Introduction to Android TCP/IP Socket Client

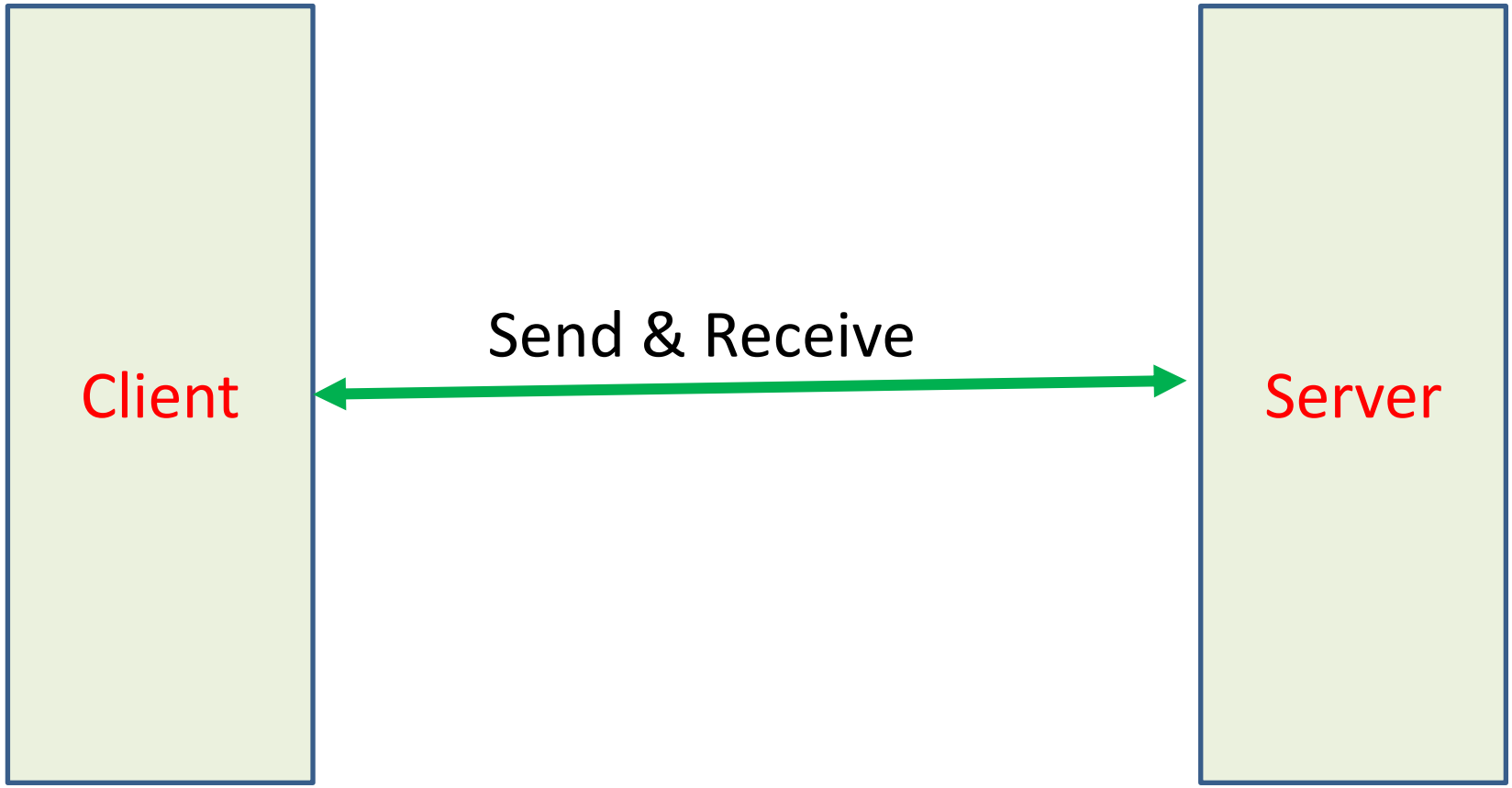
CS 436 Software Development on Mobile

Dr.Paween Khoenkaw

