring(6); //EL SUBSTRING
ES PARA QUITAR EL DIR DE DELANTE DE LAS CARPETAS

            //System.out.println("El directorio a
modificar es: " + nuevoDirectorio);

            String nuevoDirectorio =
acumulador_directorios[contador] + nombreNuevoCarpeta;

            //System.out.println("El nuevo nombre
del directorio es: " + nuevoDirectorio);

            try
            {

```

```

        cliente.rename(directorio_a_renombrar, nuevoDirectorio);

        //REFRESCAR LISTA CON LA NUEVA
CARPETA

        files = cliente.listFiles();
        llenarLista(files);
        //-----

        //-----BORRAR TODAS
LAS RUTAS A PARTIR DE LA CARPETA EDITADA-----
        //ESTO SE HACE PARA QUE NO DEN
ERROR EL BOTON DE ADELANTE, UNA VEZ QUE SE EDITA UNA CARPETA
        for (int i = (contador+1); i <
acumulador_directorios.length; i++)
        {

            //System.out.println(acumulador_directorios[i]);
            acumulador_directorios[i]
= null;

        }
    } catch (Exception e)
    {

        System.out.println(e.getMessage());
    }
    }
    else
    {
        JOptionPane.showMessageDialog(this,
"*****OPCIÓN NO VÁLIDA*****");
    }
    }
    else
    {
        JOptionPane.showMessageDialog(this,
"*****SELECCIONE UNA CARPETA*****");
    }
}

if (ae.getSource().equals(boton_subir_fichero))
{

    JFileChooser f;
    File file;
    f = new JFileChooser();

```

```

        //solo se pueden seleccionar ficheros
        f.setFileSelectionMode(JFileChooser.FILES_ONLY);
        //título de la ventana
        f.setDialogTitle("Selecciona el fichero a subir
al servidor FTP");
        //se muestra la ventana
        int returnVal = f.showDialog(f, "Cargar");
        if (returnVal == JFileChooser.APPROVE_OPTION)
        {
            //fichero seleccionado
            file = f.getSelectedFile();
            //nombre completo del fichero
            String archivo = file.getAbsolutePath();
            //solo nombre del fichero
            String nombreArchivo = file.getName();
            try
            {
                SubirFichero(archivo, nombreArchivo);
            }
            catch (IOException e1)
            {
                e1.printStackTrace();
            }
        }
    }

    if (ae.getSource().equals(boton_bajar_fichero))
    {
        if (!lista.isEmpty())
        {
            if
(!comprueba_si_es_directorio(lista.getSelectedValue().toString()
))
            {
                //System.out.println("Es un
archivo");

                try
                {
                    String
directorio_fichero_a_descargar =
lista.getSelectedValue().toString();

                    //System.out.println("*EL
ARCHIVO A DESCARGAR ES: * " + directorio_fichero_a_descargar);

```



```

//System.out.println("LA RUTA
DEL ARCHIVO ES: * " + directorioActual+
directorio_fichero_a_descargar);

    DescargarFichero(directorioActual +
directorio_fichero_a_descargar ,
directorio_fichero_a_descargar);

        } catch (Exception e)
        {

            System.out.println(e.getMessage());
        }

        }
        else
        {
            JOptionPane.showMessageDialog(this,
"*****OPCIÓN NO VÁLIDA*****");
        }

        }
        else
        {
            JOptionPane.showMessageDialog(this,
"*****SELECCIONE UN FICHERO*****");
        }
    }

    if (ae.getSource().equals(boton_borrar_fichero))
    {

        if (!lista.isSelectionEmpty())
        {
            if
(!comprueba_si_es_directorio(lista.getSelectedValue().toString()
))
            {

                try
                {

                    String
directorio_fichero_a_descargar =
lista.getSelectedValue().toString();

```

```
                                //System.out.println("*EL
ARCHIVO A DESCARGAR ES: * " + directorio_fichero_a_descargar);

                                //System.out.println("LA RUTA
DEL ARCHIVO ES: * " + directorioActual+
directorio_fichero_a_descargar);

                                BorrarFichero(directorioActual +
directorio_fichero_a_descargar ,
directorio_fichero_a_descargar);

                                } catch (Exception e)
                                {

System.out.println(e.getMessage());
                                }

                                }
                                else
                                {
                                    JOptionPane.showMessageDialog(this,
"*****OPCIÓN NO VÁLIDA*****");
                                }
                                }else
                                {
                                    JOptionPane.showMessageDialog(this,
"*****SELECCIONE UN FICHERO*****");
                                }
                                }

                                if (ae.getSource().equals(boton_renombrar_fichero))
                                {

                                    if (!lista.isEmpty())
                                    {
                                        if
(!comprueba_si_es_directorio(lista.getSelectedValue().toString()
))
                                        {

