

## Lab 7: SOCKET



### MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Triển khai được cách kết nối giữa android(client) với server bằng SOCKET.

### BÀI 1 (5 ĐIỂM)

Viết chương trình đơn giản kết nối giữa 2 ứng dụng android, 1 android(client) và 1 android(server) bằng SOCKET. Yêu cầu phía Client gửi một đoạn text nào đó thì server sẽ nhận được và hiển thị lên textview.

### HƯỚNG DẪN:

- ✓ Đầu tiên ta sẽ code phía server project trước sau đó ta sẽ code phía client
- ✓ Mở android studio tạo project ứng dụng android mới. Đặt tên ứng dụng là AndroidSocketSERVER.
- ✓ Mở file activity\_main.xml và code như bên dưới

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:orientation="vertical"

    tools:context="com.example.android.lab7_androidsocketserver.MainActivity">

    <TextView
        android:id="@+id/tv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="" />
</LinearLayout>
```

- ✓ Để lập trình networking applications ta cần thêm các permission trong file AndroidManifest.xml như sau.

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

- ✓ Tiếp tục việc tạo server, ta mở source file Main Activity và code như bên dưới

```
package com.example.android.lab7_androidsocketserver;

import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.ServerSocket;
import java.net.Socket;

public class MainActivity extends AppCompatActivity {

    private ServerSocket serverSocket;

    Handler updateConversationHandler;

    Thread serverThread = null;

    private TextView text;

    public static final int SERVERPORT = 6000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        text = (TextView) findViewById(R.id.tv);

        updateConversationHandler = new Handler();

        this.serverThread = new Thread(new ServerThread());
        this.serverThread.start();
    }

    @Override
    protected void onStop() {
        super.onStop();
        try {
            serverSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    class ServerThread implements Runnable {
        @Override
```

```

        public void run() {
            Socket socket = null;
            try {
                serverSocket = new ServerSocket(SERVERPORT);
            } catch (IOException e) {
                e.printStackTrace();
            }
            while (!Thread.currentThread().isInterrupted()) {
                try {

                    socket = serverSocket.accept();

                    CommunicationThread commnThread = new
CommunicationThread(socket);
                    new Thread(commnThread).start();

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        class CommunicationThread implements Runnable {

            private Socket clientSocket;

            private BufferedReader input;

            public CommunicationThread(Socket clientSocket) {
                this.clientSocket = clientSocket;
                try {
                    this.input = new BufferedReader(new
InputStreamReader(this.clientSocket.getInputStream()));
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }

            @Override
            public void run() {
                while (!Thread.currentThread().isInterrupted()) {
                    try {
                        String read = input.readLine();

                        updateConversationHandler.post(new
UpdateUIThread(read));

                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
        }

        class UpdateUIThread implements Runnable {
            private String msg;

```

```

        public UpdateUIThread(String str) {
            this.msg = str;
        }

        @Override
        public void run() {
            text.setText(text.getText().toString() + "Client says: " +
msg + "\n");
        }
    }
}

```

- ✓ Bây giờ đến phần code cho phía Client Project
- ✓ Tạo project ứng dụng Android mới như đã tạo với Server Application. Đặt tên AndroidSocketCLIENT
- ✓ Mở file main layout và code giao diện như sau.

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"

    tools:context="com.example.android.lab7_androidsocketclient.MainActivity">
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/edtText"
        android:text="Android Learn Socket"
        />
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/btnActive"
        android:text="Send"
        />
</LinearLayout>

```

- ✓ Thêm permission trong file AndroidManifest.xml như sau

```

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

```

- ✓ Bây giờ mở source file MainActivity.java và tiến hành code cho phía client như sau :

```

package com.example.android.lab7_androidsocketclient;

