**Project Report**

**Title:** **Bluetooth Register**

**College: School of International Education**

**Name:**

**Student ID:**

**Collaborator (Name and Student ID):谈世翔 202130430232、 王侯 202130430249、陆俊安202130430195、叶昊林202130430300**

**Teacher: Zhang Jing**

**Date:**

**Bluetooth Register**

1. Project Background

In SCUT, it is an everyday routine that checking whether the students attend classes timely and fully, for fear that some students will be absent from class deliberately. We want to find some program to make this routine more convenient. However, the current ways, such as the sign-in of some programs or uploading the cell phone location in WeChat, cannot guarantee the student wouldn’t leave class halfway and the students even could sign in at the dormitory, limited by the accuracy of positioning. So, to avoid this problem, we tried to develop an attendance checking software with a new checking mechanism, that is to check the attendance through Bluetooth communication.

In this software, teaching assistant can initiate check-ins and students can participate in check-ins. It can avoid the inconvenience of verbal check-in and solve the problems like inaccurate positioning.

2. Project Design Concept

（Describe the design concept and scheme options, clarify why you choose this design scheme, and list its characteristics.

The contents focus on the function design and corresponding requirements.）

3. Process discussion

（Focus on how the design is implemented, including a detailed description of the design process. Please express in a clear-cut and precise way.

Requirements: Each figure must include two descriptions. One above the figure, to explain the main purpose and functions. The second below the figure, to describe functions of the components, and interaction between the components, or the flow chart within the figure.）

1.UI Design

1.1 UI for MainActivity(activity\_start\_page.xml)

Design Window for StartPage UI

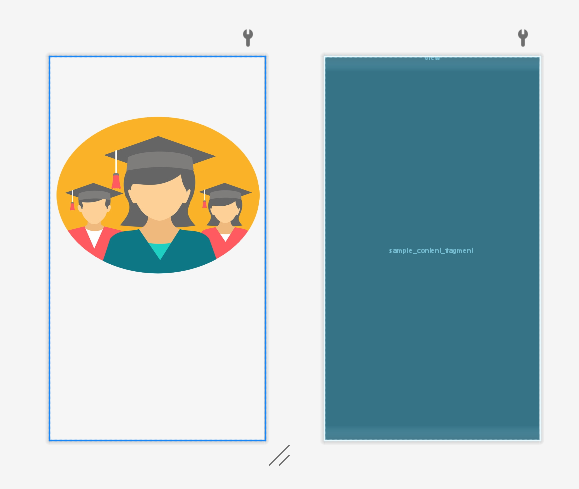


The home page consists of four parts, which are application information, entry button, identity options and team information. The function of the start page is to distinguish the identity of the check-in administrator and the participating students, and to guide them to the corresponding Activity page for check-in operation, so the function of the page is highlighted with larger buttons, and a larger position is reserved to make the software information more conspicuous.

Codes for StartPage UI

Main content are CheckBox and Button, others are textView.

1.2 UI for StudentActivity(student.xml)

Design Window for Student UI

The outlayer component is LinearLayout, inside layer is FrameLayout, a picture representing students is setted to be the backgroud.

Codes for Student UI

Define a LinearLayou whose id is "student", containing a FrameLayout whose id is sample\_content\_fragment, set the background as “@drawable/student\_icon”。

1.3Manager UI

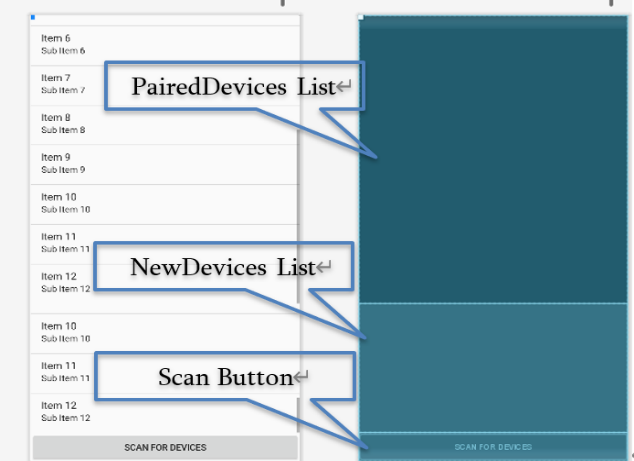
the design of the page of manager activity:

The four name of the team members are added by a function at the very beginning, and we add them to the list of students and display them on the page.

the UI code is only about a recycler view:

we set the four edges of the recycler view and the four edges of its parent to be overlapped.

