Instituto Politécnico Nacional.

Escuela Superior De Cómputo.

Materia:

Aplicaciones Para Comunicación En Red.

Tema:

Practica 01

(Reporte)

Profesor:

Axel Ernesto Moreno Cervantes.

Alumno:

Luis Enrique Rojas Alvarado.

Mario Alberto Miranda Sandoval.

Grupo:

3CM5

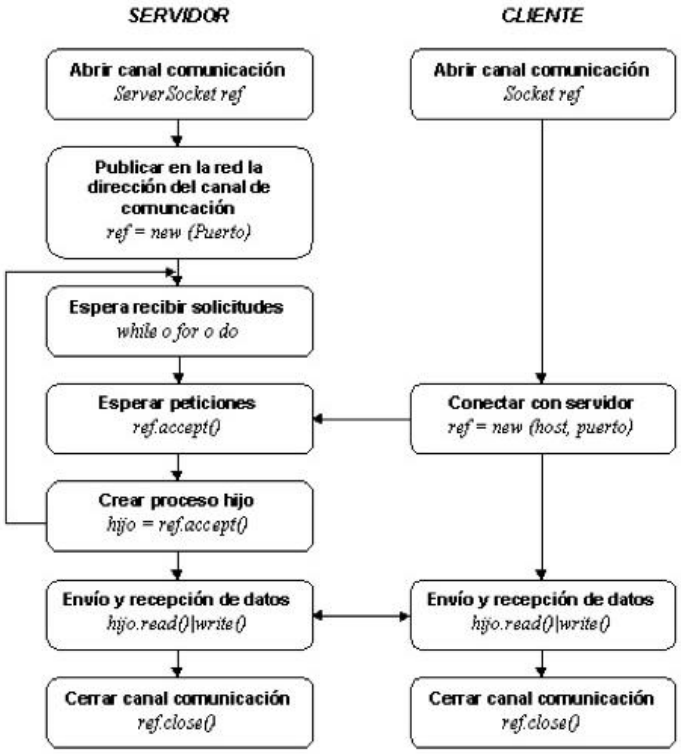
Introducción.

En la actualidad, muchos de los procesos que se ejecutan en una computadora requieren obtener o enviar información a otros procesos que se localizan en una computadora diferente. Para lograr esta comunicación se utilizan los protocolos de comunicación TCP y UDP.

El protocolo TCP (Transmission Control Protocol) establece un conducto de comunicación punto a punto entre dos computadoras, es decir, cuando se requiere la transmisión de un flujo de datos entre dos equipos, el protocolo TCP establece un conducto exclusivo entre dichos equipos por el cual los datos serán transmitidos y este perdurará hasta que la transmisión haya finalizado, gracias a esto TCP garantiza que los datos enviados de un extremo de la conexión lleguen al otro extremo y en el mismo orden en que fueron enviados. Las características que posee TCP hacen que el protocolo sea conocido como un protocolo orientado a conexión.

Los sockets son una forma de comunicación entre procesos que se encuentran en diferentes máquinas de una red, los sockets proporcionan un punto de comunicación por el cual se puede enviar o recibir información entre procesos.

Los sockets tienen un ciclo de vida dependiendo si son sockets de servidor, que esperan a un cliente para establecer una comunicación, o socket cliente que busca a un socket de servidor para establecer la comunicación.

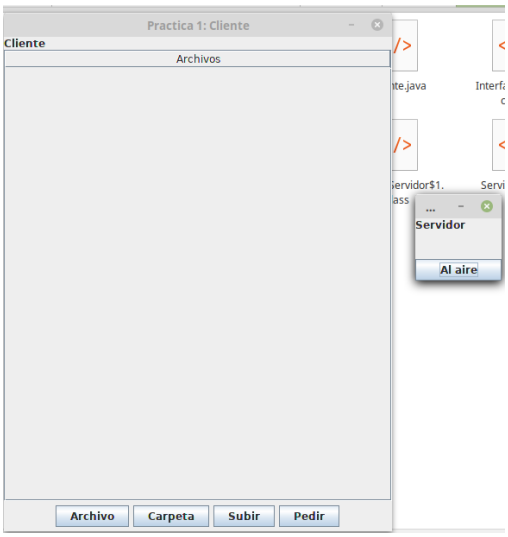


Desarrollo.

