SOCKET PROGRAMMING WI F I CHAT APP FOR ANDROID SMARTPHONE

P18-0148 P18-0075 P18-0035 P18-0112

December 24, 2020

1 Introduction

In this project, we have developed an android application that enables two users to send and receive text via WiFi through socket programming. We have implemented this project on android phone. The language used in developing android apps is Java. All the back end functions and the socket programming is implemented in Java. The front end is implemented using XML and both the back end and front end are connected together in the Java program. We have used the Android Studio IDE to program the app.

2 Program Code

2.1 Java Code - Back end
Every android app is divided into three sections. These three sections interact with each other through the code to complete the entire app.
1.Java code
2.XML code
3.Android manifest
First look at the Java code. For this project, we had to create two applications one for the server and another for the client. Both these apps were running on two different smart-phones. The Java code of android apps start with a method called onCreate(). This method defines the initial setup of all the variables and the elements in the application. We can use this section to setup the buttons and define what will happen when the button is pressed. It is also used to link the element of the front end with the back end. For example the graphic front end element Button is linked to the Java code through this method. Similarly other elements are also linked.

```
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_server);
    wmanager = (WifiManager) getSystemService(WIFI_SERVICE);
    String ip =
    Formatter.formatIpAddress(wmanager.getConnectionInfo().getIpAddress());
    smessage = (EditText) findViewById(R.id.smessage);
    chat = (TextView) findViewById(R.id.chat);
    display_status = (TextView)
    findViewById(R.id.display_status);
    display_status.setText("Server hosted on " + ip);
    Thread serverThread = new Thread(new serverThread());
    serverThread.start();
    button_sent = (Button) findViewById(R.id.button_sent);
    button_sent.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
        {
        Thread sentThread = new Thread(new sentMessage());
        sentThread.start();
        }
    });
}
```

3 Server - socket program

Definition:
A socket is one endpoint of a two-way communication link between two programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to

Server - socket program A socket is an endpoint of a two-way communication link between two programs running on the network. Socket is bound to a port number. There are basically 5 steps to create a server program in Java. Since android apps are developed using Java, the same procedures apply here also.

1. Open server socket
2. Wait for client request
3. Create I/O stream for communication
4. Perform communication
5. Close socket

A server program creates a specific type of socket that is used to listen for client requests (server socket). In the case of a connection request, the program creates a new socket through which it will exchange data with the client using input and output streams. The socket abstraction is very similar to the file concept: developers have to open a socket, perform I/O, and close it.

```
// first open a socket in the server
ServerSocket serverSocket = new ServerSocket(port_number);
// next, wait for the client to connect
Socket client = server.accept();
// create I/O streams for communication
DataInputStream is;
DataOutputStream os;
is = new DataInputStream(client.getInputStream());
os = new DataOutputStream(client.getOutputStream());
// perform communication
// To receive from client:
String line = is.readLine();
// To send to client:
os.writeBytes();
// close the client socket
client.close();
```

We have implemented the Server program with two threads. One thread is called inside the onCreate method and it continuously monitors the input stream for incoming message. The second thread runs whenever the button is pressed. This is used to write data to the output stream.

4 Client - socket program

The steps involve in creating a client program is very similar to that of the server program.
1. Open socket
2. Create I/O stream for communication
3. Perform communication
4. Close socket

```
// first open a socket in the server
Socket socket = new Socket(server_ip , port_number);
// create I/O streams for communication
DataInputStream is;
DataOutputStream os;
is = new DataInputStream(client.getInputStream());
os = new DataOutputStream(client.getOutputStream());
// perform communication
// To receive from client:
String line = is.readLine();
// To send to client:
os.writeBytes();
// close the client socket
socket.close();
```

5 XML Code Front End

The front end of the application was designed in Android Studio IDE using XML (Extensible Markup Language). The Server and client apps are designed in different manners. The server application has just one button and one text field to input text. The client application has two buttons one to facilitate the connection and one to sent messages. It also has two text fields, one to input the IP address of the server and another to input the message to be sent.