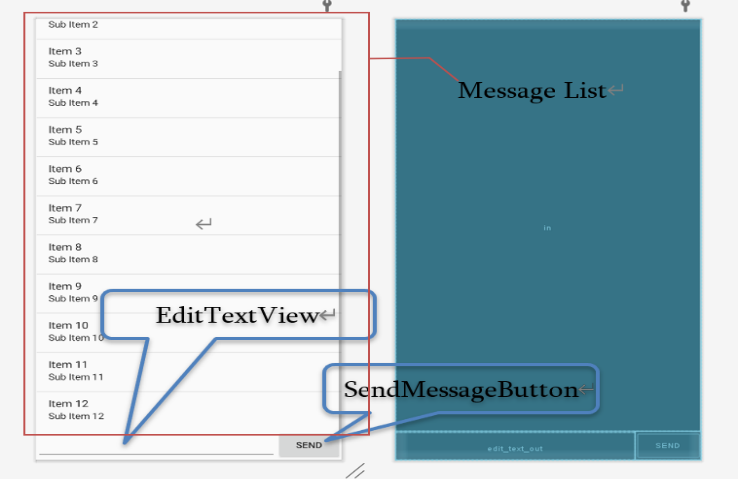
1.4 Other UI components

 （1）UI for Device List（activity\_device\_list.xml）

The Device List has two kinds, which is "PairedDevicesList" and "NewDevicesList". For,t he devices which have built bluetooth connections, their names will be added to PairerDevicesList, otherwhise, to NewDevicesList. To scan and find devices, tap the button "SCAN FOR DEVICES", the scanned devices name then will be stored into according list.

Codes for DeviceList UI

The paired devices and newly-found devices are respectively defined as paired\_devices and new\_devices in type ListView. Scan Button is defined as "button\_scan".

(2)UI Components for Bluetooth Chat（fragment\_bluetooth\_chat.xml）

There are three parts in Bluetooth Chat Component, which is MessageList, EditTextView, SendMessageButton. Editing the text to be sent by bluetooth in EditTextView, tap SendMessageButton, the words will perfectly reveal at MessageList.

2、Activity Design

2.1 MainActivity

2.2 StudentActivity

2.3ManagerActivity

3、Bluetooth Communication Class Design

3.1 蓝牙通讯碎片（BluetoothChatFragment.java）

1. Overview

The BluetoothChatFragment class is a very important class for its usages of achieving the connecting between devices and its support to sending messages from one device to another. Often this class will call another class called DeviceListActivity, we will mention it soon.

The UI we may use in this class are made of two parts.

First let's see the UI of the menu:



The two items are the choices from the menu, one with the text "Connect a Device to Register" and another one with the text "Make discoverable".

The next is the UI from the Bluetooth chat fragment:

fragment\_bluetooth\_chat:



Class Definition:

public class BluetoothChatFragment extends Fragment

Member variables:

private static final String *TAG* = "BluetoothChatFragment";

private static final int *REQUEST\_CONNECT\_DEVICE\_SECURE* = 1;

private static final int *REQUEST\_CONNECT\_DEVICE\_INSECURE* = 2;

private static final int *REQUEST\_ENABLE\_BT* = 3;

private ListView mConversationView;

private EditText mOutEditText;

private Button mSendButton;

private String mConnectedDeviceName = null;

private ArrayAdapter<String> mConversationArrayAdapter;

private StringBuffer mOutStringBuffer;

private BluetoothAdapter mBluetoothAdapter = null;

private BluetoothChatService mChatService = null;

Member functions:

private void setupChat() ;

private void ensureDiscoverable() ;

private void sendMessage(String message) ;

private TextView.OnEditorActionListener mWriteListener;

private void setStatus(int resId);

private void setStatus(CharSequence subTitle);

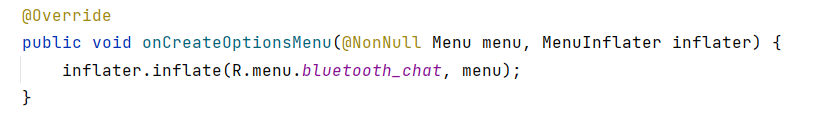
private final Handler mHandler;

public void onActivityResult(int requestCode, int resultCode, Intent data);

private void connectDevice(Intent data, boolean secure);