```

## FPT POLYTECHNIC

```

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

import java.io.BufferedWriter;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.InetAddress;
import java.net.Socket;
import java.net.UnknownHostException;

public class MainActivity extends AppCompatActivity implements
View.OnClickListener {

    private Socket socket;

    private static final int SERVERPORT = 5000;
    private static final String SERVER_IP = "10.0.2.2";

    EditText edt;
    Button btnSend;

    String str;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        edt = (EditText) findViewById(R.id.edtText);
        btnSend = (Button) findViewById(R.id.btnActive);

        btnSend.setOnClickListener(this);

        new Thread(new ClientThread()).start();
    }

    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.btnActive:
                str = edt.getText().toString();
                try {
                    PrintWriter out = new PrintWriter(new BufferedWriter
                    OutputStreamWriter(socket.getOutputStream()), true);
                    out.println(str);
                } catch (UnknownHostException e) {
                    e.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                } catch (Exception e) {
                    e.printStackTrace();
                }
                break;

```

```

    }

    class ClientThread implements Runnable {

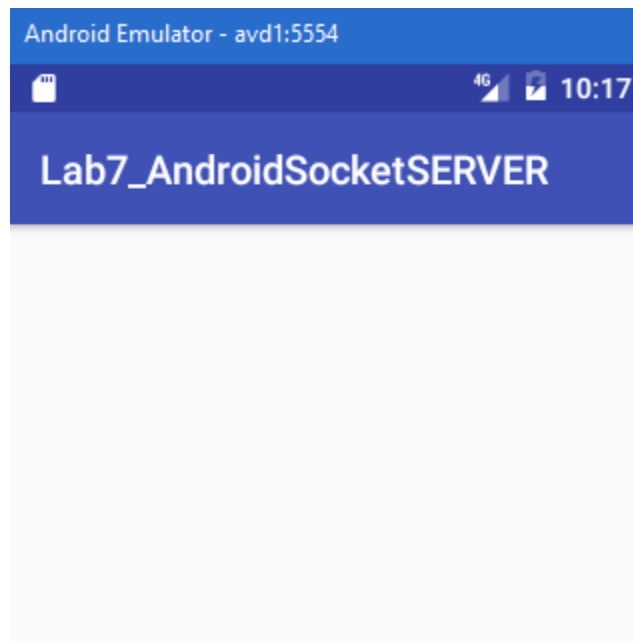
        @Override
        public void run() {
            try {
                InetAddress serverAddr = InetAddress.getByName(SERVER_IP);

                socket = new Socket(serverAddr, SERVERPORT);

            } catch (UnknownHostException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}

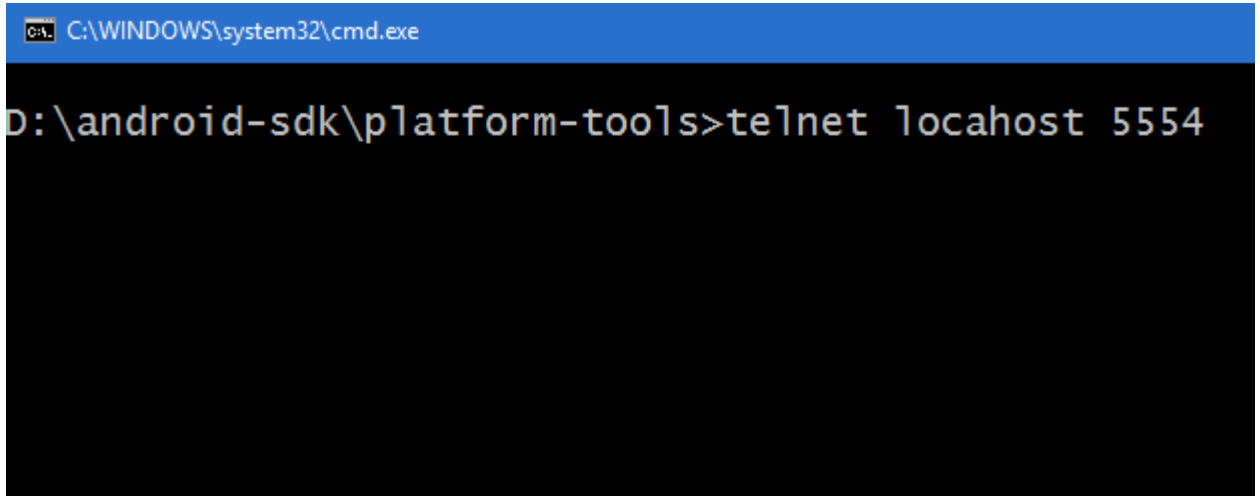
```