                                            String nombreNuevoFichero =
JOptionPane.showInputDialog(null,"Introduce el nuevo nombre del
fichero","");
```

```

        //System.out.println("*EL ARCHIVO A
DESCARGAR ES: * " + directorio_fichero_a_descargar);

        //System.out.println("LA RUTA DEL
ARCHIVO ES: * " + directorioActual+
directorio_fichero_a_descargar);

        try
        {
            cliente.rename(directorioActual
+ lista.getSelectedValue().toString(), directorioActual +
nombreNuevoFichero);

            //REFRESCAR LISTA CON LA NUEVA
CARPETA

            files = cliente.listFiles();
            llenarLista(files);
            //-----

            //-----BORRAR TODAS
LAS RUTAS A PARTIR DE LA CARPETA EDITADA-----
            //ESTO SE HACE PARA QUE NO DEN
ERROR EL BOTON DE ADELANTE, UNA VEZ QUE SE EDITA UNA CARPETA
            for (int i = (contador+1); i <
acumulador_directorios.length; i++)
            {

                //System.out.println(acumulador_directorios[i]);
                acumulador_directorios[i]
= null;

            }
        } catch (Exception e)
        {

            System.out.println(e.getMessage());

        }
        else
        {
            JOptionPane.showMessageDialog(this,
"*****OPCIÓN NO VÁLIDA*****");
        }
    }
    else
    {

```

```

                                JOptionPane.showMessageDialog(this,
"*****SELECCIONE UN FICHERO*****");
                                }
                            }

// -----BOTONES DE CABECERA-----
-----

        if (ae.getSource().equals(boton_volver))
        {

            try
            {
                if (contador > 0)
                {
                    contador--;

                    //System.out.println(acumulador_directorios[contador]);
                    directorioActual=
acumulador_directorios[contador];

                    laber_directorio.setText(directorioActual);
                    //SE ESTABLECE EL DIRECTORIO DE
TRABAJO ACTUAL

                    cliente.changeWorkingDirectory(acumulador_directorios[conta
dor]);

                    //OBTENIENDO FICHEROS Y CARPETAS DEL
DIRECTORIO

                    files = cliente.listFiles();
                    llenarLista(files);
                    //-----
-----

                                }

                                } catch (Exception e)
                                {
                                    System.out.println(e.getMessage());
                                }
                            }else
                            {

                                if (ae.getSource().equals(boton_adelantar))
                                {

```

```

        try
        {
            if
            (acumulador_directorios[contador+1] != null)
            {
                contador++;

                //System.out.println(acumulador_directorios[contador]);
                directorioActual =
                acumulador_directorios[contador];

                laber_directorio.setText(directorioActual);
                //SE ESTABLECE EL DIRECTORIO DE
                TRABAJO ACTUAL

                cliente.changeWorkingDirectory(acumulador_directorios[conta
                dor]);

                //OBTENIENDO FICHEROS Y CARPETAS
                DEL DIRECTORIO

                files = cliente.listFiles();
                llenarLista(files);
                //-----
            }
        } catch (Exception e)
        {
            System.out.println(e.getMessage());
        }

    }
    else
    {
        if
        (ae.getSource().equals(boton_refrescar))
        {
            try
            {
                //OBTENIENDO FICHEROS Y CARPETAS
                DEL DIRECTORIO

                files = cliente.listFiles();
                llenarLista(files);
                //-----
            }

```

```

        } catch (Exception e)
        {

            System.out.println(e.getMessage());
        }
    }
    else
    {
        if
        (ae.getSource().equals(boton_home))
        {

            try
            {
                acumulador_directorios =
                acumulador_directorios[0]
                contador=0; //REINICIAMOS
                directorioActual =
                directorioInicial; //EL DIRECTORIO ACTUAL PASA A SER LA CARPETA
                RAÍZ

                laber_directorio.setText(directorioActual);

                cliente.changeWorkingDirectory(directorioInicial); //SE
                ESTABLECE EL DIRECTORIO ACTUAL DE TRABAJO

                //OBTIENE LOS FICHEROS Y
                ARCHIVOS DEL DIRECTORIO
                files =
                cliente.listFiles();

                llenarLista(files); //ACTUALIZA LA LISTA
                //-----
                -----

            } catch (Exception e)
            {

                System.out.println(e.getMessage());
            }
        }
    }
}

```

```
    }  
  }  
}  
  
///-----  
-----  
/*  
*  
* LA APLICACIÓN SE DESCONECTA CADA POCO TIEMPO DEL SERVIDOR  
FTP, ESTO ES DEBIDO A MEDIDAS DE SEGURIDAD  
* NO ES UN ERROR DEL PROGRAMA. MENSAJE DE ERROR: Software  
caused connection abort: recv failed  
*  
*  
*  
*/
```