1. Detailed Analysis

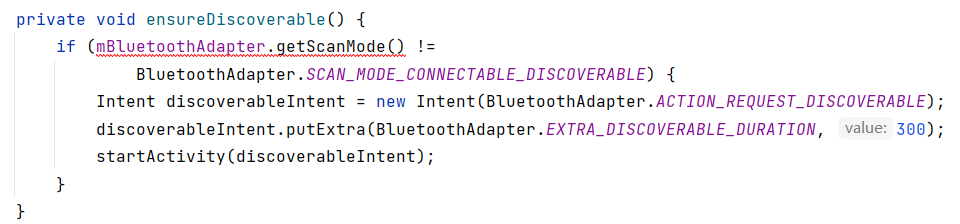
<1> The menu



The code is used to import the menu UI to the activity, and the Onclick functions are as followed:



when clicking the first item in the menu, the activity will directly call the DeviceListActivity to "borrow" its usage of connecting devices, and when we click the second item, the system will ask if you want your device to be discovered by others.



and it is realized by function "discoverable.putExtra", and the "value" filled in the brackets indicates the lasting time of the situation.

<2> The bluetooth chatting



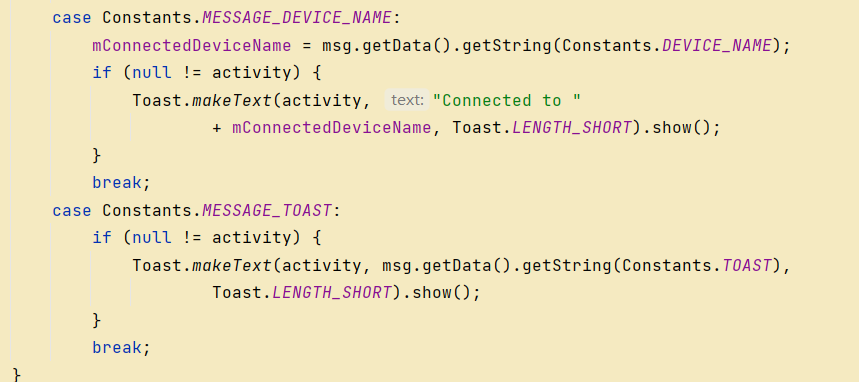
The setupChat function is the key to the bluetooth chatting between two devices. First we create a new adapter to store the messages so that we can see the chatting history on the screen, then we set this adapter to the mConversatiionView we already create before. The next is the sending button, we add an onclicklistener to it to turn the message in the textview into string variables and then send it out. Finally we initialize the bluetoothchatservice object and the Buffer.



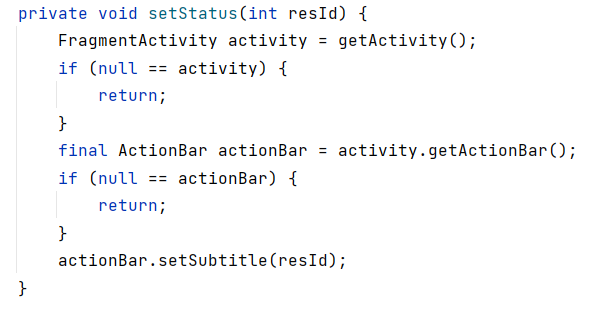
This is the sendMessage function, first we chaeck whether the device is in the connected situation, then we check if there is anything to send by checking the length of message, if the answer is yes, we order the mChatService to write in the byte and send it. Finally we originate the outbuffer to initialize it.

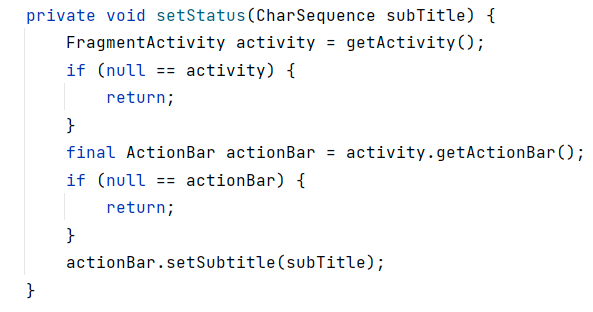
<3> The handler





The handler is used to get the information back from the Bluetoothservice. We apply "switch" to determine the state of the msg.what, if the result is 1(Constant.Message\_State\_Changed), we need to judge the value of msg.arg1, according to the result, we need to apply different actions to update the status on the action bar. If the result is MESSAGE\_READ or MESSAGE\_WRITE, we need to construct a string from the buffer, and if the result is MESSAGE\_DEVICE\_NAME or MESSAGE\_TOAST, we need to send a toast of different content.