- ✓ Để liên kết chương trình trong 2 emulators khác nhau thì quá trình như sau
- ✓ Chương trình server sẽ mở ở port 6000 trên emulator A. điều đó có nghĩa là port 6000 mở trên ip của emulator có ip là 10.0.2.15.
- ✓ Bây giờ, client ở emulator B sẽ connect tới localhost, có IP là 10.0.2.2 ở port 5000
- ✓ Localhost sẽ chuyển tiếp các gói tin tới 10.0.2.15 : 6000
- ✓ Để làm vậy ta phải làm port forwarding trên máy ảo. bây giờ ta chạy chương trình Server, mở máy ảo đầu tiên.



- ✓ Tiếp tục, bây giờ ta sẽ access console của emulator này ở localhost: 5554, mở cmd. Để kết nối ta phải làm lệnh sau:

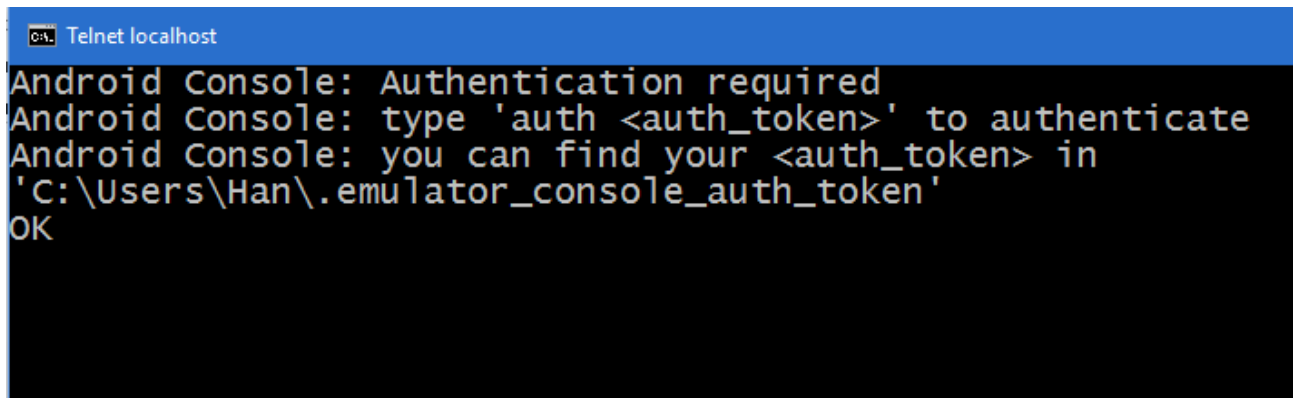
```
telnet localhost 5554
```



```
C:\WINDOWS\system32\cmd.exe

D:\android-sdk\platform-tools>telnet localhost 5554
```

OK

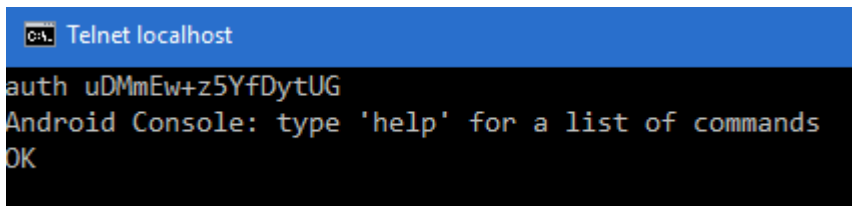


```
C:\> Telnet localhost

Android Console: Authentication required
Android Console: type 'auth <auth_token>' to authenticate
Android Console: you can find your <auth_token> in
'C:\Users\Han\.emulator_console_auth_token'
OK
```

