Android App *iClass*

F0903034 F0903033
DAI JI ZHOU ZHIYI
5090309767 5090309747

Introduction

Introduction:
In this report, we mainly focus on discussing the ideas and designing procedures of our android-b ased software called Iclass. We will show specific functions of Iclass as well as their methods of re alization. After that, a demo and future works will be displayed along with source codes.

Background

There is varies of class assistant software used in today. However we still find some short points in them.
First of all, they are not portable enough. Most of the software are targeting at the PC part. There is hardly any apps desire for mobile devices. It is unpractical to ask student to take the large inconvenience of bringing tablet to the class just for some small convenience. But it is different when coming to mobile devices. The students take the cell phone with them every day. So there won't be any additional inconvenience.
Secondly, unnecessary functions complicate the software. The users have to take a long time to learn the instruction. This will lead to a situation which neither of the developers and users wants to encounter. They might abandon the software since it brings to much inconvenience before truly offer convenience.

Design

User interface

The user interface is designed with tab. Which means for the teacher part, it has two tabs, first one is for role call and screen shot, second one is for chatting and file transmission. The user interface will be showed in demo.

Function

Role calling

Consider a situation. A university class usually contains about 50 students. Suppose a calling and a response takes 5 second. The hole procedure at least takes 5 minutes. However, there is large probability that the teacher read the students' names incorrectly. Or even more possible that the student absent the class, that the procedure takes more than 15 minutes. This is the main reason why teacher usually doesn't do the role calling which actually stay a large part in score the attendance of the student. So we improve the role calling procedure. When the student connects his device with the teachers, he will automatically send a message containing a unique information which indicates who he is. Then in the teacher's part, the state beside his name will change into attended.

Chatting

The student and teacher might some private room to chat for some problem, chatting room provide both public and private chatting selection. In public part, all the talks will be shown in every mobile device.

File sending

The teacher might assign class quiz, which can be done with a small paper. Usual way will be students handing the paper after the class. However, we now provide a selection that students hand in the quiz written by their mobile devices, which is easily to be modified and corrected.

Implementation

What is socket

So-called socket is usually used to describe the IP address and port, which is the handle of a communication chain. The application program usually sends request to the network through the "socket" or answering the network requests.

Take the J2SDK-1.3 as an example, the Socket and ServerSocket class library is located in the java.net package. ServerSocket is used for the server-side, while Socket is one to use when establishing a network connection. When successful connection occurs, both ends of the

application will have a Socket instance and operate on this instance as well as complete the required session. For a network connection, the socket is equal, and there is no difference for users on the server side or client because they won't have a different level. Regardless of the Socket or ServerSocket, their work is should be completed through the class named SocketImpl and its subclasses.

Socket API:

java.net.Socket is the inheritance of java.lang.Object, which has eight constructors. Here are the most frequently used three methods, for other methods, we can see the JDK-1.3 document.

Accept method is used to have a "block" until it receives a connection, and returns a client Socket object instance. "block" is a term, which makes the program temporarily "stay" in this place, until a session occurs, then the program continues; usually "block" is generated by the cycle.

Using getInputStream method to obtain a network connection input and returns the object instance of a IutputStream.

Using the getOutputStream method to get the input at the other end of connection, at the same time returning the object instance of an OutputStream on.

Note: both getInputStream and getOutputStream method will generate an IOException, it must be captured, because they return the stream object, which will be usually used by another stream object.

How to use it

Development principles:

Server uses ServerSocket to listen on the specified port, the port can be ordered casually (since ports below 1024 are usually reserved ports, in some operating systems they can not be used casually, so it is recommended to use ports greater than 1024), waiting for the client connection request, and after the customer connection, the session occurs; then after the session is complete, close the connection.

Client uses Socket to send a connection request to a certain port of the server, once the connection is successfully done, session will be opened; and after the completion of the session, the Socket will be closed. The client does not need to specify the port to be opened; usually a temporary and dynamic port greater than 1024 will be allocated.

Role calling

When the connection is built, the client sends a unique message which indicates who he is to the server part. The server has a array which contains all the identification information in the class. The server then checks the array an find the correspondent information that matches the sent one. And change the ViewText beside the student's name.

Chatting and file transmission

1. Initialize WIFI

```
WifiManager wifiManager = (WifiManager)getSystemService(Context.WIFI_SERVICE);
        if(!wifiManager.isWifiEnabled()){
            wifiManager.setWifiEnabled(true);
        }
        WifiInfo wifiInfo = wifiManager.getConnectionInfo();
        int ipAddress = wifiInfo.getIpAddress();
        ip = new String(intToIp(ipAddress));
```

2. Get IP

```
 public String GetHostIp(){
        try {
            for (Enumeration<NetworkInterface> en = NetworkInterface
            .getNetworkInterfaces(); en.hasMoreElements();) {
                NetworkInterface intf = en.nextElement();
                for (Enumeration<InetAddress> ipAddr = intf.getInetAddresses(); ipAddr
                .hasMoreElements();) {
                    InetAddress inetAddress = ipAddr.nextElement();
                    if (!inetAddress.isLoopbackAddress()) {
                        return inetAddress.getHostAddress();
                    }
                }
            }
        }  catch (Exception e) {

        }
        return "null";
    }
```

3. Send message

```
public void SendMessage()
    {
        try
        {
            if(mClientList.size()<1)
```

```
                    myDialog.append("\nempty");
                else
                    myDialog.append("\n"+mClientList.get(0).getInetAddress().toString());
            }
            catch(Exception ex)
            {
                myDialog.append(ex.toString());
            }
            for(int i = 0;i<mClientList.size();i++){
                Socket childSocket = mClientList.get(i);

                try {
                    PrintWriter mPrintWriter = new
PrintWriter(childSocket.getOutputStream(),true);
                    mPrintWriter.print("MSG"+typeMessage.getText()+"\n");
                    mPrintWriter.flush();

                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                    myDialog.setText(e.toString());
                }



            }
            myDialog.append("\n"+typeMessage.getText());
            //myDialog.setText(myDialog.getText()+"\n"+typeMessage.getText());
            typeMessage.setText("");
        }
```

4. Send file

```
public void SendFileReq(String FileName)
    {
        if(mClientList.size()>0)
        {
            Socket childSocket = mClientList.get(mClientList.size()-1);
            try{
                PrintWriter mPrintWriter = new
PrintWriter(childSocket.getOutputStream(),true);
                mPrintWriter.print("FILREQ"+FileName+"\n");
                mPrintWriter.flush();
            }
            catch(IOException e)
            {
```

```
                }
            }
            else
            {
                /*Socket childSocket = mClientList.get(0);
                try{
                    PrintWriter mPrintWriter = new
PrintWriter(childSocket.getOutputStream(),true);
                    mPrintWriter.print("MSGFILREQmClientListEmpty"+FileName+"\n");
                    mPrintWriter.flush();
                }
                catch(IOException e)
                {

                }*/
            }
        }
```

Notification

1. A new thread needs to be set when a connection is built.
2. A menu needs to be built in the app in order to locate the file

```
    public void jumpToLayerFile()
    {
        setContentView(R.layout.openfile);
        m_EditText = (EditText)findViewById(R.id.editTextFileName);
        buttonBack=(Button)findViewById(R.id.buttonback);
        //path = "123";
        buttonCancelFile = (Button)findViewById(R.id.buttonCancleFile);
        m_FileListView = (ListView)findViewById(R.id.listView1);

        path = Environment.getRootDirectory().getPath();
        showFiles();
        buttonBack.setOnClickListener(new OnClickListener()
        {

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                if((file.getParent())!=null)
                {
                    path = file.getParent();
```

```
                    m_EditText.setText(path);
                    showFiles();
                }
                else
                {
                    m_EditText.setText(path);
                    showFiles();
                }
            }

        });

        buttonCancelFile.setOnClickListener(new OnClickListener(){

            public void onClick(View v) {
                // TODO Auto-generated method stub
                jumpToMainLayer();
            }
        }
        );

    }

    public void jumpToMainLayer()
    {
        setContentView(R.layout.main);
        LayoutMainInit();

    }

    public void showFiles()
    {
        file = new File(path);


        IconifiedTextListAdapter iTLA = new IconifiedTextListAdapter(this);
        //int l = Environment.getRootDirectory().list().length;
        try
        {
            if(file.list()!=null)
            {
                for(int i =0;i<file.list().length;i++){
                    IconifiedText iT = new IconifiedText(file.listFiles()[i].getName());
                    //IconifiedTextView iTV = new IconifiedTextView(this,iT);
```

```
                    //m_FileListView.
                    //iTV.setText(file.listFiles()[i].getName());
                    //iTV.getView();
                    iTLA.addItem(iT);
                    //
                    iTLA.getView(i, null, m_FileListView);
                    //iTLA.addItem(iT);
                }

                    m_FileListView.setAdapter(iTLA);

                    m_FileListView.setOnItemClickListener(new OnItemClickListenerImpl());
            }
        }
        catch(Exception ex){
            m_EditText.setText(ex.toString());
        }
    }
```

Screenshot

Screenshot for inquiry: Iclass allow the professor to get the screenshot of student's current screen under the permission of the student so that some questions which are so complicated to be expressed can be understood by the professor easily.