El desarrollo de esta práctica consta de, dos interfaces una de servidor y otra de cliente y de distintas clases que se encargan de gestionar los envíos y peticiones de archivos.

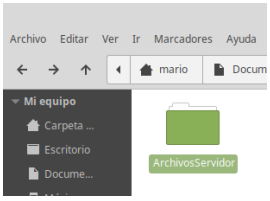
Además, que se creó una carpeta para el servidor a modo que esta sirva para guardar todos los archivos que se le son mandados a este servidor.

La separación de las clases genera que se llamen las funciones y la creación de los sockets y destrucción de los sockets en cada envió, donde destaca la clase archivo, para la gestión de estos.

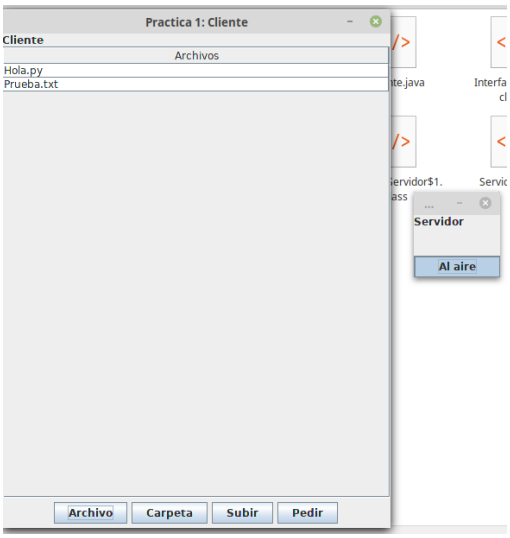


Interfaz de cliente y servidor.

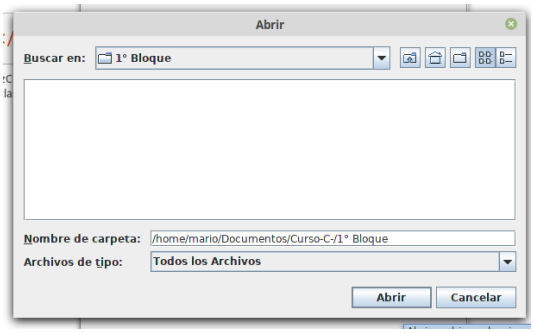
Pruebas.



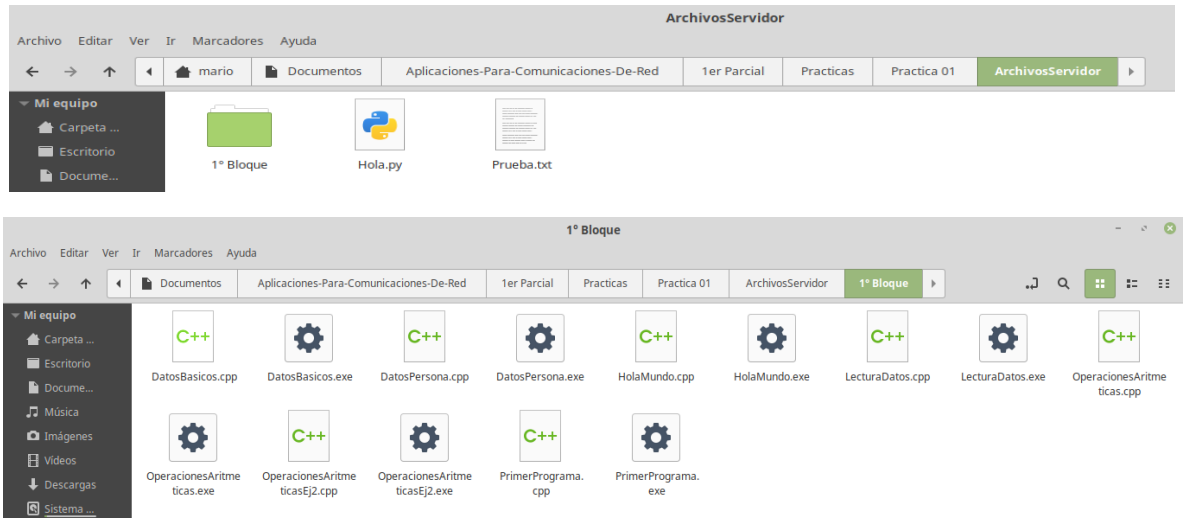
Carpeta del servidor para guardar los archivos.