Gõ tiếp

auth<auth\_token> chú ý auth token nằm ở file  
.emulator\_console\_auth\_token



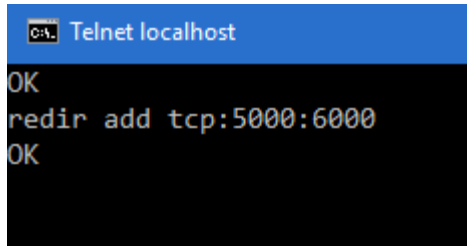
```
C:\> Telnet localhost

auth uDMmEw+z5YfDytUG
Android Console: type 'help' for a list of commands
OK
```

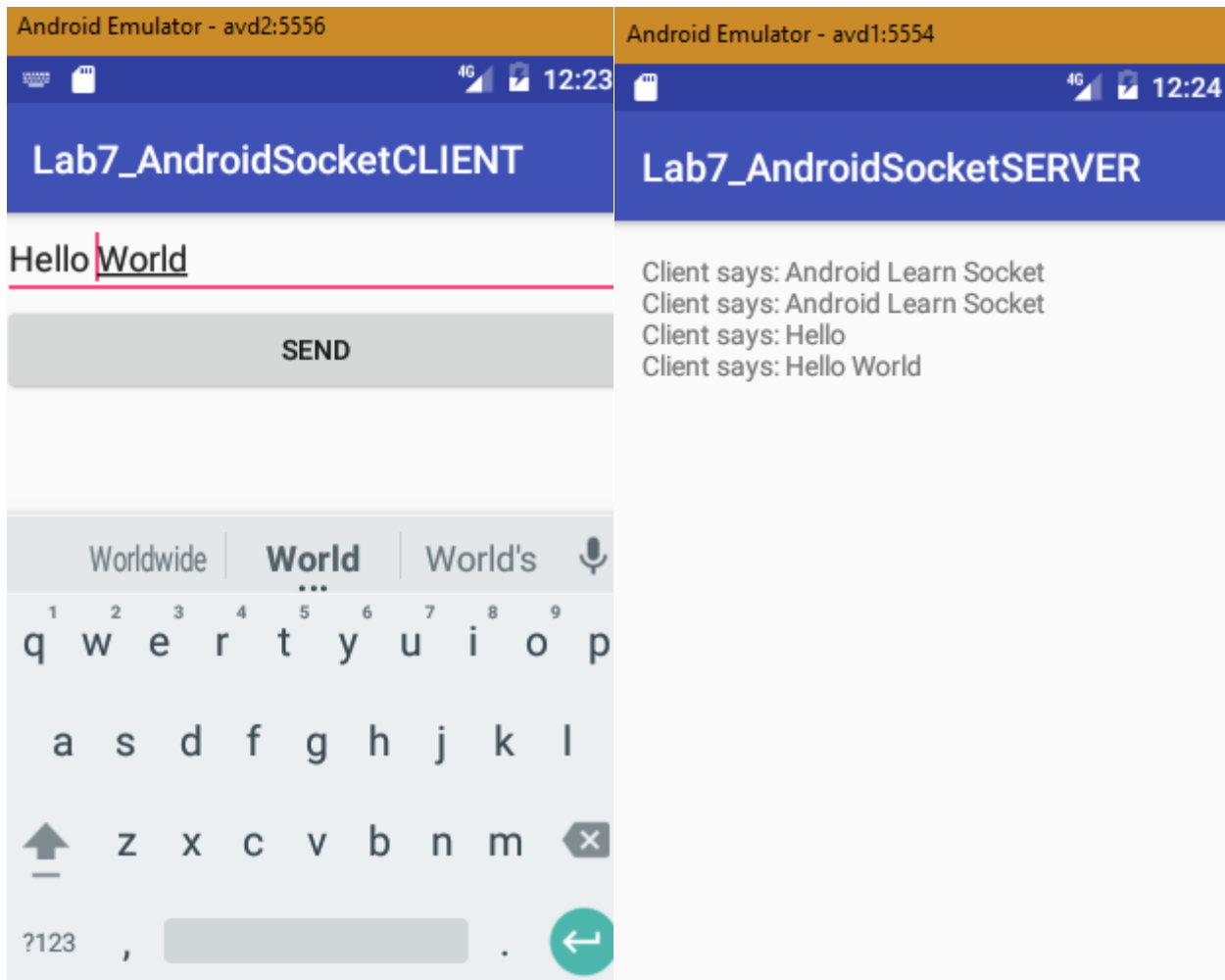
Để thực hiện chuyển cổng gõ:

```
redir add tcp:5000:6000
```