Designing procedure: Here we just give the rough outline of the designing procedure of the function named screenshot inquiry.

(1) We obtain the screen shot of the specific activity.

```
// obtain the screen shot of the specific activity
    private static Bitmap takeScreenShot(Activity activity){
```

(2) Secondly, we claim a view as the screen shot, which is the picture we want to get.

```
//claim view as the screen shot
    View view = activity.getWindow().getDecorView();
    view.setDrawingCacheEnabled(true);
    view.buildDrawingCache();
    Bitmap b1 = view.getDrawingCache();
```

(3) Thirdly, we just fill in some parameters of the screenshot we want such as obtaining the width and height of the screen shot.

```
//obtain the width and height of the screen shot
    int width = activity.getWindowManager().getDefaultDisplay().getWidth();
```

```
int height = activity.getWindowManager().getDefaultDisplay().getHeight();
```

(4) Finally, the professor can get the screen shot of the student.
Note: before this action, the professor must get the permission of the student of using the screenshot of his/her cell phone, otherwise it won't work.

Demo

The following pictures are the interfaces of server and client:

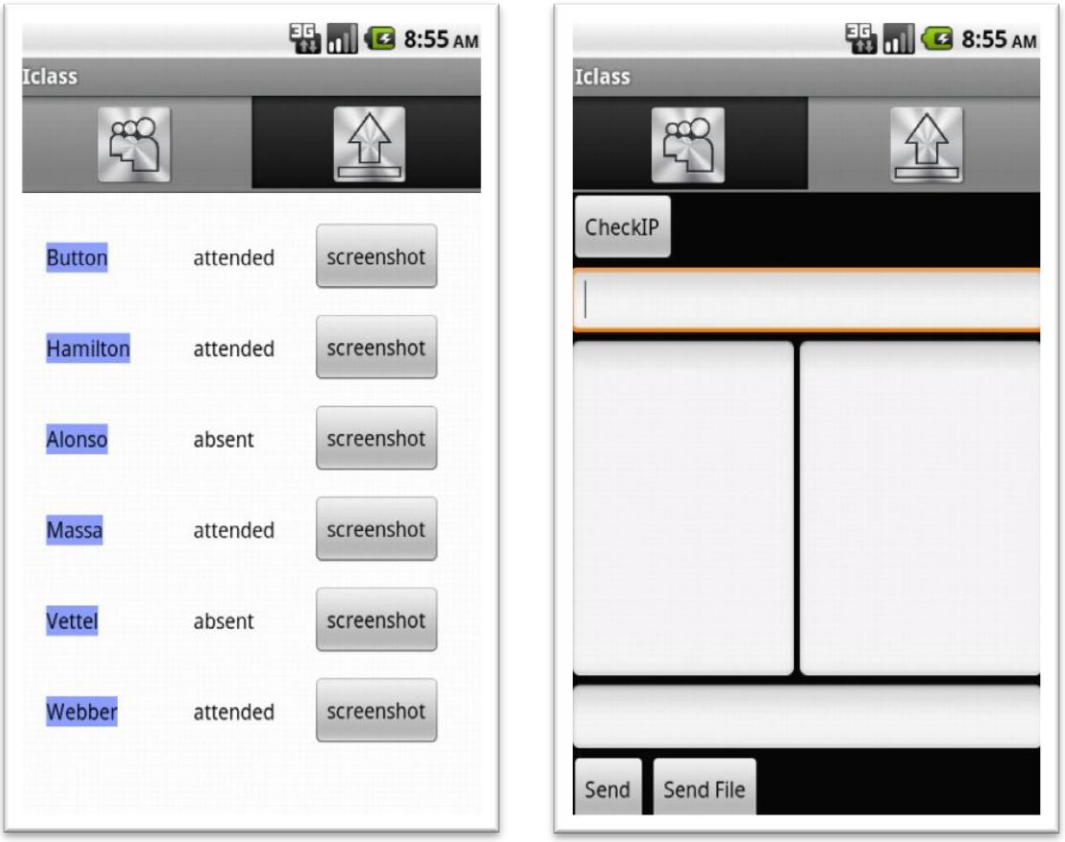

Figure 1 Interfaces of server

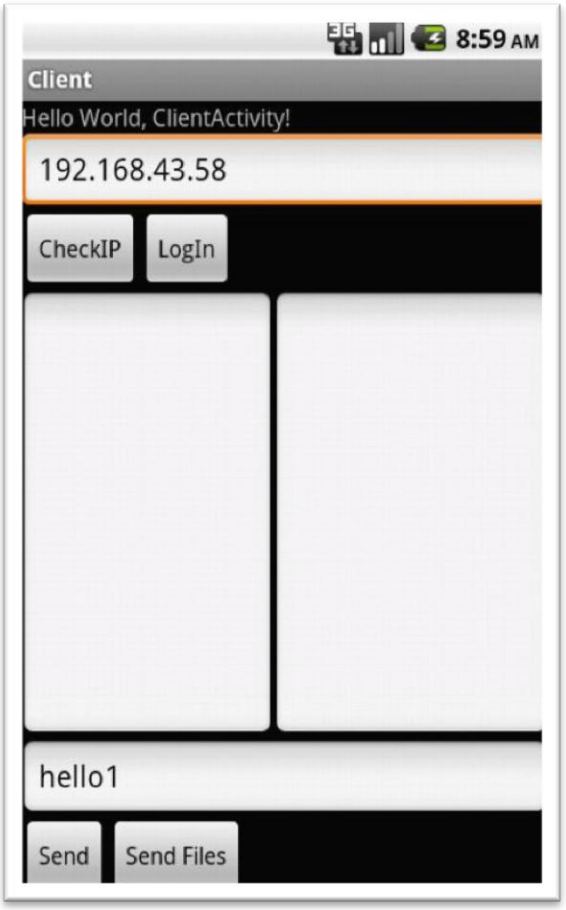

Figure 2 interface of client

We can see that as long as one client(student) logged in, the server will receive a message indicating the identification of the client(student), then the server automatically searches the matched data in the database and changes the correspondent student's name from "absent" to "attended".

Then the following pirctures show the function of chatting room for inquiry:

Figure 3 procedures of chatting room

From the picture, we can find that as long as the client(student) logged in, the student can easily chat with professor for inquiry.

Then the following pirctures show the function of uploading homeworks:

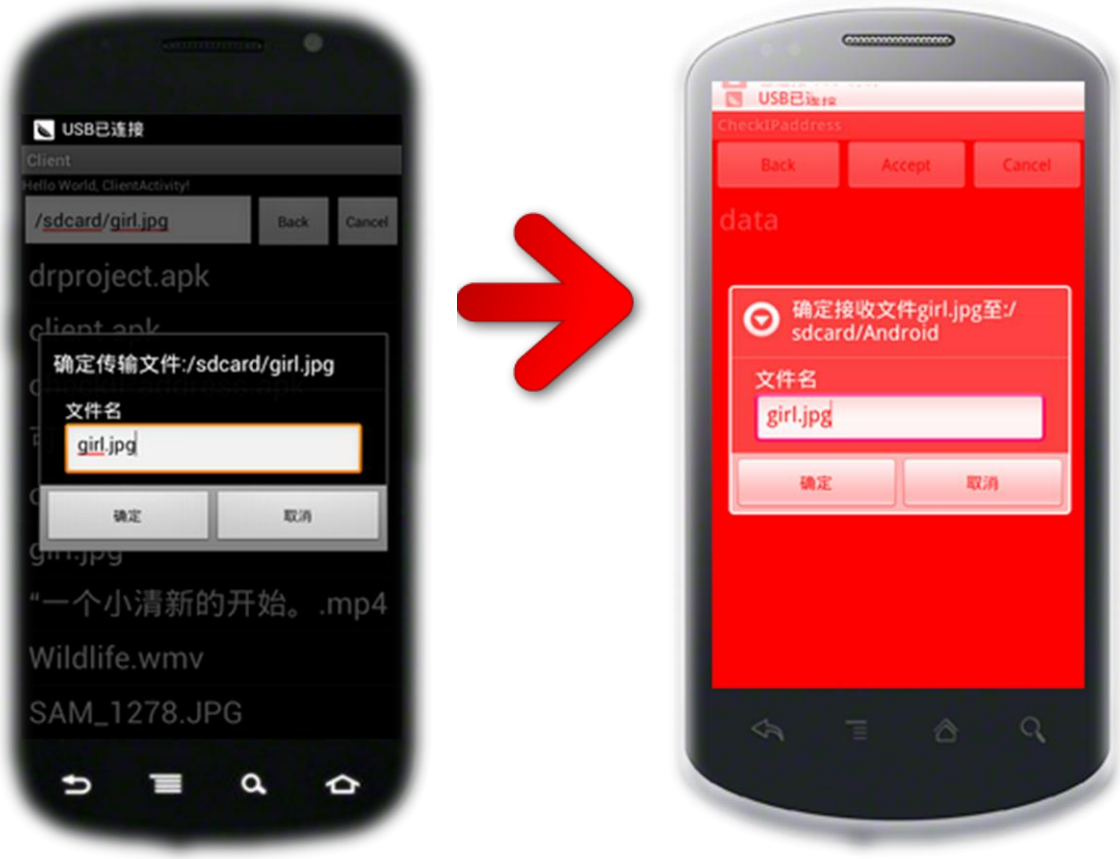

Figure 4 procedures of uploading

We can see that the client(student) choose the right path of the homework in his/her cell phone, then send it to the server(professor), the cell phone of the server(professor) will ask him to choose a path to save this file(homework).

