**Juan Camilo Restrepo Velez 000373886**

**Prototipo (Back-end)**

El prototipo consiste en desarrollar una aplicación para realizar las pruebas de rendimiento correspondientes a la conexión con el nodo Raspberri tanto para descargar archivos como para subir archivos a un servidor ftp que se encontrarán en el nodo.

En este sentido, la aplicación necesitará conectarse a un servidor ftp para lo cual se requiere de su Ip, usuario y contraseña, luego de establecer dicha conexión deberá poder subir un archivo que se elija en el dispositivo móvil y también poder elegir un archivo del servidor y descargarlo en el dispositivo.

Ahora bien, para ello se hará uso de la librería ***Apache commons* net**, la cual nos permite realizar todas estas funcionalidades en código Java, debido que la aplicación se desarrollará en Android Studio, dicha librería se encuentra en: <http://commons.apache.org/proper/commons-net/download_net.cgi>

**Implementación**

Luego de descargar la librería se debe poner en la carpeta “libs” del proyecto y darle click derecho y “add lib”, de este modo ya puede realizar el import correspondiente asi

import org.apache.commons.net.ftp.FTPClient;

Para tener todo en un menor grado de acoplamiento y mayor legibilidad del proyecto se creará una clase java denominada ConexionFTP.java que se encargará de todo el manejo de la librería previamente mencionada. Esta clase tendrá los atributos necesarios para realizar la conexión además de los métodos para gestionar la conexión, desconexión, descarga y subida de archivos.

**Conectar:** Se encarga de la conexión para lo cual se debe llamar al método .connect(Ftp,Port) ; del objeto FTPClient de la librería, enviando como parámetros la ip y el puerto

**Desconectar:** Que realiza la desconexión llamando al método .disconnect(); del objeto FTPClient .

**Descarga:** Para esta función se debe se recibe por parámetro tres variables tipo string, una para la ruta del archivo a descargar que está alojado en el servidor (remote), la otra con la ruta donde se va a guardar localmente en el dispositivo móvil (local) y el nombre con el cual se guardará local (name), para ello se utilizará el método .retrieveFile(remote,out); del objeto FTPClient donde la variable out corresponde a un objeto de la clase FileOutputStream(local+”/”+name); que en el constructor recibe la variable local concatenada con name, que quiere decir la ruta y el nombre del archivo a guardar.

**Carga:** Brinda la posibilidad de subir archivo para lo cual se debe especificar, por medio del objeto FTPClient , el tipo de archivo por el método .setFileType(FTP. *BINARY\_FILE\_TYPE*) ; y se especifica el modo pasivo con el método .enterLocalPassiveMode(); . Luego, se debe crear un objeto FileInputStream cuyo constructor recibe la ruta del archivo y por último llamar el método .storeFile(file.getName(),fis); del objeto FTPClient que recibe el nombre del archivo y el objeto previamente creado.

**Reporte:** Para la función de Descargar y Carga se debe generar un reporte de estas dos operaciones, desde las cuales se llama al método public void Reporte(String[] args) que recibe el un vector que contiene el nombre del archivo a escribir (args[0]) y el cuerpo del archivo (las demás posiciones), donde cada posición del vector es una línea (si el archivo existe, escribe desde la última línea, es decir que no sobre escribe). Los parámetros son calculados en los métodos Carga y Descarga

**Activity**

Luego de tener toda la clase lista se procede a realizar el llamado desde la aplicación. En este sentido, habrá una parte destinada a diligenciar los datos del servidor (Ip, Usuario y Contraseña) que se realizan en la ventana ***activity\_tools.xml*** la tiene su manejo lógico en la clase ***ToolsActivity.java***, donde se creará un objeto de la clase ConexionFTP que al constructor se le pasarán todos estos parámetros correspondientes y se llamará al método Conectar() mediante el objeto creado para validar los datos.

Después de realizar esta configuración se dispondrán de un botón (en la actividad principal cuya vista es ***activity\_main.xml***y su controlador es ***MainActivity.java***) para subir archivos, el cual llamará al "***File Chooser/File picker***" nativo de Android para elegir un archivo que se encuentre en el dispositivo, luego se invoca un método estático de una clase de biblioteca (GetPathUtil) que se implementó para encontrar la ruta del archivo que eligió, después, se llama al método Cargar() del objeto al cual se le enviará el un objeto File que tiene el path del archivo elegido con anterioridad, al terminar la función se llama al método Desconectar() para no gastar recursos.