## FPT POLYTECHNIC



- ✓ Trong trường hợp nếu máy kết nối không được thì sinh viên cần vào Start -> Control Panel -> Programs -> Turn Windows features on or off. Trong Windows features. Dialog box. , select Telnet client checkbox -> click ok để cài đặt.
- ✓ Bây giờ packet sẽ đi qua đường dẫn : Emulator B -> localhost 10.0.2.2: 5000 -> Emulator A ở cổng 10.0.2.15: 6000.
- ✓ Chạy project client ở máy ảo thứ 2 và test, lưu ý là server phải chạy trước





## BÀI 2 (5 ĐIỂM)

Viết chương trình Android Client – Server sử dụng Sockets-Server Implementation. (Android) Server sẽ hiện IP của mình, ở phía (Android) Client sẽ thực hiện kết nối server thông qua địa chỉ IP server cung cấp. Khi đã kết nối thành công từ phía client thì server sẽ replay to client với “Hello from server, you are #%d”%d sẽ là một số tăng sau mỗi lần kết nối.

### HƯỚNG DẪN:

- ✓ Mở Android Studio và tạo project android mới, mở file activity\_main.xml và code giao diện như bên dưới

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.androidserversocket.MainActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:autoLink="web"
        android:text="http://android-er.blogspot.com/"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/info"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView
        android:id="@+id/infoip"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <TextView
            android:id="@+id/msg"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

    </ScrollView>
```

```
</LinearLayout>
```

- ✓ Thêm permission INTERNET cho ứng dụng ở file AndroidManifest.xml

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

- ✓ Tạo class mới đặt tên là Server.java. lớp này sẽ chứa toàn bộ các thực thi của server. Trong class vừa tạo ta sẽ tạo mới object của "ServerSocket" trong 1 hàm Thread.accept() riêng. Phương thức này sẽ trả về 1 socket object đại diện cho kết nối vừa mở. IP address and port number của client có thể nhận từ socket này.

```
private class SocketServerThread extends Thread {

    int count = 0;

    @Override
    public void run() {
        try {
            // create ServerSocket using specified port
            serverSocket = new ServerSocket(socketServerPORT);
            activity.runOnUiThread(new Runnable() {

                @Override
                public void run() {
                    activity.info.setText("I'm waiting here:
" + serverSocket.getLocalPort());
                }
            });
            while (true) {
                // block the call until connection is created
                // Socket object
                Socket socket = serverSocket.accept();
                count++;
                message += "#" + count + " from " +
socket.getInetAddress() + ":" + socket.getPort() + "\n";

                activity.runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        activity.msg.setText(message);
                    }
                });

                SocketServerReplyThread
socketServerReplyThread = new SocketServerReplyThread(socket, count);
                socketServerReplyThread.run();
            }
        } catch (IOException e) {
```

```

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

```

- ✓ Tiếp tục ta sẽ tạo một object của "SocketServerReplyThread" để mở rộng thread và ta sẽ truyền socket và đếm ở hàm tạo. nhận được OutputStream tiếp theo từ Socket sử dụng hàm getOutputStream(). 1 PrintStream bây giờ được tạo sử dụng OutputStream object như là print stream mới.