Figure 5 procedures of uploading

Then the transmission of file begins, and the hint of finishing will be displayed on both cell phones to ensure the student and professor that the file is transmitted successfully.

Future development

We first want to improve the apps. We are going to add a function for the teacher to broadcast a file. The students can receive it in a same time slot no need to worry about the collision.

Different Operating System. We want to transplant our app to iOS(Apple) and RIM(BlackBerry). As we all know that apple mobile devices are currently the most popular among the students. The iOS enjoys large share of mobile OS market. So transplanting seems to be very necessary. Also Blackberry is used mostly in business field. As we add more function, it would be very likely to step into the RIM market. Furthermore, we could make it fit for an tablet PC. Since tablet PC is

becoming increasingly popular and efficient for daily use.

Source code

Server part

Main.xml(server)

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
        <AbsoluteLayout

        android:id="@+id/absolutelayout01"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@color/white"
        >

        <TextView
        android:id="@+id/tv1"
        android:layout_x="20dp"
        android:layout_y="33dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/std1"
        android:background="@color/blue"
        android:textColor="@color/black"
        >

        </TextView>

        <TextView
        android:id="@+id/tv2"
        android:layout_x="20dp"
        android:layout_y="93dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/std2"
        android:background="@color/blue"
        android:textColor="@color/black"
```

```
        >

    </TextView>

    <TextView
        android:id="@+id/tv3"
        android:layout_x="20dp"
        android:layout_y="153dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/std3"
        android:background="@color/blue"
        android:textColor="@color/black"
        >

    </TextView>

    <TextView
        android:id="@+id/tv4"
        android:layout_x="20dp"
        android:layout_y="213dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/std4"
        android:background="@color/blue"
        android:textColor="@color/black"
        >

    </TextView>

    <TextView
        android:id="@+id/tv5"
        android:layout_x="20dp"
        android:layout_y="273dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/std5"
        android:background="@color/blue"
        android:textColor="@color/black"
        >

    </TextView>

    <TextView
```

```
        android:id="@+id/tv6"
        android:layout_x="20dp"
        android:layout_y="333dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/std6"
        android:background="@color/blue"
        android:textColor="@color/black"
        >

    </TextView>

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_x="200dp"
        android:layout_y="20dp"
        android:text="@string/sc"
        >

    </Button>

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_x="200dp"
        android:layout_y="80dp"
        android:text="@string/sc"
        >

    </Button>

    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_x="200dp"
        android:layout_y="140dp"
        android:text="@string/sc"
        >

    </Button>
```

```
<Button
    android:id="@+id/button4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="200dp"
    android:layout_y="200dp"
    android:text="@string/sc"
    >

</Button>

<Button
    android:id="@+id/button5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="200dp"
    android:layout_y="260dp"
    android:text="@string/sc"
    >

</Button>

<Button
    android:id="@+id/button6"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="200dp"
    android:layout_y="320dp"
    android:text="@string/sc"
    >

</Button>

<TextView
    android:id="@+id/st1"
    android:layout_x="120dp"
    android:layout_y="33dp"
    android:layout_width="wrap_content"
    android:layout_height="20dp"
    android:text="@string/on"
    android:textColor="@color/black"
    >
```

```
    </TextView>

    <TextView
        android:id="@+id/st2"
        android:layout_x="120dp"
        android:layout_y="93dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/on"
        android:textColor="@color/black"
        >

    </TextView>

    <TextView
        android:id="@+id/st3"
        android:layout_x="120dp"
        android:layout_y="153dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/off"
        android:textColor="@color/black"
        >

    </TextView>

    <TextView
        android:id="@+id/st4"
        android:layout_x="120dp"
        android:layout_y="213dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
        android:text="@string/on"
        android:textColor="@color/black"
        >

    </TextView>

    <TextView
        android:id="@+id/st5"
        android:layout_x="120dp"
        android:layout_y="273dp"
        android:layout_width="wrap_content"
        android:layout_height="20dp"
```

```
            android:text="@string/off"
            android:textColor="@color/black"
            >

        </TextView>

        <TextView
            android:id="@+id/st6"
            android:layout_x="120dp"
            android:layout_y="333dp"
            android:layout_width="wrap_content"
            android:layout_height="20dp"
            android:text="@string/on"
            android:textColor="@color/black"
            >

        </TextView>
</AbsoluteLayout>
<LinearLayout
        android:id="@+id/linearlayout02"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical"
        >
        <Button
            android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="CheckIP" />

        <EditText
            android:id="@+id/editText1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10" >

          <requestFocus />
        </EditText>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="0.37" >
```

```
        <EditText
            android:id="@+id/editText2"
            android:layout_width="125dp"
            android:layout_height="match_parent"
            android:layout_weight="0.03"
            android:ems="10"
            android:gravity="top"
            android:inputType="textMultiLine"
            android:scrollbars="vertical" />

        <EditText
            android:id="@+id/editText4"
            android:layout_width="158dp"
            android:layout_height="match_parent"
            android:layout_weight="0.01"
            android:ems="10"
            android:gravity="top"
            android:inputType="textMultiLine"
            android:scrollbars="vertical" />
    </LinearLayout>

    <EditText
        android:id="@+id/editText3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textMultiLine" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/button2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Send" />

        <Button
            android:id="@+id/buttonSendFile"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Send File" />
```

```
    </LinearLayout>

</LinearLayout>


</FrameLayout>
```

Activity(server)

```
package Dr.IPadd;



import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.Enumeration;
import java.util.List;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

import Dr.IPadd.Server.ThreadServer;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.wifi.WifiInfo;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.os.Message;
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.AdapterView.OnItemClickListener;



public class CheckIPaddressActivity extends Activity {
	/** Called when the activity is first created. */

	Button button_checkIP;
	EditText editText_showIP;
	EditText myDialog;
	EditText guestDialog;
	Button button_sendMessage;
	EditText typeMessage;
	String ip = "";
	//Server server;
	ServerSocket ser;
	private static List<Socket> mClientList = new ArrayList<Socket>();
	private ExecutorService mExecutorService;
	private String receiveMessage="";
	Thread listenthread;
	Socket client=null;
	private String clientIP=null;
	public String errorMessage="";
	Button button_SendFile;
	private PrintWriter mPrintWriter = null;
	Socket clientReceive=null;

	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);

		LayoutMainInit();

		/*try
		{
			server = new Server(50500);
```

```
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
            editText_showIP.setText(ex.toString());
        }*/
        ServeInit();
    }

    public void LayoutMainInit()
    {
        button_checkIP = (Button)findViewById(R.id.button1);
        editText_showIP = (EditText)findViewById(R.id.editText1);
        myDialog = (EditText)findViewById(R.id.editText2);
        guestDialog = (EditText)findViewById(R.id.editText4);
        typeMessage = (EditText)findViewById(R.id.editText3);
        button_sendMessage = (Button)findViewById(R.id.button2);
        button_SendFile = (Button)findViewById(R.id.buttonSendFile);




        WifiManager wifiManager = (WifiManager)getSystemService(Context.WIFI_SERVICE);
        if(!wifiManager.isWifiEnabled()){
            wifiManager.setWifiEnabled(true);
        }
        WifiInfo wifiInfo = wifiManager.getConnectionInfo();
        int ipAddress = wifiInfo.getIpAddress();
        ip = new String(intToIp(ipAddress));



        button_checkIP.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v){
                editText_showIP.setText(ip);
            }
        });

        button_sendMessage.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v){
                SendMessage();
                //myDialog.setText(GetHostIp());
            }
        });
```

```
        button_SendFile.setOnClickListener(new Button.OnClickListener(){

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                jumpToLayerFile();
                /*
                receiveMessage = "FILREQssss";
                Message msg = new Message();
                msg.what = 0x1984;
                mHandler.sendMessage(msg);*/
            }

        });


}

ListView m_FileListView;
private String path;
File file;
EditText m_EditText;
Button buttonBack;
Button buttonCancelFile;
String fileToSend;
String fileToSendName;

public void jumpToLayerFile()
{
        setContentView(R.layout.openfile);
        m_EditText = (EditText)findViewById(R.id.editTextFileName);
        buttonBack=(Button)findViewById(R.id.buttonback);
        //path = "123";
        buttonCancelFile = (Button)findViewById(R.id.buttonCancleFile);
        m_FileListView = (ListView)findViewById(R.id.listView1);

        path = Environment.getRootDirectory().getPath();
        showFiles();
        buttonBack.setOnClickListener(new OnClickListener()
        {

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                if((file.getParent())!=null)
```