Department of Computer Science
Maejo University

Socket Client

Socket Client



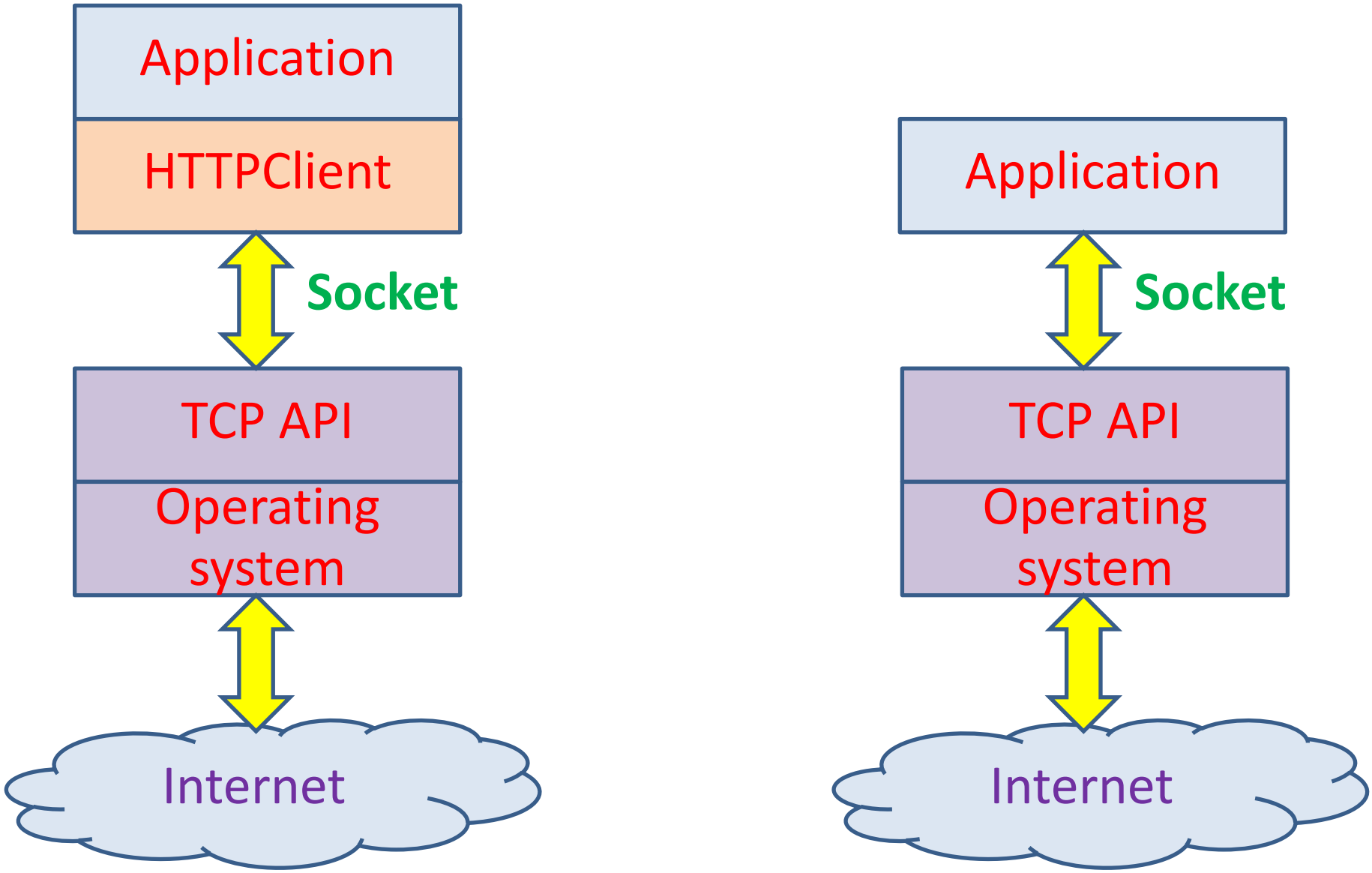
Socket Client

What is a socket?