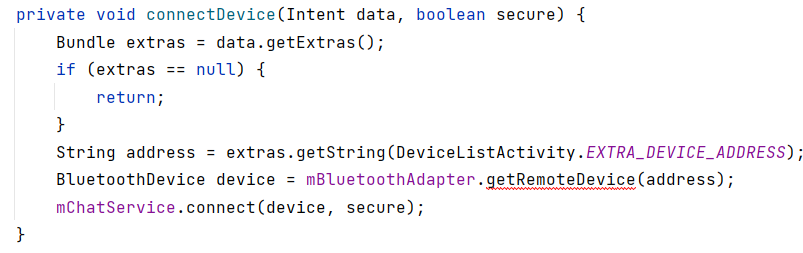
The setStatus function has two versions, they take in different arguments, but they are all used to update the status on the action bar

<4> The onActivityResult function



the function onActivityResult is used to return the data to the last activity. It takes in a unique requestcode from the function startActivityForResult, and the resultcode is return from the subactivity. When the subactivity returns a device to connect, we use function connect to connect the device; when it return a request to enable Bluetooth, if the Bluetooth is already enabled, we directly set up the chat; if it's not enabled, we send out a toast to tell the user that the Bluetooth is not enabled.

<5> The connectDevice function



Just like its name, this function is used to connect the device, it uses the getString function to get the bluetooth address of the device in the DeviceListActivity, finally it uses the mChatService to realize the connection.

3.2 Bluetooth Service（BluetoothChatService.java）

1. Overview

This class does all the work for setting up and managing Bluetooth connections with other devices. It has a thread that listens for incoming connections, a thread for connecting with a device, and a thread for performing data transmissions when connected.

On the server side, use a BluetoothServerSocket to create a listening server socket. When a connection is accepted by the BluetoothServerSocket, it will return a new BluetoothSocket to manage the connection. On the client side, use a single BluetoothSocket to both initiate an outgoing connection and to manage the connection.

Class Definition：public class BluetoothChatService

Member variables

   private static final String NAME = "BluetoothChatInsecure";

    private static final UUID MY\_UUID =

            UUID.fromString("8ce255c0-200a-11e0-ac64-0800200c9a66");

    private final BluetoothAdapter mAdapter;

    private final Handler mHandler;

    private AcceptThread mAcceptThread;

    private ConnectThread mConnectThread;

    private ConnectedThread mConnectedThread;

    private int mState;

    private int mNewState;

    public static final int STATE\_NONE = 0;

    public static final int STATE\_LISTEN = 1;

    public static final int STATE\_CONNECTING = 2;

    public static final int STATE\_CONNECTED = 3;

Member Functions：

public BluetoothChatService(Context context, Handler handler)

private synchronized void updateUserInterfaceTitle()

 public synchronized int getState()

public synchronized void start()

public synchronized void connect(BluetoothDevice device, boolean secure)

public synchronized void connected(BluetoothSocket socket, BluetoothDevice device, final String socketType)

public synchronized void stop()

public void write(byte[] out)

private void connectionFailed()

private void connectionLost()

Private Classes

 private class AcceptThread extends Thread

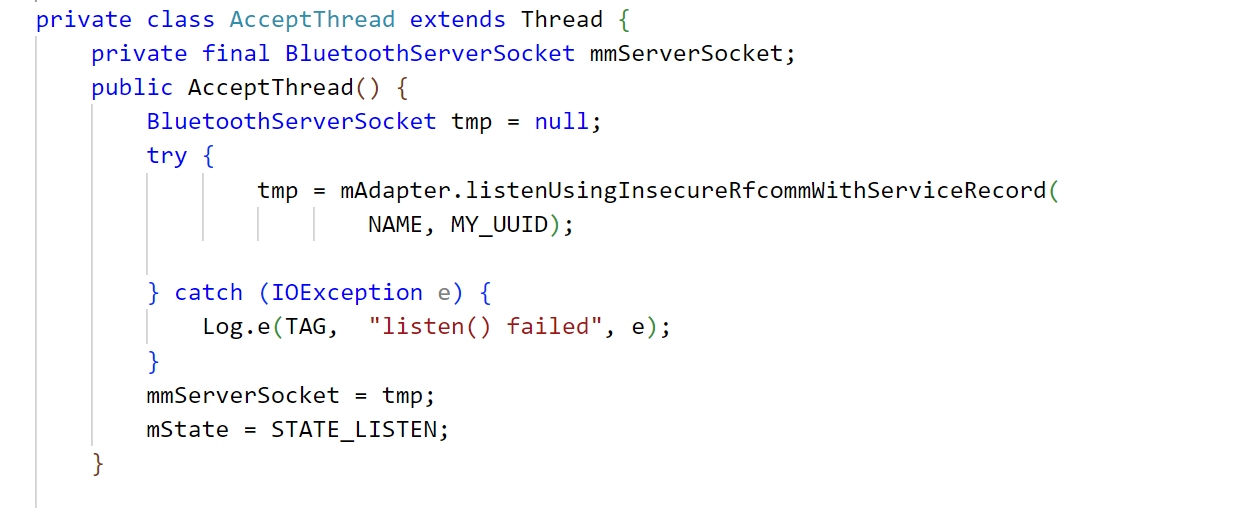
private class ConnectThread extends Thread