```
                {
                        path = file.getParent();
                        m_EditText.setText(path);
                        showFiles();
                }
                else
                {
                        m_EditText.setText(path);
                        showFiles();
                }
        }

    });

    buttonCancelFile.setOnClickListener(new OnClickListener(){

        public void onClick(View v) {
            // TODO Auto-generated method stub
            jumpToMainLayer();
        }
    }
    );

}

public void jumpToMainLayer()
{
    setContentView(R.layout.main);
    LayoutMainInit();

}

public void showFiles()
{
    file = new File(path);


    IconifiedTextListAdapter iTLA = new IconifiedTextListAdapter(this);
    //int l = Environment.getRootDirectory().list().length;
    try
    {
        if(file.list()!=null)
        {
            for(int i =0;i<file.list().length;i++){
```

```
                        IconifiedText iT = new IconifiedText(file.listFiles()[i].getName());
                        //IconifiedTextView iTV = new IconifiedTextView(this,iT);
                        //m_FileListView.
                        //iTV.setText(file.listFiles()[i].getName());
                        //iTV.getView();
                        iTLA.addItem(iT);
                        //
                        iTLA.getView(i, null, m_FileListView);
                        //iTLA.addItem(iT);
                    }

                        m_FileListView.setAdapter(iTLA);

                        m_FileListView.setOnItemClickListener(new OnItemClickListenerImpl());
                }
            }
            catch(Exception ex){
                m_EditText.setText(ex.toString());
            }
        }

        private class OnItemClickListenerImpl implements OnItemClickListener{
            @SuppressWarnings("unchecked")
            public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,
                    long arg3) {
                // TODO Auto-generated method stub
                if(file.listFiles()!=null){
                path = file.listFiles()[arg2].getPath();
                m_EditText.setText(path);
                if(file.listFiles()[arg2].isDirectory())
                {
                    showFiles();
                }
                else
                {
                    LayoutInflater factory = LayoutInflater.from(CheckIPaddressActivity.this);

                    fileToSendName = file.listFiles()[arg2].getName();
                    fileToSend = path;
                    final View DialogView = factory.inflate(R.layout.rename, null);
                    AlertDialog dlg = new AlertDialog.Builder(CheckIPaddressActivity.this)
                    .setTitle("确定传输文件:"+path).setView(DialogView).setPositiveButton("确
定",
                    new DialogInterface.OnClickListener() {
```

```
                    public void onClick(DialogInterface dialog, int which) {
                        // TODO Auto-generated method stub
                        //FileResourceActivity.this.finish();
                        //dialog.cancel();
                        SendFileReq(fileToSendName);
                        jumpToMainLayer();



                    }
                }).setNegativeButton("取消",
                new DialogInterface.OnClickListener()
                {
                    public void onClick(DialogInterface dialog, int which) {
                        // TODO Auto-generated method stub
                        //FileResourceActivity.this.finish();
                        fileToSend = null;
                        dialog.cancel();
                    }
                }).create();
                dlg.show();

            }
        }}
    }

    private String intToIp(int i){
        return(i & 0xFF)+"."+((i>>8)&0xFF)+"."+((i>>16)&0xFF)+"."+((i>>24)&0xFF);
    }

    public String GetHostIp(){
        try {
            for (Enumeration<NetworkInterface> en = NetworkInterface
            .getNetworkInterfaces(); en.hasMoreElements();) {
                NetworkInterface intf = en.nextElement();
                for (Enumeration<InetAddress> ipAddr = intf.getInetAddresses(); ipAddr
                .hasMoreElements();) {
                    InetAddress inetAddress = ipAddr.nextElement();
                    if (!inetAddress.isLoopbackAddress()) {
                        return inetAddress.getHostAddress();
                    }
                }
            }
```

```
        }    catch (Exception e) {

        }
        return "null";
    }

    public void SendFileReq(String FileName)
    {
        if(mClientList.size()>0)
        {
            Socket childSocket = mClientList.get(mClientList.size()-1);
            try{
                PrintWriter mPrintWriter = new
PrintWriter(childSocket.getOutputStream(),true);
                mPrintWriter.print("FILREQ"+FileName+"\n");
                mPrintWriter.flush();
            }
            catch(IOException e)
            {

            }
        }
        else
        {
            /*Socket childSocket = mClientList.get(0);
            try{
                PrintWriter mPrintWriter = new
PrintWriter(childSocket.getOutputStream(),true);
                mPrintWriter.print("MSGFILREQmClientListEmpty"+FileName+"\n");
                mPrintWriter.flush();
            }
            catch(IOException e)
            {

            }*/
        }
    }

    public void SendMessage()
    {
        try
        {
            if(mClientList.size()<1)
                myDialog.append("\nempty");
```

```
                else
                    myDialog.append("\n"+mClientList.get(0).getInetAddress().toString());
            }
            catch(Exception ex)
            {
                myDialog.append(ex.toString());
            }
            for(int i = 0;i<mClientList.size();i++){
                Socket childSocket = mClientList.get(i);

                try {
                    PrintWriter mPrintWriter = new
PrintWriter(childSocket.getOutputStream(),true);
                    mPrintWriter.print("MSG"+typeMessage.getText()+"\n");
                    mPrintWriter.flush();

                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                    myDialog.setText(e.toString());
                }



            }
            myDialog.append("\n"+typeMessage.getText());
            //myDialog.setText(myDialog.getText()+"\n"+typeMessage.getText());
            typeMessage.setText("");
        }

        private Runnable doThread = new Runnable() {
            public void run(){
                while(true)
                {

                    try
                    {
                        Message msg = new Message();
                        //msg.what = 0x1981;
                        //receiveMessage = new String("running");
                        //mHandler.sendMessage(msg);
                        client = ser.accept();
                        //msg = new Message();
                        msg.what = 0x1982;
```

```
                        clientIP = client.getInetAddress().toString();
                        mHandler.sendMessage(msg);

                        //msg = new Message();
                        //msg.what = 0x1981;
                        //receiveMessage = new String("running2");
                        //mHandler.sendMessage(msg);

                        mClientList.add(client);
                        //myDialog.append("size:"+mClientList.size()+"");
                        mExecutorService.execute(new ThreadServer(client));

                        Thread.sleep(100);
                    }
                    catch(Exception ex)
                    {
                        Message msg = new Message();
                        msg.what = 0x1983;
                        errorMessage=ex.toString();
                        mHandler.sendMessage(msg);
                        continue;
                    }
                }
            }
        };

        public void ServeInit()
        {
            try
            {
                ser = new ServerSocket(50500);
                mExecutorService = Executors.newCachedThreadPool();
                client = null;

        guestDialog.setText("serverStart\n"+ser.getLocalSocketAddress().toString()+"\n"+ser.getInet
Address().toString());


                listenthread =new Thread(doThread,"WaitingforLog");
                listenthread.start();

            }
            catch(Exception ex)
            {
```

```
                ex.printStackTrace();
                guestDialog.setText(ex.toString());
            }
        }

        class ThreadServer implements Runnable
        {
            private Socket mSocket;
            private BufferedReader mBufferedReader;
            private PrintWriter mPrintWriter;
            private String mStrMSG;


            public ThreadServer(Socket socket) throws IOException
            {
                this.mSocket = socket;
                mBufferedReader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            }

            public void run()
            {
                try
                {

                    //mSocket.setReceiveBufferSize(65534);
                    int buffersize = mSocket.getReceiveBufferSize();
                    while(true){
                        //mStrMSG = mBufferedReader.readLine())!=null
                        char[] tempbyte = new char[buffersize];
                        mBufferedReader.read(tempbyte);
                        mStrMSG = String.valueOf(tempbyte);

                        if(mStrMSG.trim().equals("exit"))
                        {
                            mClientList.remove(mSocket);
                            mBufferedReader.close();
                            mPrintWriter.close();
                            mStrMSG="user:"+this.mSocket.getInetAddress()+"exit!";
                            receiveMessage = mStrMSG;
                            Message msg = new Message();
                            msg.what = 0x1981;
                            mHandler.sendMessage(msg);
                            //gText.setText(gText.getText()+"\n"+mStrMSG);
```

```
                    mSocket.close();
                    break;
                }
                else
                {
                    if(mStrMSG.trim().substring(0,3).equals("MSG"))
                    {

    mStrMSG=mSocket.getInetAddress()+":"+mStrMSG.trim().substring(3);
                        receiveMessage = mStrMSG;
                        Message msg = new Message();
                        msg.what = 0x1981;
                        mHandler.sendMessage(msg);
                    }
                    else if(mStrMSG.trim().substring(0,3).equals("FIL"))
                    {
                        if(mStrMSG.trim().substring(3,6).equals("REQ"))
                        {
                            //receiveMessage =
mSocket.getInetAddress()+"REQFILETransmission:"+receiveMessage.trim().substring(6);
                            //receiveMessage =
mSocket.getInetAddress()+"REQFILETransmission:"+mStrMSG.trim().substring(6);
                            receiveMessage = mStrMSG.trim().substring(6);
                            clientReceive = mSocket;
                            Message msg = new Message();
                            msg.what = 0x1984;
                            mHandler.sendMessage(msg);