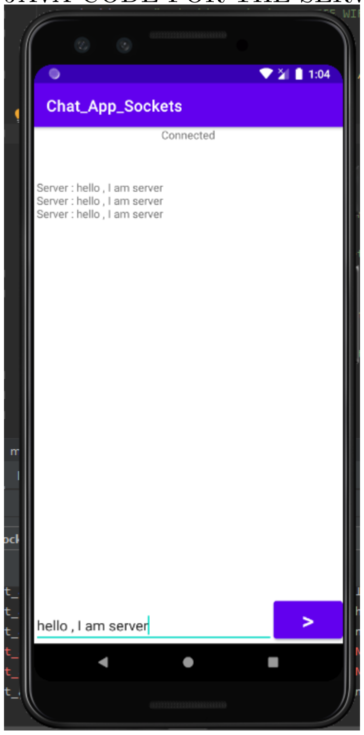
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.abby.server.ServerActivity">
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Status"
    android:id="@+id/display_status"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=">"
    android:textSize="30dp"
    android:id="@+id/button_sent"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/smessage"
    android:layout_alignParentBottom="true"
    android:layout_toLeftOf="@+id/button_sent"
    android:layout_toStartOf="@+id/button_sent" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:id="@+id/chat"
    android:layout_below="@+id/display_status"
    android:layout_marginTop="31dp"
    android:layout_above="@+id/button_sent"
    android:layout_alignRight="@+id/button_sent"
    android:layout_alignEnd="@+id/button_sent"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />
</RelativeLayout>
```

6 Client XML Code

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.abby.client.ClientActivity">
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/server_ip"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_toLeftOf="@+id/connect_phones"
    android:layout_toStartOf="@+id/connect_phones" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Connect"
    android:id="@+id/connect_phones"
    android:layout_alignBottom="@+id/server_ip"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_alignLeft="@+id/sent_button"
    android:layout_alignStart="@+id/sent_button" />
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/smessage"
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:hint="enter text to sent"
    android:layout_toLeftOf="@+id/sent_button"
    android:layout_toStartOf="@+id/sent_button" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=">"
    android:textSize="30dp"
    android:id="@+id/sent_button"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />
<TextView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:text=""
    android:id="@+id/chat"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/server_ip"
    android:layout_above="@+id/sent_button" />
</RelativeLayout>
```

7 All of Code

JAVA CODE FOR THE SERVER PROGRAM



Connected

Server : hello , I am server
Server : hello , I am server
Server : hello , I am server

hello , I am server

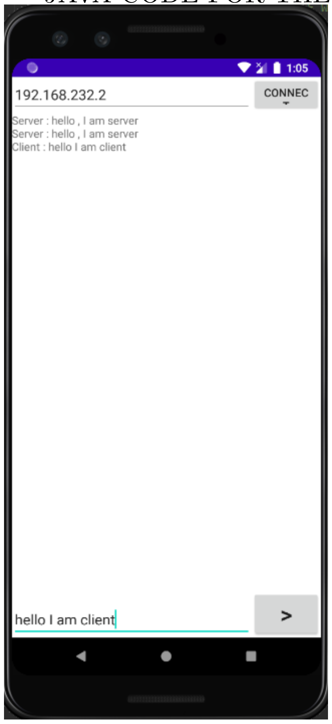
```java
package com.example.abby.server;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.text.format.Formatter;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
public class ServerActivity extends AppCompatActivity {
    Button button_sent;
    EditText smessage;
    TextView chat,display_status;
    String str,msg="";
```

```
int serverport = 10000;
ServerSocket serverSocket;
Socket client;
Handler handler = new Handler();
WifiManager wmanager;
@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_server);
    wmanager = (WifiManager) getSystemService(WIFI_SERVICE);
    String ip =
        Formatter.formatIpAddress(wmanager.getConnectionInfo().getIpAddress());
    smessage = (EditText) findViewById(R.id.smessage);
    chat = (TextView) findViewById(R.id.chat);
    display_status = (TextView)
        findViewById(R.id.display_status);
    display_status.setText("Server hosted on " + ip);
    Thread serverThread = new Thread(new serverThread());
    serverThread.start();
    button_sent = (Button) findViewById(R.id.button_sent);

    button_sent.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
        {
            Thread sentThread = new Thread(new sentMessage());
            sentThread.start();
        }
    });
}

    class sentMessage implements Runnable
    {
        @Override
        public void run()
        {
            try
            {
                Socket client = serverSocket.accept();
                DataOutputStream os = new
                    DataOutputStream(client.getOutputStream());
                str = smessage.getText().toString();
                msg = msg + "\n Server : " + str;
                handler.post(new Runnable()
                {
                    @Override
                    public void run()
                    {
```

```
                    chat.setText(msg);
                }
            });
            os.writeBytes(str);
            os.flush();
            os.close();
            client.close();
        }
        catch(IOException e)
        {
        }
    }
}
public class serverThread implements Runnable
{
    @Override
    public void run()
    {
        try
        {
            while(true)
            {
                serverSocket = new ServerSocket(serverport);
                Socket client = serverSocket.accept();
                handler.post(new Runnable() {
                    @Override
                    public void run() {
                        display_status.setText("Connected");
                    }
                });
                DataInputStream in = new
                    DataInputStream(client.getInputStream());
                String line = null;
                while((line = in.readLine()) != null)
                {
                    msg = msg + "\n Client : " + line;
                    handler.post(new Runnable()
                    {
                    @Override
                    public void run()
                    {
                        chat.setText(msg);
                    }
                });
            }
            in.close();
            client.close();
            Thread.sleep(100);
        }
    }
```

```
            catch (Exception e)
            {
            }
        }
    }
}
```