private class ConnectedThread extends Thread

(2) Detail Analysis

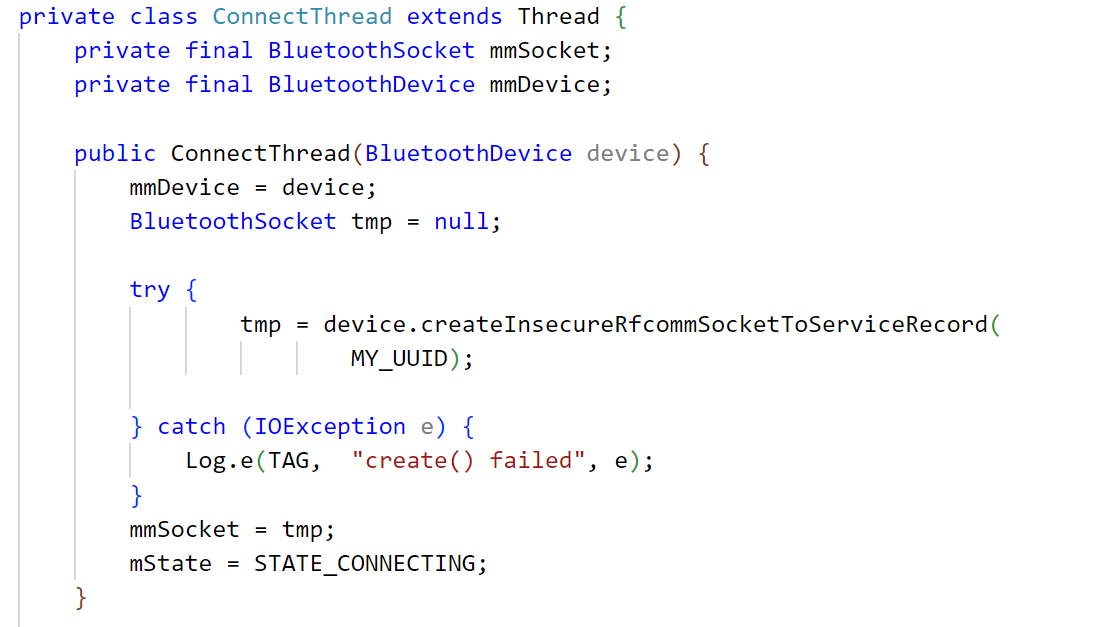
<1> In class AcceptThread

This thread runs while listening for incoming connections. It behaves like a server-side client. It runs until a connection is accepted or until cancelled.



The most common type of Bluetooth socket is RFCOMM, which is the type supported by the Android APIs.To listen to a BluetoothSocket, use listenUsingInsecureRfcommWithServiceRecord() .