    //jumpToLayoutReceive(mStrMSG.substring(6),this.mSocket);
                        }
                        else if(mStrMSG.trim().substring(3,6).equals("ACK"))
                        {
                            receiveMessage = "开始传输";
                            //jumpToMainLayer();
                            Message msg = new Message();
                            msg.what = 0x1981;
                            mHandler.sendMessage(msg);
                            //sendFileBySocket(sendFileRead(path),mSocket);
                        }
                        else if(mStrMSG.trim().substring(3,6).equals("REF"))
                        {
                            receiveMessage = mSocket.getInetAddress()+"拒绝接收";
                            //jumpToMainLayer();
                            Message msg = new Message();
```

```
                                        msg.what = 0x1981;
                                        mHandler.sendMessage(msg);

                                    }
                                    else if(mStrMSG.trim().substring(3,6).equals("SND"))
                                    {

    mStrMSG=mSocket.getInetAddress()+":";//+mStrMSG.substring(6);
                                        //tempbyte;
                                        //StringBuffer sb = new StringBuffer();
                                        //int c;
                                        String temp = new String(tempbyte);



                                        receiveMessage = temp.trim();

                                        while(receiveMessage.lastIndexOf("ZYEOF")==-1){
                                        tempbyte = new char[buffersize];
                                        mBufferedReader.read(tempbyte);
                                        temp = new String(tempbyte);
                                        receiveMessage = receiveMessage+temp.trim();
                                        //receiveMessage = temp.substring(0,1).trim();
                                        //Message msg = new Message();
                                        //msg.what = 0x1981;
                                        //mHandler.sendMessage(msg);
                                        }
                                        receiveMessage                                          =
receiveMessage.substring(0,receiveMessage.lastIndexOf("ZYEOF"));
                                        /*if(receiveMessage.lastIndexOf("ZYEOF")!=-1){

                                            receiveMessage                                      =
"LastLine:"+receiveMessage.substring(0,receiveMessage.lastIndexOf("ZYEOF"));
                                            Message msg = new Message();
                                            msg.what = 0x1981;
                                            mHandler.sendMessage(msg);
                                        }
                                        else
                                        {
                                            receiveMessage = receiveMessage.substring(0,7);
                                            Message msg = new Message();
                                            msg.what = 0x1981;
                                            mHandler.sendMessage(msg);
                                        }*/
```

```
if(writeStringToFile(pathReceive+"/"+fileToReceiveName,receiveMessage.substring(6),null))
                                        {


receiveMessage="Done!"+pathReceive+"/"+fileToReceiveName;

                                        Message msg = new Message();
                                        msg.what = 0x1981;
                                        mHandler.sendMessage(msg);}
                                        else
                                        {

//receiveMessage="False!"+pathReceive+"/"+fileToReceiveName;
                                            //msg = new Message();
                                            //msg.what = 0x1981;
                                            //mHandler.sendMessage(msg);
                                        }
                                    }
                                    else if(mStrMSG.trim().substring(3,6).equals("EOF"))
                                    {
                                        String temp = new String(tempbyte);
                                        receiveMessage = temp.trim();
                                        receiveMessage = "LastLineEOF:";
                                        Message msg = new Message();
                                        msg.what = 0x1981;
                                        mHandler.sendMessage(msg);
                                    }
                                }
                                //gText.setText(gText.getText()+"\n"+mStrMSG);

                            }
                        }
                    }
                    catch(IOException e)
                    {
                        e.printStackTrace();
                        myDialog.append("\n"+e.toString());
                    }
                }
            }

            public String sendFileRead(String path){
```

```
        String data = "";
        try {

            FileInputStream input = new FileInputStream(new File(path));
            StringBuffer sb = new StringBuffer();
            int c;
            while((c = input.read()) != -1)
            {
                sb.append((char)c);
            }
            input.close();
            data = sb.toString()+"\n";
        } catch (FileNotFoundException e1) {
            // TODO Auto-generated catch block
            //editText_showIP.setText(e1.toString()+"not found");
            receiveMessage = e1.toString()+"not found";
            Message msg = new Message();
            msg.what = 0x1981;
            mHandler.sendMessage(msg);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            //editText_showIP.append("/n"+e.toString()+"not found");
            receiveMessage = e.toString();
            Message msg = new Message();
            msg.what = 0x1981;
            mHandler.sendMessage(msg);
        }
        catch(Exception e){
            receiveMessage = e.toString();
            Message msg = new Message();
            msg.what = 0x1981;
            mHandler.sendMessage(msg);
        }
        return data;
    }

    public void sendFileBySocket(String data, Socket destination){
        if(destination!=null){
            try {
                mPrintWriter=new PrintWriter(destination.getOutputStream(),true);

                mPrintWriter.print("FILSND\n");
                //receiveMessage = "FILACK";
                //Message msg= new Message();
```

```
                    //msg.what = 0x1981;
                    //mHandler.sendMessage(msg);
                    mPrintWriter.flush();

                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    receiveMessage = e.toString();
                    Message msg= new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);

                }

                }
                else{
                    receiveMessage = "no childSocket";
                    Message msg= new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);
                }

    }

    Handler mHandler = new Handler()
    {
        public void handleMessage(Message msg)
        {
            switch(msg.what){
            case 0x1981:
                guestDialog.append("\n"+receiveMessage);
                //guestDialog.setText(guestDialog.getText()+"\n"+receiveMessage);
                break;
            case 0x1982:
                myDialog.append("\n"+clientIP+":log in!");
                //myDialog.setText(myDialog.getText()+"\n"+clientIP+":log in!");
                break;
            case 0x1983:
                myDialog.setText("\n"+errorMessage);
                //myDialog.setText(myDialog.getText()+"\n"+errorMessage);
                break;
            case 0x1984:
                jumpToLayoutReceive(receiveMessage,clientReceive);
                myDialog.append("\n"+receiveMessage.substring(6));
                break;
```

```
            case 0x1985:
                jumpToMainLayer();
                myDialog.append("\n"+receiveMessage);
                break;
            case 0x1986:
                break;
            }

        }
    };

    public void jumpToLayoutReceive(String FileName, Socket childSocket){
        setContentView(R.layout.openfilereceive);
        layoutOpenFileReceiveInit(FileName,childSocket);
    }

    Button backReceive;
    Button cancelReceive;
    Button confirmReceive;
    ListView mListViewReceive;
    String pathReceive;
    File fileReceive;
    String fileToReceive;
    String fileToReceiveName;
    Socket childSocketReceive;

    public void layoutOpenFileReceiveInit(String FileName, Socket childSocket){
        backReceive = (Button)findViewById(R.id.buttonbackFileReceive);
        cancelReceive = (Button)findViewById(R.id.buttonCancelFileReceive);
        confirmReceive = (Button)findViewById(R.id.buttonConFirmReceive);
        mListViewReceive = (ListView)findViewById(R.id.listView1Receive);

        pathReceive = Environment.getRootDirectory().getParent();
        fileToReceiveName = FileName;
        showFilesReceive();
        childSocketReceive = childSocket;
        backReceive.setOnClickListener(new OnClickListener()
        {

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                if((fileReceive.getParent())!=null)
                {
                    pathReceive = fileReceive.getParent();
```

```
                        //m_EditText.setText(path);
                        showFilesReceive();
                    }
                    else
                    {
                        //m_EditText.setText(path);

                        showFilesReceive();
                    }
                }

            });

            cancelReceive.setOnClickListener(new OnClickListener(){

                public void onClick(View v) {
                    // TODO Auto-generated method stub
                    //
                    jumpToMainLayer();
                    sendFileACK(false,childSocketReceive);

                }
            }
            );

            confirmReceive.setOnClickListener(new OnClickListener(){

                public void onClick(View arg0) {
                    // TODO Auto-generated method stub

                    LayoutInflater factory = LayoutInflater.from(CheckIPaddressActivity.this);
                    fileToReceive = pathReceive;
                    File f = new File(pathReceive);
                    //fileToReceiveName = fileReceive.listFiles()[arg2].getName();
                    fileToReceive = f.getParent();
                    final View DialogView = factory.inflate(R.layout.rename, null);
                    AlertDialog dlg = new AlertDialog.Builder(CheckIPaddressActivity.this)
                    .setTitle("确定接收文件"+fileToReceiveName+"至:"+pathReceive).setView(DialogView).setPositiveButton("确定",
                    new DialogInterface.OnClickListener() {

                        public void onClick(DialogInterface dialog, int which) {
                            // TODO Auto-generated method stub
                            //FileResourceActivity.this.finish();
```

```
                        //dialog.cancel();
                        //SendFileReq(fileToReceiveName);
                        //jumpToMainLayer();
                        jumpToMainLayer();
                        sendFileACK(true,childSocketReceive);

                    }
                }).setNegativeButton("取消",
                new DialogInterface.OnClickListener()
                {
                    public void onClick(DialogInterface dialog, int which) {
                        // TODO Auto-generated method stub
                        //FileResourceActivity.this.finish();
                        fileToReceive = null;
                        dialog.cancel();
                    }
                }).create();
                dlg.show();


            }

        });