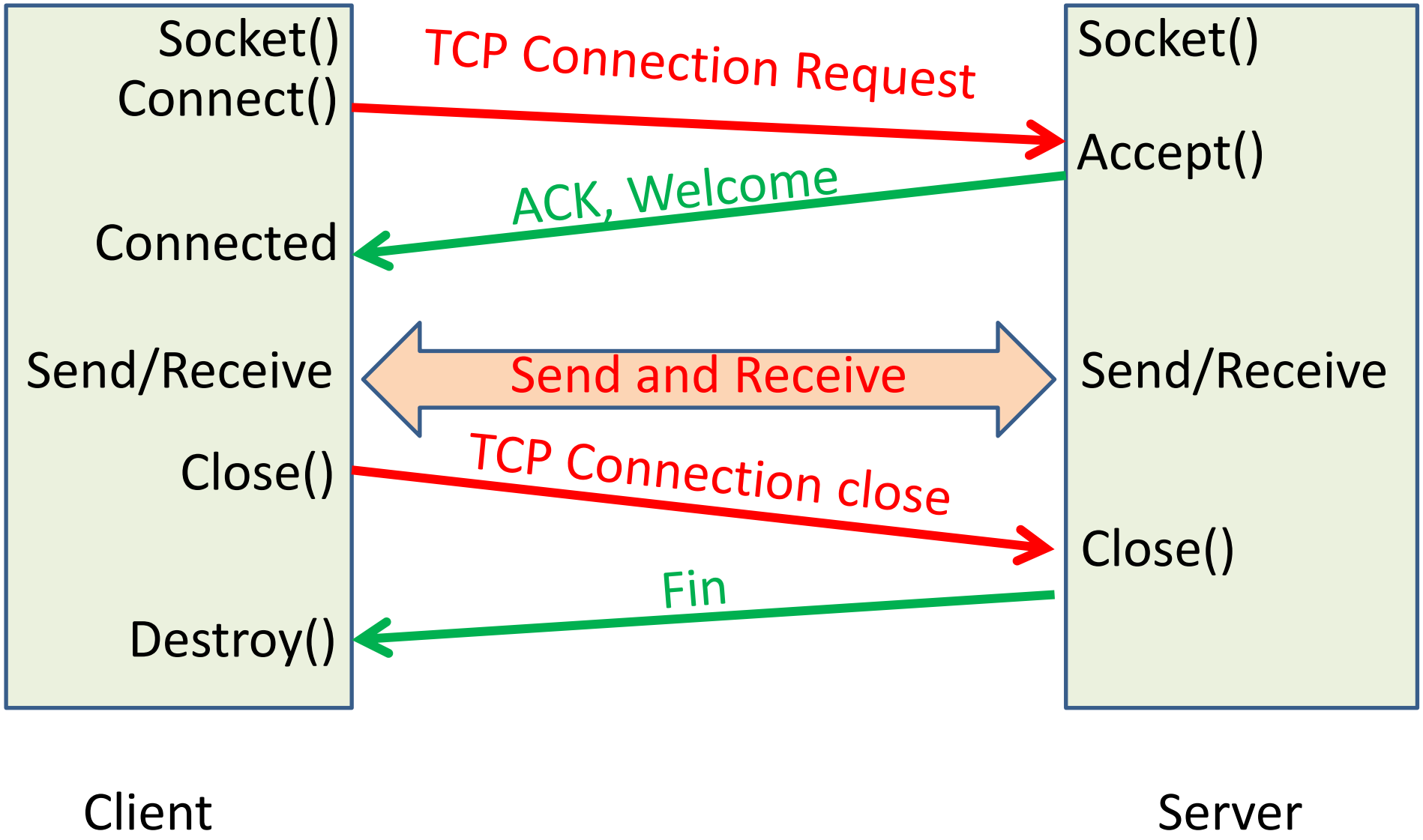
- Socket
  - The combination of an IP address and a port number. (RFC 793 ,original TCP specification)
  - The name of the Berkeley-derived *application programming interfaces* (APIs) for applications using TCP/IP protocols.
  - Two types
    - Stream socket : reliable two-way connected communication streams
    - Datagram socket