Para la parte de descargar se llenan las variables remote (ruta del archivo en el servidor FTP), local (ruta del directorio a guardar localmente) y name (nombre con el cual se guardará el archivo) que son las que se enviarán como parámetro a la funcion Descargar() de objeto ConexionFTP para realizar la descarga del documento especificado en la variable remote y guárdalo en la ubicación especificada en local con el nombre name.

**NOTA:**

Se debe agregar en el AndroidManifest.xml las siguientes líneas:

<uses-permission android:name="android.permission.INTERNET"/>  
<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"/>  
<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"/>

Esto para permitir el acceso a Internet, permitir la lectura y escritura de archivos con las líneas del manifest.

En el método onCreate de la clase ***MainActivity.java*** que maneja la vista principal la siguiente línea:

StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder().permitNetwork().build());

Para permitir acceder a la red desde el hilo principal ya que desde el API 9 se verifica que esto no se haga porque lo recomendado es acceder a la red desde un hilo hijo.

Y crear un método denomino permiso() que contiene el siguiente código:

if (android.os.Build.VERSION.*SDK\_INT* >= android.os.Build.VERSION\_CODES.*M*){  
 if (checkSelfPermission(Manifest.permission.*READ\_EXTERNAL\_STORAGE*) == PackageManager.*PERMISSION\_DENIED* ||  
 checkSelfPermission(Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*) == PackageManager.*PERMISSION\_DENIED* ) {  
 String[] permissions = {Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*,Manifest.permission.*READ\_EXTERNAL\_STORAGE*};  
 requestPermissions(permissions,1);  
 }  
}

Para pedir los permisos (en tiempo de ejecución) que se establecieron en cuanto a la lectura y escritura de archivos, el cual es requerido para los métodos Cargar() y Descargar()

**Referencias**

Máster en Desarrollo de Aplicaciones Android. Un ejemplo de un cliente / servidor de ECHO. Androidcurso.com. (2020). Tomado de (19/03/2020): <http://www.androidcurso.com/index.php/recursos/43-unidad-10-internet-sockets-http-y-servicios-web/322-un-ejemplo-de-un-cliente-servidor-de-echo>

Cliente FTP en Java. Cursos de Programación de 0 a Experto. (2020). Tomado de (19/03/2020): <https://unipython.com/cliente-ftp-en-java/>

Mostrar Cuadro de Diálogo para selecionar un archivo o ruta en Android. Stack Overflow en español. Tomado de (19/03/2020): <https://es.stackoverflow.com/questions/15481/mostrar-cuadro-de-diálogo-para-selecionar-un-archivo-o-ruta-en-android>

Apache Commons Net 3.6 API. Commons.apache.org. (2020). Tomado de (19/03/2020): <http://commons.apache.org/proper/commons-net/javadocs/api-3.6/index.html>

Commons.apache.org. (2020). Tomado de (19/03/2020): <http://commons.apache.org/proper/commons-net/examples/ftp/FTPClientExample.java>

FTPClient (Apache Commons Net 3.6 API). Commons.apache.org. (2020). Tomado de (19/03/2020): <https://commons.apache.org/proper/commons-net/apidocs/org/apache/commons/net/ftp/FTPClient.html>

Implementación de un selector de archivos en Android y copia del archivo seleccionado en otra ubicación Flip Android. Flipandroid.com. (2020). Tomado de (20/03/2020): <https://www.flipandroid.com/implementacin-de-un-selector-de-archivos-en-android-y-copia-del-archivo-seleccionado-en-otra-ubicacin.html>

Android permission doesn't work even if I have declared it. Stack Overflow. (2020). Tomado de (20/03/2020): <https://stackoverflow.com/questions/32635704/android-permission-doesnt-work-even-if-i-have-declared-it>

Implementación de métodos muy completos y sencillos: <http://www.myjavazone.com/2011/12/primera-entrega-de-java-zone-cliente.html>