**CSE 5345**

**Project Report**

**Project 2: Interfacing a wireless device with a computer**

**Team 10**

Sandeep Yerramsetti (1001156902)

Dileep Kumar Budavati (1001244113)

Pranav Teja Garikapati (1001276676)

INDEX

1. Project Description 3

2. Overview 3

3. project implementation 4

4. Implementation overview 8

5. Flow Diagram 10

6. Responsibilities 10

7. References 11

1. **Project Description:**

Implement an application that interacts with a computer wirelessly. Either the mobile or the computer should initiate a connection to the other device.

**Requirements:**

a. The user should be able to initiate the connection from the mobile device or the computer.

b. If the mobile device initiates the connection, a list of files on the computer should be displayed on the mobile device and vice versa.

c. The user should be able to view, download and upload files from the mobile device to the computer and vice versa.

d. Progress should be shown on the either end of the download.

e. Language/ Framework: Android for the mobile application and Python/Java for the desktop interface. Interfacing on the computer can be done via application or a web-browser.

1. **Requirements Overview**

* Software Required: Android Studio, Eclipse
* Hardware Required:
  + Android Device of minimum API of 2.1
  + Computer with minimum of dual core CPU, 4 GB RAM, 20 GB of disk space
* Technology Used: Android 5.0 (Lollipop), Java
* Framework android 2.1
* Wi-Fi network

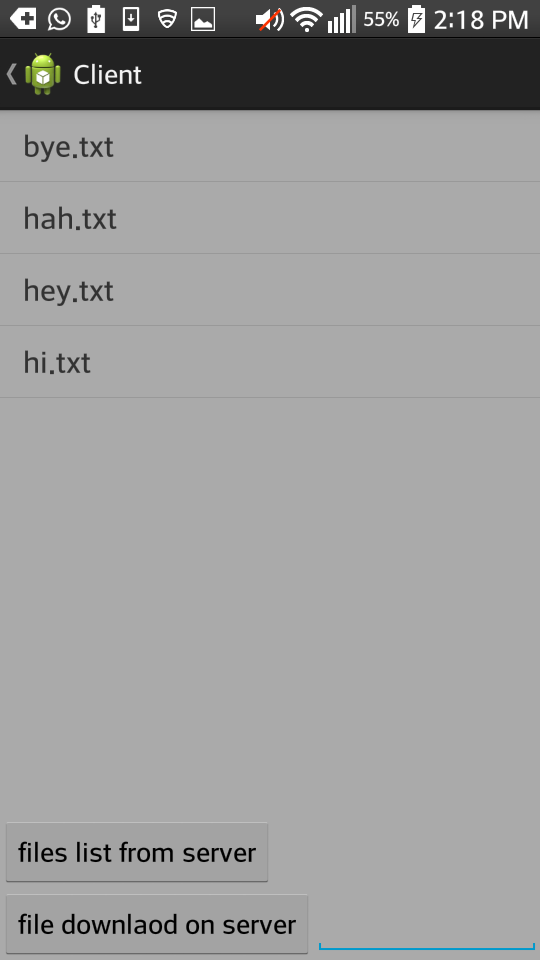
1. **Project Implementation**

There are two phases in interfacing the wireless devices and transferring data between them

1. Desktop Server and Android Client
2. Android Server and Desktop Client
   1. **Desktop as Sever and Android as Client**