Socket is IP Address + Port

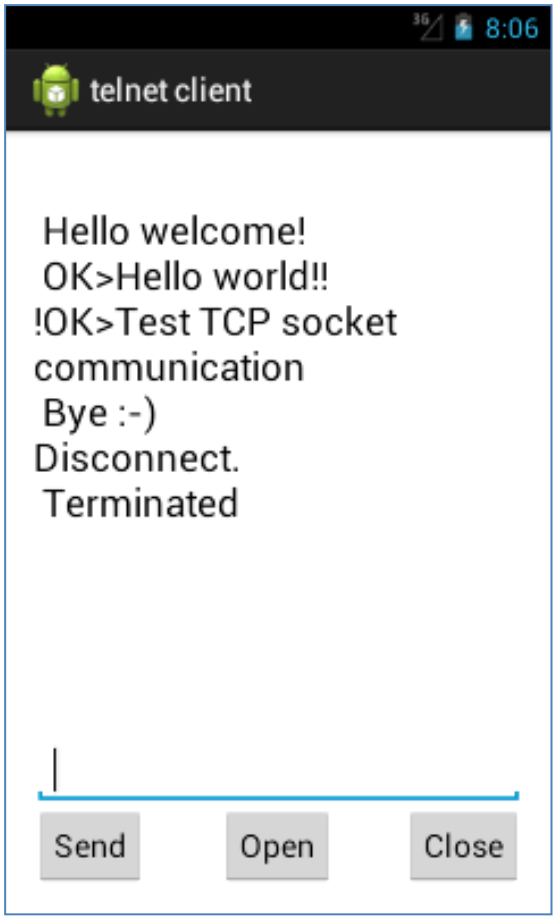
Socket Client

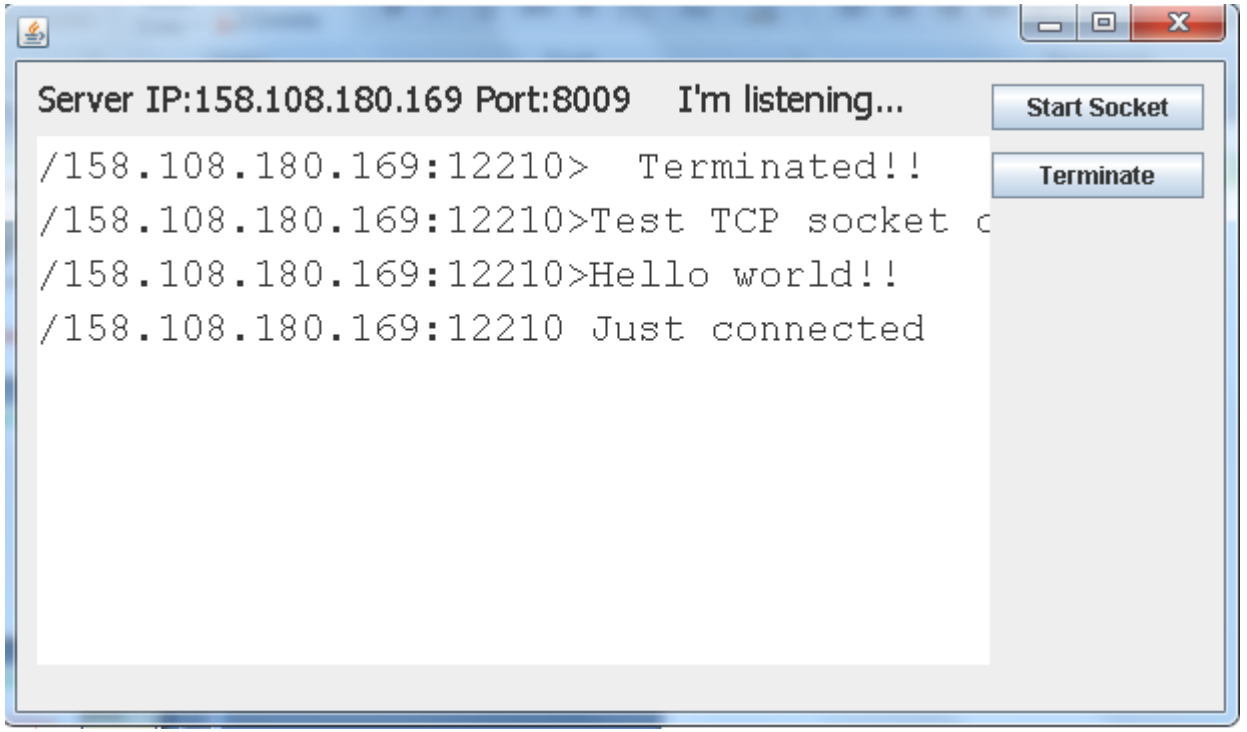


Socket Client



Socket Client





Project: android_communication_tcpclient

Socket Client

Create TCP Socket client class

Methods

Connect()

CloseConnection()

SendMessage()

Events

OnMessageReceived()

Disconnect()

Socket Client

```
public class TCP_Client {
    public static final String SERVERIP = "158.108.180.169"; //your server IP address
    public static final int SERVERPORT = 8009;
    private OnMessageReceived mMessageListener = null;
    PrintWriter out;
    BufferedReader in;
    Socket socket;
    public TCP_Client( OnMessageReceived listener ) {        // constructor
                            mMessageListener = listener;             }

    public void sendMessage(String message){// send data to server
    }

    public void CloseConnection(){      // close connection
    }
                public interface OnMessageReceived {
                    public void messageReceived(String message);
                    public void Disconnect();
                 }
                public void Connect() {// start socket and wait for incoming messages
                }
}
```

Socket Client

```
public void Connect() {
try {
InetAddress serverAddr = InetAddress.getByName(SERVERIP);
socket = new Socket(serverAddr, SERVERPORT);
out = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream())), true);
 in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
 while (true) {
  String msgFromServer=in.readLine();
    if(msgFromServer==null)
      {   // disconnect
            break;
       }
mMessageListener.messageReceived(msgFromServer);
                }
 mMessageListener.Disconnect();
 } catch (UnknownHostException e) {
} catch (IOException e) {
}
                              }
```

Block Loop

Socket Client

```
public void sendMessage(String message){
        if (out != null && !out.checkError()) {
            out.println(message);
            out.flush();
        }