Enviando múltiples archivos por separado.

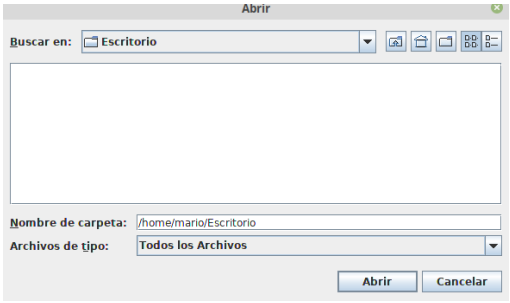


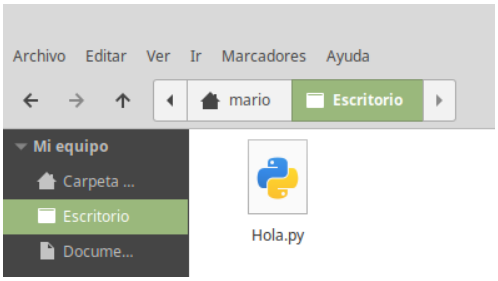
Mandando una carpeta con su contenido.



Archivos recibidos en el servidor.

Pidiendo un archivo, el primer cuadro muestra el archivo a ser pedido que se encuentra el servidor, mientras que el segundo muestra al usuario donde escoger la ubicación para guardar su archivo.





Como se puede ver se seleccionó el destino en el escritorio, y el archivo ha llegado a su destino.

Código.

Archivo.java

```java
import java.io.File;

public class Archivo {
    public Archivo(String nombre, long tamanio, String path, File file) {
        this.nombre = nombre;
        this.tamanio = tamanio;
        this.path = path;
        this.file = file;
    }

    public String getNombre() { return nombre; }
    public long getTamanio() { return tamanio; }
    public String getPath() { return path; }
    public File getFile() { return file; }

    public void setNombre(String nombre) { this.nombre = nombre; }
    public void setTamanio(long tamanio) { this.tamanio = tamanio; }
    public void setPath(String path) { this.path = path; }
    public void setFile(File file) { this.file = file; }

    private String nombre;
    private long tamanio;
    private String path;
    private File file;
}
```

InterfazCliente.java

```java
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.table.*;
import java.io.*;
import java.util.*;

public class InterfazCliente extends JFrame {
    private static final long serialVersionUID = 1L;
    public InterfazCliente() {
        setTitle("Practica 1: Cliente");
        setBounds(300, 100, 450, 600);
        setResizable(false);

        panelPrincipal = new JPanel();
        panelInferior = new JPanel();
        cliente = new JLabel("Cliente");
        elegirArchivo = new JButton("Archivo");
        elegirCarpeta = new JButton("Carpeta");
        subirArchivo = new JButton("Subir");
        pedirArchivo = new JButton("Pedir");
        tablaCliente = new JTable();
        misArchivos = new ArrayList<>();
        modelo = (DefaultTableModel) tablaCliente.getModel();
        modelo.addColumn("Archivos");