    }

    public void sendFileACK(boolean answer,Socket childSocket){
        if(childSocket!=null){
        try {
            mPrintWriter=new PrintWriter(childSocket.getOutputStream(),true);
            if(answer)
            {
                mPrintWriter.print("FILACK\n");
                receiveMessage = "FILACK";
                Message msg= new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);
            }
            else
            {
                mPrintWriter.print("FILREF\n");
                receiveMessage = "FILREF";
                Message msg= new Message();
                msg.what = 0x1981;
```

```
                    mHandler.sendMessage(msg);
                }
                mPrintWriter.flush();

            } catch (IOException e) {
                // TODO Auto-generated catch block
                receiveMessage = e.toString();
                Message msg= new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);

            }

            }
            else{
                receiveMessage = "no childSocket";
                Message msg= new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);
            }
        }



        public void showFilesReceive()
        {
            fileReceive = new File(pathReceive);


            IconifiedTextListAdapter iTLA = new IconifiedTextListAdapter(this);
            //int l = Environment.getRootDirectory().list().length;
            try
            {
                if(fileReceive.list()!=null)
                {
                    for(int i =0;i<fileReceive.list().length;i++){
                        IconifiedText iT = new IconifiedText(fileReceive.listFiles()[i].getName());
                        //IconifiedTextView iTV = new IconifiedTextView(this,iT);
                        //m_FileListView.
                        //iTV.setText(file.listFiles()[i].getName());
                        //iTV.getView();
                        iTLA.addItem(iT);
                        //
                        iTLA.getView(i, null, mListViewReceive);
```

```
                        //iTLA.addItem(iT);
                    }

                    mListViewReceive.setAdapter(iTLA);

                    mListViewReceive.setOnItemClickListener(new
OnItemClickListenerImplReceive());
                }
            }
            catch(Exception ex){
                //m_EditText.setText(ex.toString());
            }
        }

        private class OnItemClickListenerImplReceive implements OnItemClickListener{
            @SuppressWarnings("unchecked")
            public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,
                    long arg3) {
                // TODO Auto-generated method stub
                if(fileReceive.listFiles()!=null){

                    //m_EditText.setText(path);
                    if(!fileReceive.listFiles()[arg2].isFile())
                    {
                        pathReceive = fileReceive.listFiles()[arg2].getPath();
                        showFilesReceive();
                    }
                    else
                    {




                    }
                }

                }
        }

        public boolean writeStringToFile(String fileName, String content,
                String enc) {
                File file = new File(fileName);

                try {
                if (file.isFile()) {
```

```
                    //file.deleteOnExit();
                    //file = new File(file.getAbsolutePath());
                    }
                    DataOutputStream    os = null;
                    if (enc == null || enc.length() == 0) {
                    os = new DataOutputStream(new FileOutputStream(file));
                    } else {
                    //os = new DataOutputStream(new FileOutputStream(file), enc);
                    }
                    //os.write(content);

                    os.writeBytes(content);
                    os.close();
                    } catch (Exception e) {
                    e.printStackTrace();
                    receiveMessage = e.toString();
                    Message msg = new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);
                    return false;
                    }
                    return true;
                    }

}
```

Main.xml(client)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />




    <EditText
        android:id="@+id/editTextIP"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
        android:ems="10"
        android:text="192.168.43.58" >

        <requestFocus />
    </EditText>


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="CheckIP" />

        <Button
            android:id="@+id/Button_In"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="LogIn" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.37" >



        <EditText
            android:id="@+id/EditText01"
            android:layout_width="125dp"
            android:layout_height="match_parent"
            android:layout_weight="0.03"
            android:ems="10"
            android:gravity="top"
            android:inputType="textMultiLine"
            android:scrollbars="vertical" />
```

```
        <EditText
            android:id="@+id/editText2"
            android:layout_width="158dp"
            android:layout_height="match_parent"
            android:layout_weight="0.01"
            android:ems="10"
            android:gravity="top"
            android:inputType="textMultiLine"
            android:scrollbars="vertical" />

    </LinearLayout>


    <EditText
        android:id="@+id/EditText02"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/Button_Send"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Send" />

        <Button
            android:id="@+id/buttonSendFile"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Send Files" />

    </LinearLayout>

</LinearLayout>
```

Activity(client)

```
package my.client.namespace;

import java.io.BufferedReader;
```

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.InetSocketAddress;
import java.net.Socket;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

//import Dr.IPadd.CheckIPaddressActivity.ThreadServer;



import android.R.string;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.net.wifi.WifiInfo;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.os.Message;
import android.text.Editable;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.AdapterView.OnItemClickListener;







public class ClientActivity extends Activity {
    /** Called when the activity is first created. */
    Button Button_1;
```

```
EditText text;
String ip="";
private final String    DEBUG_TAG = "Activity01";
String SERVERIP="";
private static final int SERVERPORT = 50500;
private Thread mThread = null;
private Socket mSocket = null;
private Button mButton_In = null;
private Button mButton_Send = null;
private EditText mEditText01 = null;
private EditText mEditText02 = null;
EditText iptext;
private BufferedReader mBufferedReader = null;
private PrintWriter mPrintWriter = null;
private String mStrMSG ="";
private InetSocketAddress isa = null;

private String receiveMessage="";

private ExecutorService mExecutorService;

//private ListView m_FileListView;

private Button buttonSendFile;

@Override
public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
      SERVERIP = new String("192.168.1.115");
      mExecutorService = Executors.newCachedThreadPool();
      layoutMainInit();
      /*Button_1 = (Button) this.findViewById(R.id.button1);
      iptext = (EditText)this.findViewById(R.id.editTextIP);


      mExecutorService = Executors.newCachedThreadPool();

      WifiManager wifiManager=(WifiManager)getSystemService(Context.WIFI_SERVICE);
      if(!wifiManager.isWifiEnabled()){
            wifiManager.setWifiEnabled(true);
      }
      WifiInfo wifiInfo=wifiManager.getConnectionInfo();
      int ipAddress=wifiInfo.getIpAddress();
```

```
        ip=intToIp(ipAddress);


        Button_1.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v){
                iptext.setText(ip);
            }
        });

        mButton_In = (Button) this.findViewById(R.id.Button_In);
        mButton_Send = (Button) this.findViewById(R.id.Button_Send);
        mEditText01=(EditText)this.findViewById(R.id.EditText01);
        mEditText02=(EditText)this.findViewById(R.id.EditText02);
        text=(EditText)this.findViewById(R.id.editText2);
        SERVERIP = new String("192.168.1.115");

        mEditText02.setText("hello1");
        mButton_In.setOnClickListener(new OnClickListener()
        {
            public void onClick(View v)
            {
                try
                {

                    if(!iptext.getText().equals("")){
                        SERVERIP = iptext.getText().toString();
                    }
                    mSocket=new Socket();
                    isa = new InetSocketAddress(SERVERIP, 50500);
                    mSocket.connect(isa, 5000);

                    //mBufferedReader = new BufferedReader(new
InputStreamReader(mSocket.getInputStream()));

                    mEditText01.setText("log in success");
                    //mRunnable.run();
                    mExecutorService.execute(new ThreadServer(mSocket));
                }
                catch (Exception e)
                {
                    Log.e(DEBUG_TAG, e.toString());
                    mEditText01.setText("Log in to "+SERVERIP+"failed!\n"+e.toString());

                }
```

```
                }
            });

        mButton_Send.setOnClickListener(new OnClickListener()
        {
            public void onClick(View v)
            {
            try
            {
                String str = mEditText02.getText().toString();
                mPrintWriter=new PrintWriter(mSocket.getOutputStream(),true);
                mPrintWriter.print("MSG"+str+"\n");
                mPrintWriter.flush();
                mEditText01.append("\n"+str);
            }
            catch (Exception e)
            {
                Log.e(DEBUG_TAG, e.toString());
                receiveMessage = "rune"+e.toString();
                Message msg = new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);
            }
        }
    });*/
        //mThread = new Thread(mRunnable);
        //mThread.start();

    }

    public void layoutMainInit()
    {
        Button_1 = (Button) this.findViewById(R.id.button1);
        iptext = (EditText)this.findViewById(R.id.editTextIP);
        WifiManager wifiManager=(WifiManager)getSystemService(Context.WIFI_SERVICE);
        if(!wifiManager.isWifiEnabled()){
            wifiManager.setWifiEnabled(true);
        }
        WifiInfo wifiInfo=wifiManager.getConnectionInfo();
        int ipAddress=wifiInfo.getIpAddress();
        ip=intToIp(ipAddress);
        Button_1.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v){
                iptext.setText(ip);
```

```
            }
        });
        mButton_In = (Button) this.findViewById(R.id.Button_In);
        mButton_Send = (Button) this.findViewById(R.id.Button_Send);
        mEditText01=(EditText)this.findViewById(R.id.EditText01);
        mEditText02=(EditText)this.findViewById(R.id.EditText02);
        text=(EditText)this.findViewById(R.id.editText2);
        mEditText02.setText("hello1");
        buttonSendFile = (Button)this.findViewById(R.id.buttonSendFile);
        mButton_In.setOnClickListener(new OnClickListener()
        {
            public void onClick(View v)
            {
                try
                {

                    if(!iptext.getText().equals("")){
                        SERVERIP = iptext.getText().toString();
                    }
                    mSocket=new Socket();
                    isa = new InetSocketAddress(SERVERIP, 50500);
                    mSocket.connect(isa, 5000);