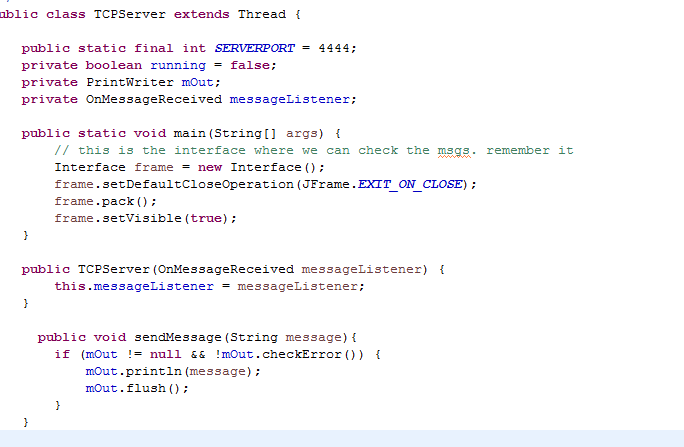
*S****teps:***

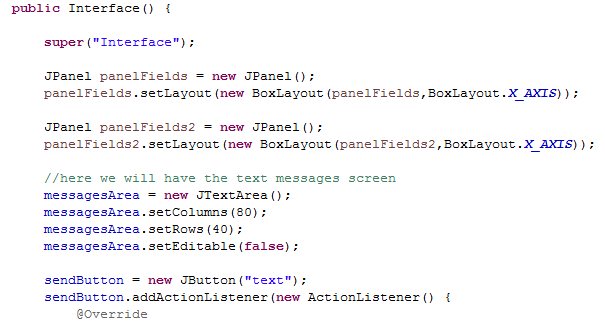
1. Download and unzip the project folder and select the TCPServer.java and interface.java import the files onto eclipse workspace.
2. Run the TCPServer.java
3. We need to make a folder by the name: test in the folder C:\Users\Rishi\Desktop\sandy\Server , which contains a list of files the client would be able to see and request for download and also would contain the list of files which the client has uploaded to server.
4. Download and unzip the **File transfer** file and import this onto Android Studio.
5. In the mobile device, we need to make sure that we have a download folder by the name: Client Downloads in the path: ExternalStroragePublicDirctory. This folder would contain the downloaded files.
6. Run MultiThreadServer.java
7. Also, we need another folder with the name: Client Uploads in ExternalStroragePublicDirctory. which would consist of a list of files that the client would want to upload to the server.
8. To run, copy the .apk file and install it and run the application.

THE APPLICATION UI:



THE CODE SNIPPETS:

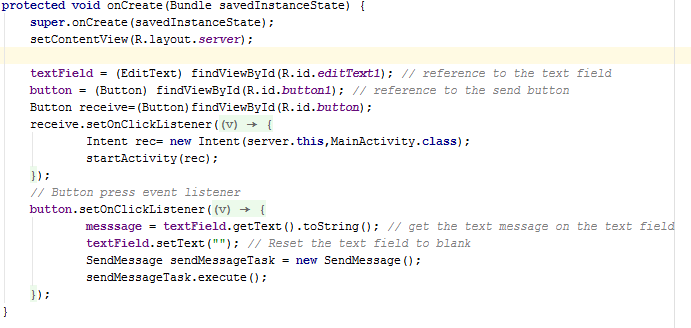


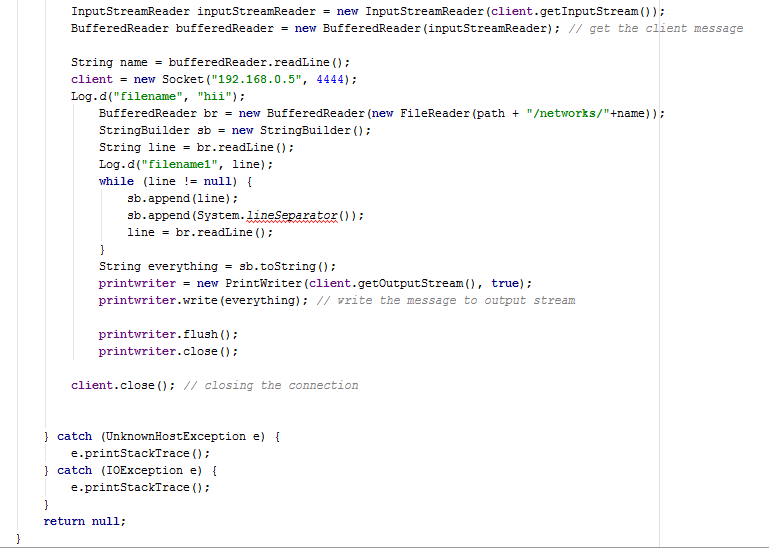


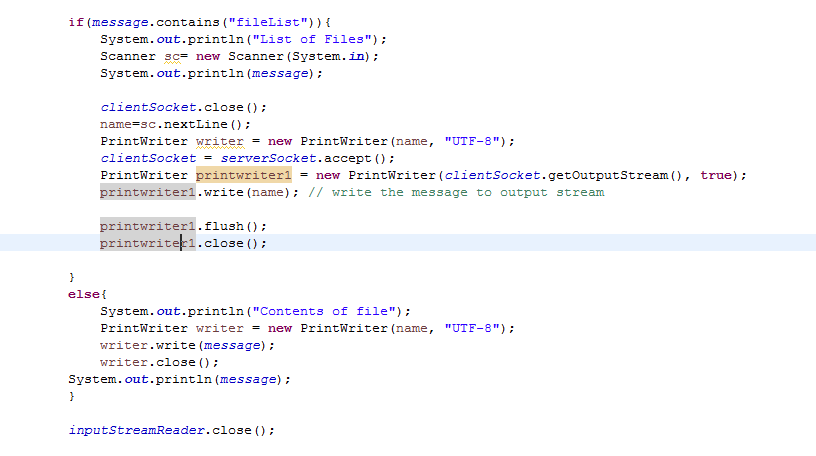
**3.2 Android as Server and Desktop as Client**