<2> In class ConnectThread extends Thread

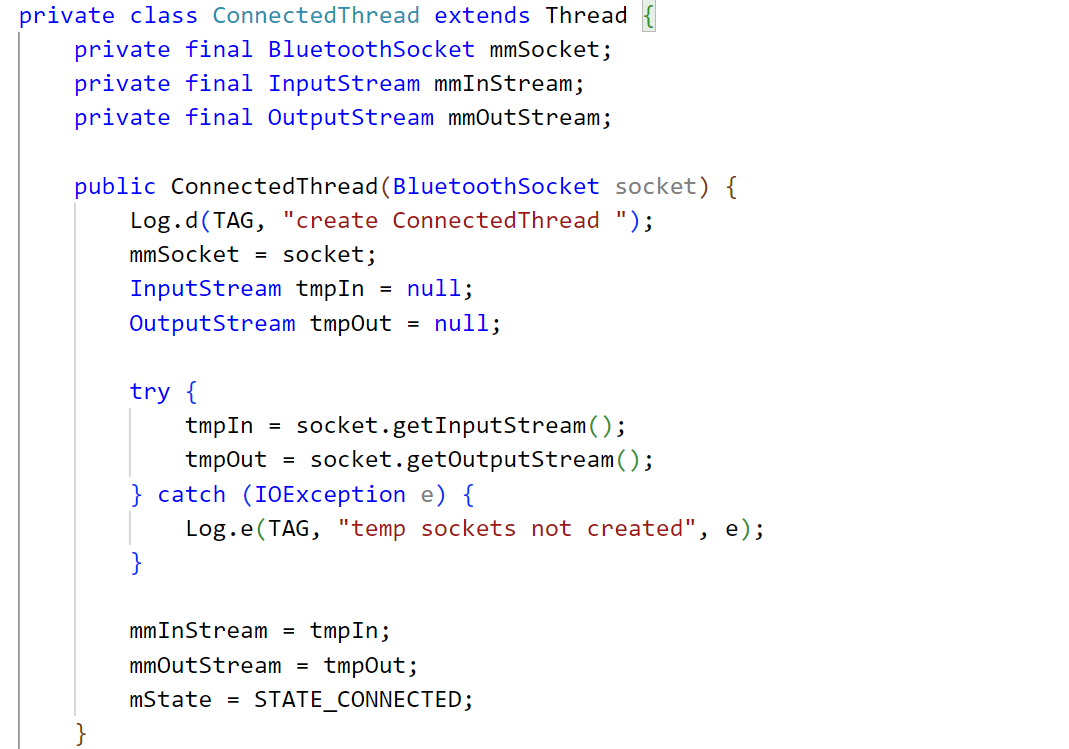
This thread runs while attempting to make an outgoing connection with a device. It runs straight through; the connection either succeeds or fails.

Get a BluetoothSocket for a connection with the given BluetoothDevice.Set the state to STATE\_CONNECTING.

<3>In class ConnectedThread extends Thread

This thread runs during a connection with a remote device.

It handles all incoming and outgoing transmissions.



After connected, users can send messages. mmInstream and mmOutstream can obtain the input and output information.