JAVA CODE FOR THE Client PROGRAM



```
192.168.232.2
Server : hello , I am server
Server : hello , I am server
Client : hello I am client
hello I am client
```

```
package com.example.abby.client;
import android.app.Activity;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
public class ClientActivity extends Activity {
    EditText serverIp,smessage;
   TextView chat;
   Button connectPhones,sent;
   String serverIpAddress = "",msg = "",str;
   Handler handler = new Handler();
   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_client);
       chat = (TextView) findViewById(R.id.chat);
       serverIp = (EditText) findViewById(R.id.server_ip);
       smessage = (EditText) findViewById(R.id.smessage);
       sent = (Button) findViewById(R.id.sent_button);
       sent.setOnClickListener(new View.OnClickListener() {
       @Override
       public void onClick(View v)
       {
       Thread sentThread = new Thread(new sentMessage());
       sentThread.start();
       }
   });
   connectPhones = (Button) findViewById(R.id.connect_phones);
   connectPhones.setOnClickListener(new View.OnClickListener()
   {
       @Override
       public void onClick(View v)
       {
           serverIpAddress = serverIp.getText().toString();
           if (!serverIpAddress.equals(""))
           {
               Thread clientThread = new Thread(new
                  ClientThread());
               clientThread.start();


           }
       }
   });
}
class sentMessage implements Runnable
{
   @Override
   public void run()
   {
       try
       {
           InetAddress serverAddr =
               InetAddress.getByName(serverIpAddress);
           Socket socket = new Socket(serverAddr, 10000); //
               create client socket
           DataOutputStream os = new
               DataOutputStream(socket.getOutputStream());
           str = smessage.getText().toString();
           str = str + "\n";
           msg = msg + "Client : " + str;
           handler.post(new Runnable() {
               @Override
               public void run() {
                  chat.setText(msg);
               }
           });
           os.writeBytes(str);
           os.flush();
           os.close();
           socket.close();
       }
       catch(IOException e)
       {
       }
   }
}
public class ClientThread implements Runnable
{
   public void run()
   {
       try
       {
           while(true)
           {
               InetAddress serverAddr =
                  InetAddress.getByName(serverIpAddress);
           Socket socket = new Socket(serverAddr, 10000); //
               create client socket

           DataInputStream in = new
               DataInputStream(socket.getInputStream());
           String line = null;
           while ((line = in.readLine()) != null)
           {
               msg = msg + "Server : " + line + "\n";
               handler.post(new Runnable()
               {
                  @Override
                  public void run()
                  {
                      chat.setText(msg);
                  }
               });
           }
           in.close();
           socket.close();
           Thread.sleep(100);
       }
   }
   catch (Exception e)
   {}
}
}
}
```