    }

    public void CloseConnection(){
        try {
                                socket.close();
                        } catch (IOException e) {
                                e.printStackTrace();
                        }
    }
```

Socket Client

Main Thread

```
public class connectTask extends AsyncTask<Void,String,Void> {
        @Override
        protected void onPostExecute(Void result) {
        // disconnect
        }
        @Override
        protected void onProgressUpdate(String... values) {
        // received data
        }
        @Override
        protected Void doInBackground(Void... params) {
        // connect to server and wait for data
        return null;
        }
         }
```

Socket Client

Main Thread

```
@Override
protected Void doInBackground(Void... params) {
 mTCP_Client = new TCP_Client(new TCP_Client.OnMessageReceived() {
          @Override
           //here the messageReceived method is implemented
           public void messageReceived(String message) {
               publishProgress(message);
            }
          @Override
  public void Disconnect() {
             publishProgress("Disconnect.");
                                }
                    });
mTCP_Client.Connect();
return null;
                        }
```

Socket Client

Main Thread

```
@Override
protected void onPostExecute(Void result) {
super.onPostExecute(result);
textview1.setText(textview1.getText()+"\n Terminated");
                        }

@Override
protected void onProgressUpdate(String... values) {
textview1.setText(textview1.getText()+"\n"+values[0]);
super.onProgressUpdate(values);
                        }
```

Socket Client

GUI

Start connection

```
new connectTask().execute();
```

Send data

```
String msg=edittext1.getText().toString();
 if (mTCP_Client != null) {
    mTCP_Client.sendMessage(msg);
    edittext1.setText("");
         }
```

Start connection

```
mTCP_Client.CloseConnection();
```

```
<uses-permission
android:name="android.permission.INTERNET"/>
```