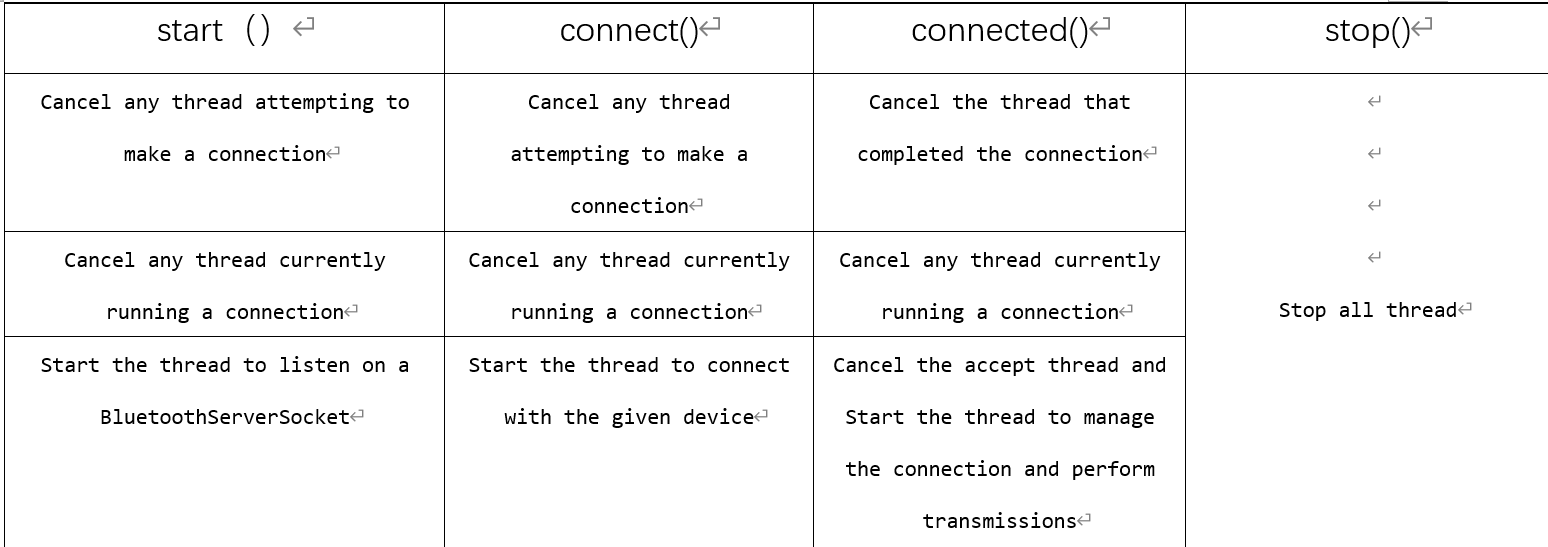
Begin mConnectedThread



byte[] buffer is used to store the buffer reading from the input stream. Start a handler send the massage to the UI activity.

<4> Life cycle functions

Four functions are prepared for control the life of the chat service, which are start(), connect() connected(), stop(). Their usages are shown in the following table.



3.3 Device List（DeviceListActivity.java）

1. Overview

Reveal the new devices and paired devices scanned by bluetooth.

Class Definition：public class DeviceListActivity extends Activity

Member Variables：

public static String EXTRA\_DEVICE\_ADDRESS = "device\_address"

private BluetoothAdapter mBtAdapter

private ArrayAdapter<String> mNewDevicesArrayAdapter

Member Functions：

@Override

    protected voidonCreate(BundlesavedInstanceState)

 @Override

    protected void onDestroy()；

 private void doDiscovery() ；

Private Objectss：

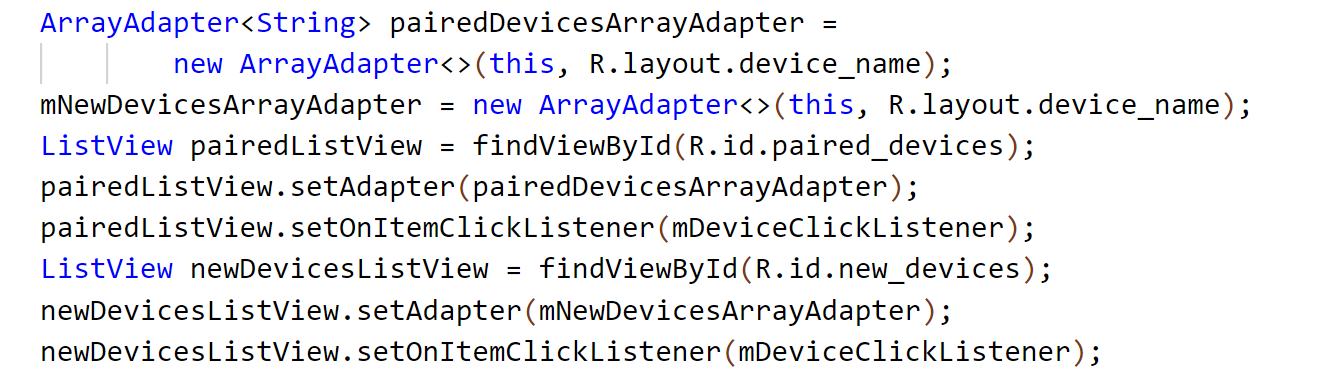
private AdapterView.OnItemClickListener mDeviceClickListener

private final BroadcastReceiver mReceiver

(2) Detail Analysis

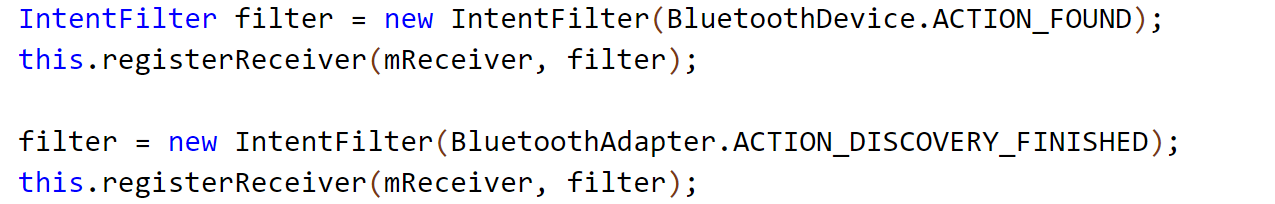
Class DeviceListActivity is used for obtaining device information. Member variable BluetoothAdapter mBtAdapter is the basic object in Android bluetooth，having the ability running bluetooth-concerned functions，and ArrayAdapter<String> mNewDevicesArrayAdapter can save the name of new devices, then sending the data to UI interface.