```

```
27.         elegirArchivo.addActionListener(new ActionListener() {
28.             public void actionPerformed(ActionEvent e) {
29.                 file = new JFileChooser();
30.                 file.requestFocus();
31.                 int r = file.showOpenDialog(InterfazCliente.this);
32.                 if(r == JFileChooser.APPROVE_OPTION) {
33.                     File f = file.getSelectedFile();
34.                     misArchivos.add(new Archivo(f.getName(), f.length(), f.getAbso
    lutePath(), f));
35.                     modelo.addRow(new Object[] { f.getName() });
36.                 }
37.             }
38.         });
39.
40.         subirArchivo.addActionListener(new ActionListener() {
41.             public void actionPerformed(ActionEvent e) {
42.                 miCliente = new Cliente(HOST, PUERTO);
43.                 miCliente.enviarArchivo(misArchivos, "");
44.                 modelo.setRowCount(0);
45.             }
46.         });
47.
48.         elegirCarpeta.addActionListener(new ActionListener() {
49.             public void actionPerformed(ActionEvent e) {
50.                 JFileChooser jf = new JFileChooser();
51.                 jf.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
52.                 jf.requestFocus();
53.                 int r = jf.showOpenDialog(InterfazCliente.this);
54.                 if (r == JFileChooser.APPROVE_OPTION) {
55.                     File archivoSeleccionado = jf.getSelectedFile();
56.                     modelo.addRow(new Object[]{"Carpeta: " + archivoSeleccionado.g
    etName()});
57.                     modelo.setRowCount(0);
58.                     miCliente = new Cliente(HOST, PUERTO);
59.                     miCliente.carpetas(archivoSeleccionado, "", misArchivos);
60.                 }
61.             }
62.         });
63.
64.         pedirArchivo.addActionListener(new ActionListener() {
65.             public void actionPerformed(ActionEvent e) {
66.                 JFileChooser jf = new JFileChooser("./ArchivosServidor/");
67.                 jf.requestFocus();
68.                 int r = jf.showOpenDialog(InterfazCliente.this);
69.                 if (r == JFileChooser.APPROVE_OPTION) {
70.                     JFileChooser jf2 = new JFileChooser();
71.                     jf2.requestFocus();
72.                     jf2.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
73.                     int r2 = jf2.showOpenDialog(InterfazCliente.this);
74.                     if(r2 == JFileChooser.APPROVE_OPTION) {
75.                         File f = jf.getSelectedFile();
76.                         File f2 = jf2.getSelectedFile();
77.                         Archivo a = new Archivo(f.getName(), f.length(), f.getAbso
    lutePath(), f);
78.                         miCliente = new Cliente(HOST, PUERTO);
79.                         miCliente.peticionArchivo(a, f2.getAbsolutePath());
80.                     }
81.                 }
82.             }
83.         });
84.
```

```
85.         panelInferior.add(elegirArchivo);
86.         panelInferior.add(elegirCarpeta);
87.         panelInferior.add(subirArchivo);
88.         panelInferior.add(pedirArchivo);
89.         panelPrincipal.setLayout(new BorderLayout());
90.         panelPrincipal.add(cliente, BorderLayout.NORTH);
91.         panelPrincipal.add(new JScrollPane(tablaCliente), BorderLayout.CENTER);
92.         panelPrincipal.add(panelInferior, BorderLayout.SOUTH);
93.         add(panelPrincipal);
94.
95.         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
96.         setVisible(true);
97.     }
98.
99.     public static void main(String[] args) {
100.             new InterfazCliente();
101.         }
102.
103.         private JPanel panelPrincipal;
104.         private JLabel cliente;
105.         private JTable tablaCliente;
106.         private DefaultTableModel modelo;
107.         private JPanel panelInferior;
108.         private JButton elegirArchivo;
109.         private JButton subirArchivo;
110.         private JButton elegirCarpeta;
111.         private JButton pedirArchivo;
112.         private JFileChooser file;
113.         private ArrayList <Archivo> misArchivos;
114.         private Cliente miCliente;
115.         private final int PUERTO = 9000;
116.         private final String HOST = "127.0.0.1";
117.     }
```

InterfazServidor.java

```
1.  import java.awt.*;
2.  import java.awt.event.ActionEvent;
3.  import java.awt.event.ActionListener;
4.
5.  import javax.swing.*;
6.
7.  public class InterfazServidor extends JFrame{
8.      private static final long serialVersionUID = 2L;
9.      public InterfazServidor() {
10.         setBounds(600, 100, 100, 100);
11.         setTitle("Practica 1: Servidor");
12.         setResizable(false);
13.
14.         panelPrincipal = new JPanel();
15.         servidor = new JLabel("Servidor");
16.         conectar = new JButton("Al aire");
17.
18.         conectar.addActionListener(new ActionListener() {
19.             public void actionPerformed(ActionEvent arg0) {
20.                 miServidor = new Servidor(PUERTO);
21.                 miServidor.conectar();
22.             }
23.         });
24.
```

```
        panelPrincipal.setLayout(new BorderLayout());
        panelPrincipal.add(servidor, BorderLayout.NORTH);
        panelPrincipal.add(conectar, BorderLayout.SOUTH);
        add(panelPrincipal);

        setDefaultCloseOperation(3);
        setVisible(true);
    }

    public static void main(String[] args) {
        new InterfazServidor();
    }

    private JPanel panelPrincipal;
    private JLabel servidor;
    private JButton conectar;
    private Servidor miServidor;
    private final int PUERTO = 9000;
}
```