```

private class SocketServerReplyThread extends Thread {

    private Socket hostThreadSocket;
    int cnt;

    SocketServerReplyThread(Socket socket, int c) {
        hostThreadSocket = socket;
        cnt = c;
    }

    @Override
    public void run() {
        OutputStream outputStream;
        String msgReply = "Hello from Server, you are #" + cnt;

        try {
            outputStream = hostThreadSocket.getOutputStream();
            PrintStream printStream = new
PrintStream(outputStream);
            printStream.print(msgReply);
            printStream.close();

            message += "replayed: " + msgReply + "\n";

            activity.runOnUiThread(new Runnable() {

                @Override
                public void run() {
                    activity.msg.setText(message);
                }
            });

        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            message += "Something wrong! " + e.toString() +
"\n";
        }

        activity.runOnUiThread(new Runnable() {

            @Override
            public void run() {
                activity.msg.setText(message);
            }
        });
    }
}

```

```

        });
    }
}

```

✓ Cuối cùng ta cần 1 phương thức để get IP address của server chúng ta.

```

public String getIpAddress() {
    String ip = "";
    try {
        Enumeration<NetworkInterface> enumNetworkInterfaces =
NetworkInterface.getNetworkInterfaces();
        while (enumNetworkInterfaces.hasMoreElements()) {
            NetworkInterface networkInterface =
enumNetworkInterfaces.nextElement();
            Enumeration<InetAddress> enumInetAddress =
networkInterface.getInetAddresses();
            while (enumInetAddress.hasMoreElements()) {
                InetAddress inetAddress =
enumInetAddress.nextElement();

                if (inetAddress.isSiteLocalAddress()) {
                    ip += "SiteLocalAddress: " +
inetAddress.getHostAddress() + "\n";
                }

            }

        } catch (SocketException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            ip += "Something Wrong! " + e.toString() + "\n";
        }
        return ip;
    }
}

```

✓ Toàn bộ code thực thi của class Server.java như sau

```

package com.example.androidserversocket;

import java.io.IOException;
import java.io.OutputStream;
import java.io.PrintStream;
import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.SocketException;
import java.util.Enumeration;

public class Server {
    MainActivity activity;
    ServerSocket serverSocket;
    String message = "";
}

```

```

        static final int socketServerPORT = 8080;

        public Server(MainActivity activity) {
            this.activity = activity;
            Thread socketServerThread = new Thread(new
SocketServerThread());
            socketServerThread.start();
        }

        public int getPort() {
            return socketServerPORT;
        }

        public void onDestroy() {
            if (serverSocket != null) {
                try {
                    serverSocket.close();
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }

        private class SocketServerThread extends Thread {

            int count = 0;

            @Override
            public void run() {
                try {
                    // create ServerSocket using specified port
                    serverSocket = new ServerSocket(socketServerPORT);
                    activity.runOnUiThread(new Runnable() {

                        @Override
                        public void run() {
                            activity.info.setText("I'm waiting here:
" + serverSocket.getLocalPort());
                        }
                    });
                    while (true) {
                        // block the call until connection is created
                        // Socket object
                        Socket socket = serverSocket.accept();
                        count++;
                        message += "#" + count + " from " +
socket.getInetAddress() + ":" + socket.getPort() + "\n";

                        activity.runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                activity.msg.setText(message);
                            }
                        });
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        }
    }
}

```

```

socketServerReplyThread = new SocketServerReplyThread(socket, count);
socketServerReplyThread.run();

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

private class SocketServerReplyThread extends Thread {

    private Socket hostThreadSocket;
    int cnt;

    SocketServerReplyThread(Socket socket, int c) {
        hostThreadSocket = socket;
        cnt = c;
    }

    @Override
    public void run() {
        OutputStream outputStream;
        String msgReply = "Hello from Server, you are #" + cnt;

        try {
            outputStream = hostThreadSocket.getOutputStream();
            PrintStream printStream = new
PrintStream(outputStream);
            printStream.print(msgReply);
            printStream.close();

            message += "replayed: " + msgReply + "\n";

            activity.runOnUiThread(new Runnable() {

                @Override
                public void run() {
                    activity.msg.setText(message);
                }
            });

        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            message += "Something wrong! " + e.toString() +
"\n";
        }

        activity.runOnUiThread(new Runnable() {

            @Override
            public void run() {
                activity.msg.setText(message);
            }
        });
    }
}

```