                    //mBufferedReader = new BufferedReader(new
InputStreamReader(mSocket.getInputStream()));

                    mEditText01.setText("log in success");
                    //mRunnable.run();
                    mExecutorService.execute(new ThreadServer(mSocket));
                }
                catch (Exception e)
                {
                    Log.e(DEBUG_TAG, e.toString());
                    mEditText01.setText("Log in to "+SERVERIP+"failed!\n"+e.toString());

                }
                }
            });

        mButton_Send.setOnClickListener(new OnClickListener()
        {
            public void onClick(View v)
            {
            try
```

```
            {
                String str = mEditText02.getText().toString();
                mPrintWriter=new PrintWriter(mSocket.getOutputStream(),true);
                mPrintWriter.print("MSG"+str+"\n");
                mPrintWriter.flush();
                mEditText01.append("\n"+str);
            }
            catch (Exception e)
            {
                Log.e(DEBUG_TAG, e.toString());
                receiveMessage = "rune"+e.toString();
                Message msg = new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);
            }
        }
    });
        buttonSendFile.setOnClickListener(new OnClickListener(){

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                //jumpToLayoutReceive("");
                //jumpToLayerFile();
                jumpToLayerFile();
                //SendFileReq("ssss");
            }

        });
    }

    ListView m_FileListView;
    private String path;
    File file;
    EditText m_EditText;
    Button buttonBack;
    Button buttonCancelFile;
    String fileToSend;
    String fileToSendName;

    public void jumpToLayerFile()
    {
        setContentView(R.layout.openfile);
        m_EditText = (EditText)findViewById(R.id.editTextFileName);
        buttonBack=(Button)findViewById(R.id.buttonback);
```

```
        //path = "123";
        buttonCancelFile = (Button)findViewById(R.id.buttonCancleFile);
        m_FileListView = (ListView)findViewById(R.id.listView1);

        path = Environment.getRootDirectory().getAbsolutePath();
        showFiles();
        buttonBack.setOnClickListener(new OnClickListener()
        {

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                if((file.getParent())!=null)
                {
                    path = file.getParent();
                    m_EditText.setText(path);
                    showFiles();
                }
                else
                {
                    m_EditText.setText(path);
                    showFiles();
                }
            }

        });

        buttonCancelFile.setOnClickListener(new OnClickListener(){

            public void onClick(View v) {
                // TODO Auto-generated method stub
                jumpToMainLayer();
            }
        }
        );

    }

    public void showFiles()
    {
        file = new File(path);


        IconifiedTextListAdapter iTLA = new IconifiedTextListAdapter(this);
        //int l = Environment.getRootDirectory().list().length;
```

```
        try
        {
            if(file.list()!=null)
            {
                for(int i =0;i<file.list().length;i++){
                    IconifiedText iT = new IconifiedText(file.listFiles()[i].getName());
                    //IconifiedTextView iTV = new IconifiedTextView(this,iT);
                    //m_FileListView.
                    //iTV.setText(file.listFiles()[i].getName());
                    //iTV.getView();
                    iTLA.addItem(iT);
                    //
                    iTLA.getView(i, null, m_FileListView);
                    //iTLA.addItem(iT);
                }

                    m_FileListView.setAdapter(iTLA);

                    m_FileListView.setOnItemClickListener(new OnItemClickListenerImpl());
            }
        }
        catch(Exception ex){
            m_EditText.setText(ex.toString());
        }
    }

    private class OnItemClickListenerImpl implements OnItemClickListener{
        @SuppressWarnings("unchecked")
        public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,
                long arg3) {
            // TODO Auto-generated method stub
            if(file.listFiles()!=null){
            path = file.listFiles()[arg2].getPath();
            m_EditText.setText(path);
            if(file.listFiles()[arg2].isDirectory())
            {

                showFiles();
            }
            else
            {
                LayoutInflater factory = LayoutInflater.from(ClientActivity.this);

                fileToSendName = file.listFiles()[arg2].getName();
```

```
                    fileToSend = file.listFiles()[arg2].getPath();
                    final View DialogView = factory.inflate(R.layout.rename, null);
                    AlertDialog dlg = new AlertDialog.Builder(ClientActivity.this)
                    .setTitle("确定传输文件:"+fileToSend).setView(DialogView).setPositiveButton("确定",
                    new DialogInterface.OnClickListener() {

                        public void onClick(DialogInterface dialog, int which) {
                            // TODO Auto-generated method stub
                            //FileResourceActivity.this.finish();
                            //dialog.cancel();
                            jumpToMainLayer();
                            if(fileToSendName==null)
                                SendFileReq("ssdd");
                            else
                                SendFileReq(fileToSendName);


                        }
                    }).setNegativeButton("取消",
                    new DialogInterface.OnClickListener()
                    {
                        public void onClick(DialogInterface dialog, int which) {
                            // TODO Auto-generated method stub
                            //FileResourceActivity.this.finish();
                            fileToSend = null;
                            dialog.cancel();
                        }
                    }).create();
                    dlg.show();

                }
            }}
        }

        public void SendFileReq(String FileName)
            {
                Socket childSocket = mSocket;
                if(mSocket!=null)
                {
                    try{
                        //PrintWriter mPrintWriter = new PrintWriter(childSocket.getOutputStream(),true);
                        //mPrintWriter.print("FILREQ"+FileName+"\n");
```

```
                        //mPrintWriter.flush();
                        mPrintWriter = new PrintWriter(childSocket.getOutputStream(),true);
                        mPrintWriter.print("FILREQ"+FileName+"\n");
                        mPrintWriter.flush();
                    }
                    catch(Exception e)
                    {

                    }
                }
            }


    private String intToIp(int i){
        return(i & 0xFF)+"."+((i>>8)&0xFF)+"."+((i>>16)&0xFF)+"."+(i>>24 & 0xFF);
    }

    class ThreadServer implements Runnable
    {
        private Socket mSocket;
        private BufferedReader mBufferedReader;
        private PrintWriter mPrintWriter;
        private String mStrMSG;


        public ThreadServer(Socket socket) throws IOException
        {
            this.mSocket = socket;
            mBufferedReader = new BufferedReader(new
InputStreamReader(mSocket.getInputStream()));
        }

        public void run()
        {
            try
            {
                int buffersize = mSocket.getReceiveBufferSize();

                while(true){
                    /*StringBuffer readbuffer = new StringBuffer();
                    int c;
                    while((c = mBufferedReader.read()) != -1)
                    {
                        readbuffer.append((char)c);
```

```
                        }
                        receiveMessage = readbuffer.toString();*/
                        char[] tempbyte = new char[buffersize];
                        mBufferedReader.read(tempbyte);
                        receiveMessage = String.valueOf(tempbyte).trim();
                        //receiveMessage = mBufferedReader.readLine().trim();
                        //if(receiveMessage.substring(0, 2))
                        Message msg = new Message();
                        //mSocket.getInetAddress()+":"
                        if(receiveMessage.substring(0,3).equals("MSG"))
                        {
                            receiveMessage                                                  =
mSocket.getInetAddress()+":"+receiveMessage.substring(3);
                            msg.what = 0x1981;
                            mHandler.sendMessage(msg);
                        }
                        else if(receiveMessage.substring(0,3).equals("FIL"))
                        {
                            if(receiveMessage.substring(3,6).equals("REQ"))
                            {
                                receiveMessage                                              =
mSocket.getInetAddress()+"REQFILETransmission:"+receiveMessage.substring(6);
                                //jumpToLayoutReceive(receiveMessage.substring(6));
                                msg.what = 0x1984;
                                mHandler.sendMessage(msg);

                            }
                            else if(receiveMessage.substring(3, 6).equals("ACK"))
                            {
                                receiveMessage = "开始传输";
                                msg.what = 0x1981;
                                mHandler.sendMessage(msg);
                                sendFileBySocket(sendFileRead(path),mSocket);
                                //sendFileRead(path);

                                //receiveMessage = "传输完成";
                                //msg.what = 0x1981;
                                //mHandler.sendMessage(msg);
                            }
                            else if(receiveMessage.substring(3, 6).equals("REF"))
                            {
                                receiveMessage = mSocket.getInetAddress()+"拒绝接收";
```

```
                                msg.what = 0x1981;
                                mHandler.sendMessage(msg);
                            }
                            else if(receiveMessage.substring(3, 6).equals("SND"))
                            {

                            }
                            else
                            {
                                msg.what = 0x1981;
                                mHandler.sendMessage(msg);

                            }
                        }
                        else
                        {
                            receiveMessage = receiveMessage.substring(0,3)+"!!";
                            msg.what = 0x1981;
                            mHandler.sendMessage(msg);
                        }
                    }
                }
                catch(Exception e)
                {
                    //e.printStackTrace();
                    //text.append(e.toString());
                    receiveMessage = e.toString();
                    Message msg = new Message();
                    msg = new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);
                }
            }
        }

public String sendFileRead(String path){

            String data = "";
            try {

                FileInputStream input = new FileInputStream(new File(path));
                StringBuffer sb = new StringBuffer();
                sb.ensureCapacity((int)new File(path).length());
                int c;
```