Cliente.java

```
import java.io.*;
import java.net.*;
import java.util.*;

public class Cliente {
    public Cliente(String host, int puerto){
        this.host = host;
        this.puerto = puerto;
    }

    public void enviarArchivo(ArrayList <Archivo> misArchivos, String destino) {
        try {
            for(Archivo a : misArchivos) {
                File f = a.getFile();
                cliente = new Socket(this.host, this.puerto);
                dis = new DataInputStream(new FileInputStream(f.getAbsoluteFile())
);
                dos = new DataOutputStream(cliente.getOutputStream());

                long e = 0;
                int n = 0;
                int porcentaje = 0;

                dos.writeInt(0);
                dos.flush();
                dos.writeUTF(a.getNombre());
                dos.flush();
                dos.writeLong(a.getTamanio());
                dos.flush();
                dos.writeUTF(destino);
                dos.flush();

                while (e < a.getTamanio()) {
                    byte[] b = new byte[2000];
                    n = dis.read(b);
                    e += n;
                    dos.write(b, 0, n);
                    dos.flush();
```

```
                    porcentaje = (int) ((e * 100) / a.getTamanio());
                    System.out.print("\rPorcentaje:" + porcentaje + "%");
                }

                System.out.println("Archivo Enviado");
                dos.close();
                dis.close();
                cliente.close();
            }
            misArchivos.clear();
        } catch(Exception e) { e.printStackTrace(); };
    }

    public void carpetas(File carpeta, String destino, ArrayList<Archivo> misArchivos) {
        if(destino.equals(""))
            destino = carpeta.getName();
        else
            destino = destino + "\\" + carpeta.getName();

        for(File archivo : carpeta.listFiles()) {
            if(archivo.isDirectory())
                carpetas(carpeta, destino, misArchivos);
            else{
                misArchivos.add(new Archivo(archivo.getName(), archivo.length(), destino, archivo));
            }
        }
        enviarArchivo(misArchivos ,destino);
    }

    public void peticionArchivo(Archivo archivoPedido, String destino) {
        try {
            cliente = new Socket(this.host, this.puerto);
            dos = new DataOutputStream(cliente.getOutputStream());
            dos.writeInt(1);
            dos.flush();
            dos.writeUTF(archivoPedido.getNombre());
            dos.flush();
            dos.writeLong(archivoPedido.getTamanio());
            dos.flush();
            dos.writeUTF(archivoPedido.getPath());
            dos.flush();

            System.out.println(destino);
            dis = new DataInputStream(cliente.getInputStream());
            String nombre = dis.readUTF();
            long tam = dis.readLong();
            String ruta = dis.readUTF();

            long r = 0;
            int n = 0;
            int porcentaje = 0;
            DataOutputStream archivo = new DataOutputStream(new FileOutputStream(destino + "/" + nombre));

            while(r < tam) {
                byte[] b = new byte[2000];
                n = dis.read(b);
                r += n;
                archivo.write(b, 0, n);
```

```
                archivo.flush();
                porcentaje = (int) ((r*100) / tam);
                System.out.println("Se ha recibido " + porcentaje + "%");
            }

                System.out.println("Se ha recicibido el archivo");

                archivo.close();
                dos.close();
                dis.close();
            } catch (Exception e) { e.printStackTrace(); }
        }

        private String host;
        private int puerto;
        private Socket cliente;
        private DataInputStream dis;
        private DataOutputStream dos;
    }
```