```

    }

    public String getIpAddress() {
        String ip = "";
        try {
            Enumeration<NetworkInterface> enumNetworkInterfaces =
NetworkInterface.getNetworkInterfaces();
            while (enumNetworkInterfaces.hasMoreElements()) {
                NetworkInterface networkInterface =
enumNetworkInterfaces.nextElement();
                Enumeration<InetAddress> enumInetAddress =
networkInterface.getInetAddresses();
                while (enumInetAddress.hasMoreElements()) {
                    InetAddress inetAddress =
enumInetAddress.nextElement();

                    if (inetAddress.isSiteLocalAddress()) {
                        ip += "SiteLocalAddress: " +
inetAddress.getHostAddress() + "\n";
                    }

                }
            }

        } catch (SocketException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            ip += "Something Wrong! " + e.toString() + "\n";
        }
        return ip;
    }
}

```

- ✓ Mở file MainActivity.java và tạo đối tượng của class Server vừa tạo sau đó thực thi code .

```

package com.example.androidserversocket;

import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;

public class MainActivity extends Activity {
    Server server;
    TextView info,infoip, msg;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        info = (TextView) findViewById(R.id.info);
    }
}

```

```

        infoip = (TextView) findViewById(R.id.infoip);
        msg = (TextView) findViewById(R.id.msg);
        server = new Server(this);
        infoip.setText(server.getIpAddress() + ":" + server.getPort());

    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        server.onDestroy();
    }

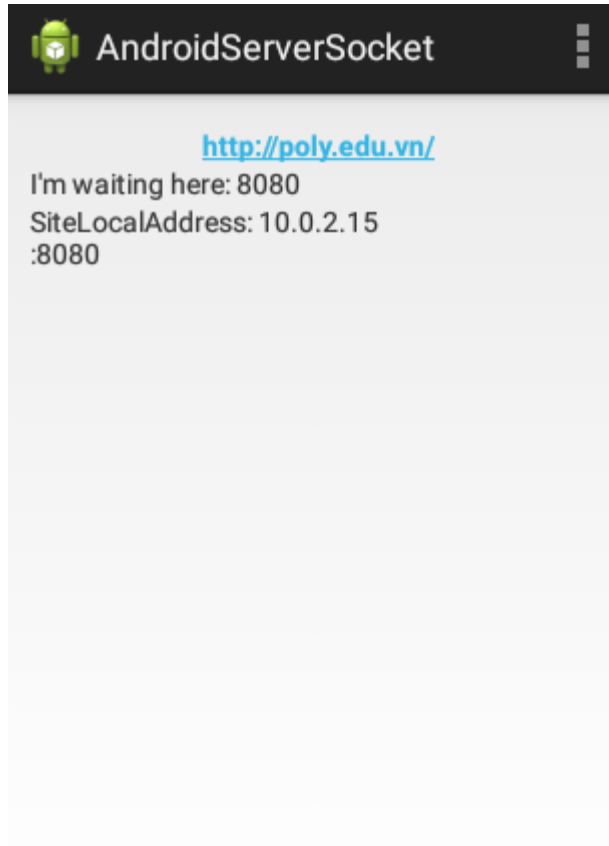
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}

```

✓ Tiến hành chạy project android server trên máy thật





- ✓ Tiếp theo ta sẽ viết code cho phía Client. Project android mới và code file `activity_main.xml` layout như sau.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.androidsocketclient.MainActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginBottom="20dp"
        android:layout_marginTop="20dp"
        android:autoLink="web"
        android:text="http://poly.edu.vn/"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/addressEditText"
```

```

        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Server ip address" />

        <EditText
            android:id="@+id/portEditText"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Server port number" />

        <Button
            android:id="@+id/connectButton"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dp"
            android:text="Connect..." />

        <Button
            android:id="@+id/clearButton"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Clear" />

        <TextView
            android:id="@+id/responseTextView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

    </LinearLayout>