```
            while((c = input.read()) != -1)
            {
                sb.append((char)c);
                //if(sb.length()>=sb.capacity()-5){
                //    data = data+sb.toString();
                //    sb = new StringBuffer();
                //}
                /*if(sb.length()==65518){
                    data=sb.toString();
                    sb = new StringBuffer();
                    mPrintWriter=new PrintWriter(mSocket.getOutputStream(),true);
                    mPrintWriter.print("FILSND"+data);
                    mPrintWriter.flush();
                    receiveMessage = data.substring(0,1);
                    Message msg = new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);
                }*/

            }
            input.close();
            //data = data+sb.toString();
            data=sb.toString();

            /*mPrintWriter=new PrintWriter(mSocket.getOutputStream(),true);
            mPrintWriter.print("FILSND"+data);
            mPrintWriter.flush();
            receiveMessage = data.substring(0,1)+"ZYEOF";
            Message msg = new Message();
            msg.what = 0x1981;
            mHandler.sendMessage(msg);*/


        } catch (FileNotFoundException e1) {
            // TODO Auto-generated catch block
            //editText_showIP.setText(e1.toString()+"not found");
            receiveMessage = e1.toString()+"not found";
            Message msg = new Message();
            msg.what = 0x1981;
            mHandler.sendMessage(msg);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            //editText_showIP.append("/n"+e.toString()+"not found");
```

```
                receiveMessage = e.toString();
                Message msg = new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);
            }
            catch(Exception e){
                receiveMessage = e.toString();
                Message msg = new Message();
                msg.what = 0x1981;
                mHandler.sendMessage(msg);
            }
            return data;
        }

        public void sendFileBySocket(String data, Socket destination){
            if(destination!=null){
                try {
                    mPrintWriter=new PrintWriter(destination.getOutputStream(),true);

                    mPrintWriter.print("FILSND"+data+"ZYEOF\n");
                    //receiveMessage = "FILACK";
                    //Message msg= new Message();
                    //msg.what = 0x1981;
                    //mHandler.sendMessage(msg);
                    mPrintWriter.flush();
                    receiveMessage = "传输完成\n";
                    Message msg = new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);

                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    receiveMessage = e.toString();
                    Message msg= new Message();
                    msg.what = 0x1981;
                    mHandler.sendMessage(msg);

                }

                }
                else{
                    receiveMessage = "no childSocket";
                    Message msg= new Message();
                    msg.what = 0x1981;
```

```
                mHandler.sendMessage(msg);
            }

    }

    private Runnable mRunnable= new Runnable()
    {
        public void run()
        {
            while(true)
            {
                try
                {
                    if((mStrMSG = mBufferedReader.readLine())!=null)
                    {
                        mStrMSG+="\n";
                        mHandler.sendMessage(mHandler.obtainMessage());
                    }
                    Thread.sleep(100);
                }
                catch (Exception e)
                {
                    Log.e(DEBUG_TAG, e.toString());
                }
            }
        }
    };

    Handler mHandler = new Handler()
    {
        public void handleMessage(Message msg)
        {
            switch(msg.what){
            case 0x1981:
                text.append("\n"+receiveMessage);
                //guestDialog.append("\n"+receiveMessage);
                //guestDialog.setText(guestDialog.getText()+"\n"+receiveMessage);
                break;
            case 0x1982:
                //myDialog.append("\n"+clientIP+":log in!");
                //myDialog.setText(myDialog.getText()+"\n"+clientIP+":log in!");
                break;
            case 0x1983:
                //myDialog.setText("\n"+errorMessage);
```

```
                //myDialog.setText(myDialog.getText()+"\n"+errorMessage);
                break;
            case 0x1984:
                jumpToLayoutReceive(receiveMessage.substring(6));
                break;
            case 0x1985:
                jumpToMainLayer();
                text.append("\n"+receiveMessage);
                break;
            case 0x1986:
                break;
            }

        }
    };


    public void jumpToLayoutReceive(String FileName){
        setContentView(R.layout.openfilereceive);
        layoutOpenFileReceiveInit(FileName);
    }

    Button backReceive;
    Button cancelReceive;
    Button confirmReceive;
    ListView mListViewReceive;
    String pathReceive;
    File fileReceive;
    String fileToReceive;
    String fileToReceiveName;

    public void layoutOpenFileReceiveInit(String FileName){
        backReceive = (Button)findViewById(R.id.buttonbackFileReceive);
        cancelReceive = (Button)findViewById(R.id.buttonCancelFileReceive);
        confirmReceive = (Button)findViewById(R.id.buttonConFirmReceive);
        mListViewReceive = (ListView)findViewById(R.id.listView1Receive);

        pathReceive = Environment.getRootDirectory().getParent();
        showFilesReceive();

        backReceive.setOnClickListener(new OnClickListener()
        {

            public void onClick(View arg0) {
```

```
                    // TODO Auto-generated method stub
                    if((fileReceive.getParent())!=null)
                    {
                        pathReceive = fileReceive.getParent();
                        //m_EditText.setText(path);
                        showFilesReceive();
                    }
                    else
                    {
                        //m_EditText.setText(path);

                        showFilesReceive();
                    }
                }

            });

            cancelReceive.setOnClickListener(new OnClickListener(){

                public void onClick(View v) {
                    // TODO Auto-generated method stub
                    //
                    sendFileACK(false);
                    jumpToMainLayer();
                }
            }
            );

            confirmReceive.setOnClickListener(new OnClickListener(){

                public void onClick(View arg0) {
                    // TODO Auto-generated method stub

                    LayoutInflater factory = LayoutInflater.from(ClientActivity.this);
                    fileToReceive = pathReceive;
                    File f = new File(pathReceive);
                    //fileToReceiveName = fileReceive.listFiles()[arg2].getName();
                    fileToReceive = f.getParent();
                    final View DialogView = factory.inflate(R.layout.rename, null);
                    AlertDialog dlg = new AlertDialog.Builder(ClientActivity.this)
                    .setTitle("确定接收文件至:"+pathReceive).setView(DialogView).setPositiveButton("确定",
                    new DialogInterface.OnClickListener() {
```

```
                        public void onClick(DialogInterface dialog, int which) {
                            // TODO Auto-generated method stub
                            //FileResourceActivity.this.finish();
                            //dialog.cancel();
                            //SendFileReq(fileToReceiveName);
                            //jumpToMainLayer();
                            sendFileACK(true);
                            jumpToMainLayer();
                        }
                    }).setNegativeButton("取消",
                    new DialogInterface.OnClickListener()
                    {
                        public void onClick(DialogInterface dialog, int which) {
                            // TODO Auto-generated method stub
                            //FileResourceActivity.this.finish();
                            fileToReceive = null;
                            dialog.cancel();
                        }
                    }).create();
                    dlg.show();


                }

            });

        }

        public void jumpToMainLayer(){
            setContentView(R.layout.main);
            layoutMainInit();
        }

        public void sendFileACK(boolean answer){
            if(mSocket!=null){
            try {
                mPrintWriter=new PrintWriter(mSocket.getOutputStream(),true);
                if(answer)
                    mPrintWriter.print("FILACK\n");
                else
                    mPrintWriter.print("FILREF\n");
                mPrintWriter.flush();

            } catch (IOException e) {
```

```
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

            }
        }

        public void showFilesReceive()
        {
            fileReceive = new File(pathReceive);


            IconifiedTextListAdapter iTLA = new IconifiedTextListAdapter(this);
            //int l = Environment.getRootDirectory().list().length;
            try
            {
                if(fileReceive.list()!=null)
                {
                    for(int i =0;i<fileReceive.list().length;i++){
                        IconifiedText iT = new IconifiedText(fileReceive.listFiles()[i].getName());
                        //IconifiedTextView iTV = new IconifiedTextView(this,iT);
                        //m_FileListView.
                        //iTV.setText(file.listFiles()[i].getName());
                        //iTV.getView();
                        iTLA.addItem(iT);
                        //
                        iTLA.getView(i, null, mListViewReceive);
                        //iTLA.addItem(iT);
                    }

                    mListViewReceive.setAdapter(iTLA);

                    mListViewReceive.setOnItemClickListener(new
OnItemClickListenerImplReceive());
                }
            }
            catch(Exception ex){
                //m_EditText.setText(ex.toString());
            }
        }

        private class OnItemClickListenerImplReceive implements OnItemClickListener{
            @SuppressWarnings("unchecked")
            public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,
```

```
                    long arg3) {
                // TODO Auto-generated method stub
                if(fileReceive.listFiles()!=null){

                    //m_EditText.setText(path);
                    if(!fileReceive.listFiles()[arg2].isFile())
                    {
                        pathReceive = fileReceive.listFiles()[arg2].getPath();
                        showFilesReceive();
                    }
                    else
                    {



                    }
                }

                }
        }


}
```