***Steps:***

1. Download and unzip the **Client.java** and import the file onto eclipse workspace.
2. We need to make a folder in the path of the project workspace which would hold file that the client would want to upload.
3. Similarly we need to make a folder by the name: **downloads** in the path: D:\ downloads, which would hold client Downloads.
4. Server side in the mobile, we need to make a folder by the name: **Networks** in the ExternalStroragePublicDirctory. which would hold a list of files in the mobile which the user could view and download and also have all the client uploaded files.
5. Run **Client.java** to run the app.

****

****

****

**4. Implementation Overview:**

The project has two parts:

1. PC as server and Android device as client are connected over Wi-Fi network and transfer files from server to client.
2. Android Device being Server and PC as client trying to communicate using a Desktop java console.

To achieve the communication between the two devices connected over the same WIFI network, there are connected over a TCP/IP socket connection. In the first part of project where PC is server and android as client, when we run the server program on Eclipse IDE and we start server on PC application, then the server is active. Now on the client side we run android application which is already hard coded with the server IP address both server and client are connected for further communication.

Now on the application interface on android device we have two buttons SEND and RECEIVE. Receive button we get us to the client part of the application where we have two more activity buttons “files list from server” and “file download on server”, which gives the list files on server and which downloads the selected file from the server to mobile device respectively. Files listing is done by using the buffer reader which reads the output from object output stream of the server.java and client side requests are handled using AsyncTask and files will be downloaded to a specifies path.

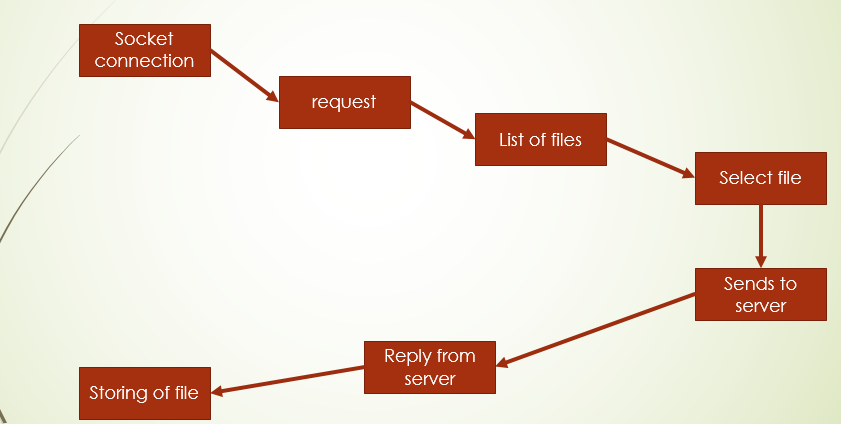
In the Second part of the project, android server is implemented using multithreading concepts in android and client desktop application is implemented using Swings. The same android application acts as client and server which can be triggered by clicking SEND Button on the android application which will start server and listen to a port.

\*\*\*\*\*\*

Server has two parts of code, list files and download. The android device communicates by sending certain messages for upload and download. Server waits for the client’s request. If the request is **list files** server will send the list of all filenames in the directory. If the request is **download file** the server reads the file name and sends the content of the file to client

Upon TCP connection being established, client will send list list request to view a list of file available on the server by clicking download files button. When the file to be downloaded is selected, it sends **downloading** message with file name. When server sends the requested file, client creates a new file and copies the contents to it. Client can also upload the file by clicking the upload files button, which will list the files under a directory on the client side. When the file to be uploaded is selected, it sends upload file with a file name, file length and the file contents which has to be uploaded to the server.

1. **Flow Diagram**

****

**Fig 1: File Transfer**

1. **Responsibilities**

The design, coding, development, implementation, testing and documentation was equally executed by all the team members.

1. **References**
2. http://android-er.blogspot.com/2014/02/android-sercerclient-example-client.html
3. http://examples.javacodegeeks.com/android/core/socket-core/android-socket-example/
4. http://www.tutorialspoint.com/java/java\_networking.html
5. http://developer.android.com/reference/android/os/AsyncTask.html
6. https://docs.oracle.com/javase/tutorial/networking/sockets/
7. http://stackoverflow.com/questions/9671546/asynctask-android-example