Servidor.java

```
import java.io.*;
import java.net.*;

public class Servidor {
    public Servidor(int puerto) {
        this.puerto = puerto;
    }

    public void conectar() {
        try {
            servidor = new ServerSocket(puerto);
            servidor.setReuseAddress(true);
            System.out.println("Servidor establecido");
            for(;;) {
                cliente = servidor.accept();
                DataInputStream dis = new DataInputStream(cliente.getInputStream()
);
                int accion = dis.readInt();
                String nombre = dis.readUTF();
                long tam = dis.readLong();
                String ruta = dis.readUTF();

                if (accion == 0) {
                    nombre = directorio(nombre, ruta);
                    nombre = "./ArchivosServidor/" + nombre;
                    recibirArchivo(nombre, tam, dis);
                } else if(accion == 1) {
                    enviaArchivoPedido(nombre, tam, ruta);
                }

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

    public void recibirArchivo(String nombre, long tam, DataInputStream dis) {
        try {
            int porcentaje = 0;
```

```
38.                 long r = 0;
39.                 int n = 0;
40.                 DataOutputStream dos = new DataOutputStream(new FileOutputStream(n
    ombre));
41.                 while(r < tam) {
42.                     byte[] b = new byte[2000];
43.                     n = dis.read(b);
44.                     r += n;
45.                     dos.write(b, 0, n);
46.                     dos.flush();
47.                     porcentaje = (int) ((r*100) / tam);
48.                     System.out.print("\rSe ha recibido " + porcentaje + "%");
49.                 }
50.
51.                 dos.close();
52.         } catch(Exception e) { e.printStackTrace(); }
53.     }
54.
55.     public String directorio(String nombre, String ruta) {
56.         if(!ruta.equals("")) {
57.             File carpeta = new File("./ArchivosServidor/" + ruta);
58.             if(!carpeta.exists()) {
59.                 try {
60.                     if (carpeta.mkdir())
61.                         System.out.println("Carpeta creada");
62.                     else
63.                         System.out.println("No se creo la carpeta");
64.                 }catch(SecurityException se) { se.printStackTrace(); }
65.             }
66.             nombre = ruta + "//" + nombre;
67.         }
68.         return nombre;
69.     }
70.
71.     public void enviaArchivoPedido(String nombre, long tam, String ruta) {
72.         //System.out.println("Todo bien");
73.         try {
74.             int porcentaje = 0;
75.             long r = 0;
76.             int n = 0;
77.             DataInputStream dis = new DataInputStream(new FileInputStream(ruta));

78.             DataOutputStream dos = new DataOutputStream(cliente.getOutputStream())
    ;
79.
80.             dos.writeUTF(nombre);
81.             dos.flush();
82.             dos.writeLong(tam);
83.             dos.flush();
84.             dos.writeUTF(ruta);
85.             dos.flush();
86.
87.             while(r < tam) {
88.                 byte[] b = new byte[2000];
89.                 n = dis.read(b);
90.                 r += n;
91.                 dos.write(b, 0, n);
92.                 dos.flush();
93.                 porcentaje = (int) ((r*100) / tam);
94.                 System.out.print("\rPorcentaje" + porcentaje + "%");
95.             }
```

```
96.             System.out.println("");
97.             dos.close();
98.             dis.close();
99.          } catch(Exception e) { e.printStackTrace(); }
100.         }
101.
102.         private int puerto;
103.         private ServerSocket servidor;
104.         private Socket cliente;
105.     }
```

Dificultades Encontradas.

La mayor dificultad fue pedir los archivos y su ruta donde se almacenarían, ya que no encontrábamos los métodos necesarios para poder manejar esas ultimas partes, de ahí en fuera todo normal.

Conclusiones.

Luis Enrique Rojas Alvarado.

Este programa es muy interesante debido a que es algo con lo que tenemos contacto todos los días, por ejemplo, para hacer algún programa o reporte de práctica entre compañeros porque nos compartimos los archivos por medio de Facebook o alguna otra aplicación.

La parte de la comunicación fue bastante simple debido a que se explicó muy bien dentro del curso en el salón y se resolvieron todas las dudas por parte del profesor, sin duda, una práctica con un nivel de complejidad bastante aceptable y acorde a lo visto hasta el momento.

Mario Alberto Miranda Sandoval.

Acerca de esta aplicación, fue algo fácil e interesante hacerla, ya que ahora comprendo mejor el envío de archivos mediante sockets de flujo, además de como el modo de hacerlo enviando por partes dentro del DataOutputStream y el buffer mandado.

Sobre las peticiones al servidor, fue relativamente fácil ya que, con el mandado de la información del archivo o carpeta, se mandaba una pequeña bandera a modo de descriptor, el único inconveniente fue el manejar la ruta elegida del archivo, ya que desconocíamos que el método getSelectedFile(), funcionaba para carpetas y si, ya después de recibir el archivo seleccionado con el método getAbsolutePath() fue suficiente para poder terminar la práctica.