```

✓ Thêm permission INTERNET trong file AndroidManifest.xml

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

- ✓ Tạo class mới đặt tên là Client.java. Lớp này sẽ chứa toàn bộ code thực thi của Client và sẽ kế thừa AsyncTask để thực hiện multitasking dễ dàng hơn. Ta tạo 1 socket sử dụng IP và port detail được cung cấp bởi user. Tiếp theo getInputStream() được gọi trong socket đã tạo để get InputStream và read data từ socket này. Kết nối bị block cho đến khi không có data nào được đọc. sử dụng try catch để handle exceptions.

```

package com.example.androidsocketclient;

import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.Socket;
import java.net.UnknownHostException;

import android.os.AsyncTask;
import android.widget.TextView;

public class Client extends AsyncTask<Void, Void, Void>{

```

```

        String dstAddress;
        int dstPort;
        String response = "";
        TextView textResponse;

        Client(String addr, int port, TextView textResponse) {
            dstAddress = addr;
            dstPort = port;
            this.textResponse = textResponse;
        }

        @Override
        protected Void doInBackground(Void... params) {
            // TODO Auto-generated method stub
            Socket socket = null;

            try {
                socket = new Socket(dstAddress, dstPort);

                ByteArrayOutputStream byteArrayOutputStream = new
                ByteArrayOutputStream(
                    1024);
                byte[] buffer = new byte[1024];

                int bytesRead;
                InputStream inputStream = socket.getInputStream();

                /*
                 * notice: inputStream.read() will block if no data return
                 */
                while ((bytesRead = inputStream.read(buffer)) != -1) {
                    byteArrayOutputStream.write(buffer, 0, bytesRead);
                    response += byteArrayOutputStream.toString("UTF-8");
                }

            } catch (UnknownHostException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                response = "UnknownHostException: " + e.toString();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                response = "IOException: " + e.toString();
            } finally {
                if (socket != null) {
                    try {
                        socket.close();
                    } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }
            }
            return null;
        }

        @Override
        protected void onPostExecute(Void result) {
            textResponse.setText(response);
        }
    }

```

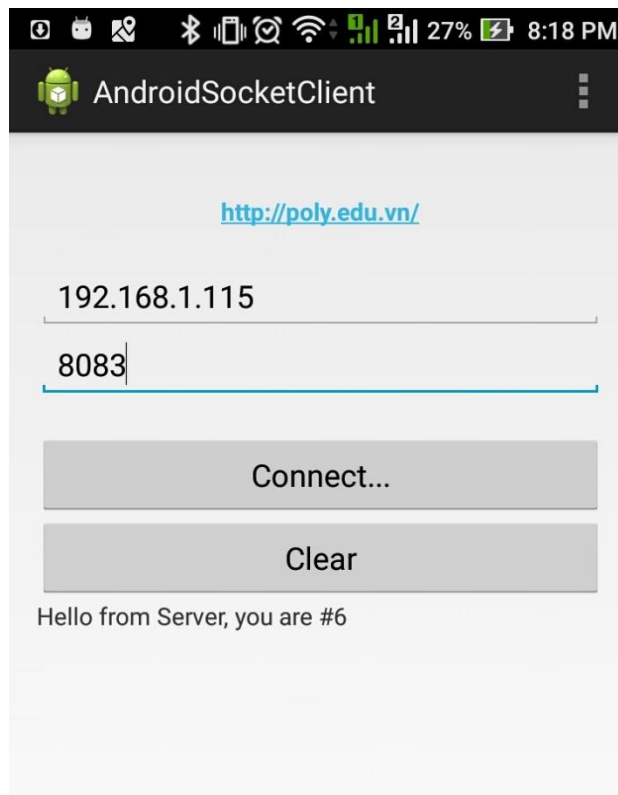
```

        super.onPostExecute(result);
    }
}

```

- ✓ Mở file MainActivity.java, tạo object của Client.java truyền IP và port của server và trả kết quả lên textview. Code sẽ như sau.

- ✓ Tiến hành chạy project android client trên thật thứ 2.



- ✓ Test kết quả như sau. 2 máy điện thoại phải có cùng wifi, và client sẽ nhập IP và port giống với server cung cấp