<1> In protected void onCreate(Bundle savedInstanceState)

Define pairedDevicesArrayAdapter and its UI components

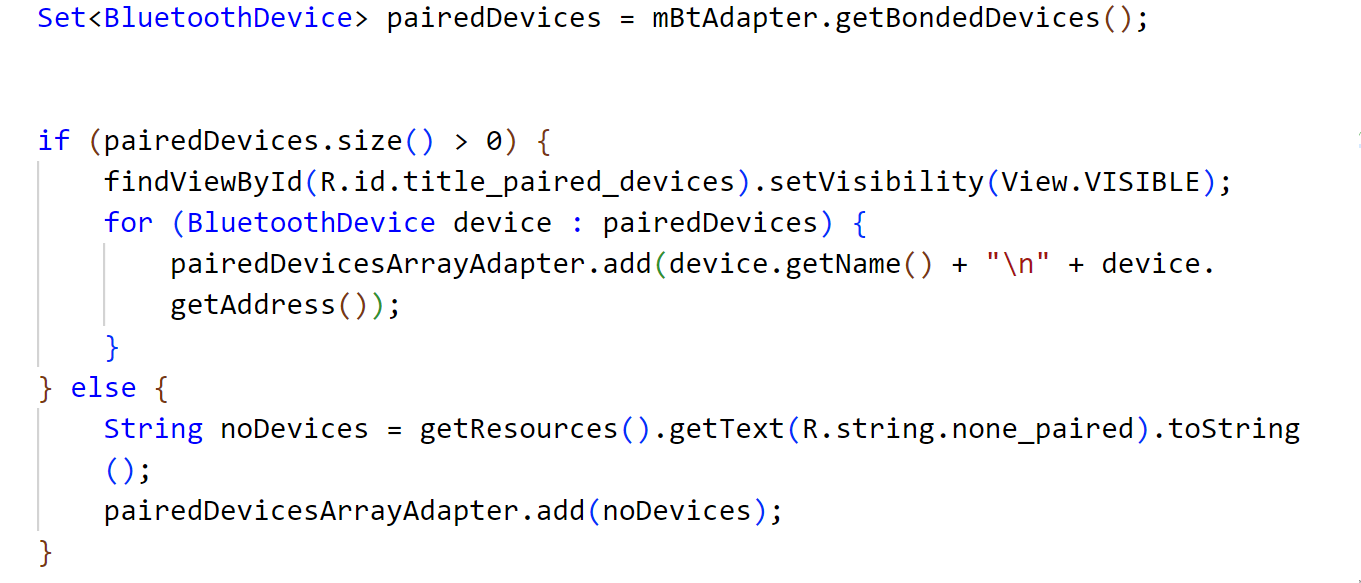
Initialize pairedDevicesArrayAdapter and mNewDevicesArrayAdapter, choose ListView as the UI component, and initialize PairedListView, newDevicesListView. Set the listener for clicking.

Define the Receiver of Broadcast



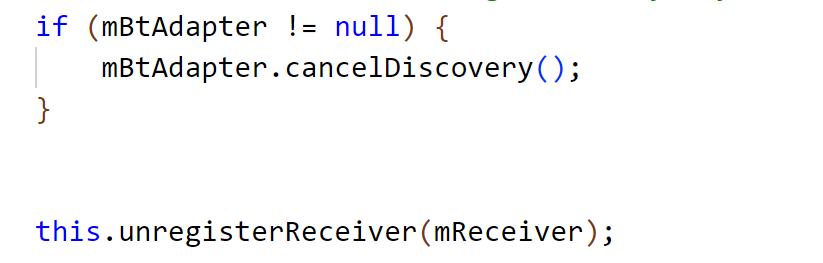
BluetoothDevice.ACTION\_FOUND is the constant value when a device is found，BluetoorhAdapter.ACTION\_DISCOVERY\_FINISHED is the constant value when the discovery is finished.

Get the current pairedDeviceList, add them to the Adapter



If pairedDevices.size() is greater than zero, that means there exists paired devices. Then add the device names and MAC address to Adapter, otherwise, reveal "no paired devices".

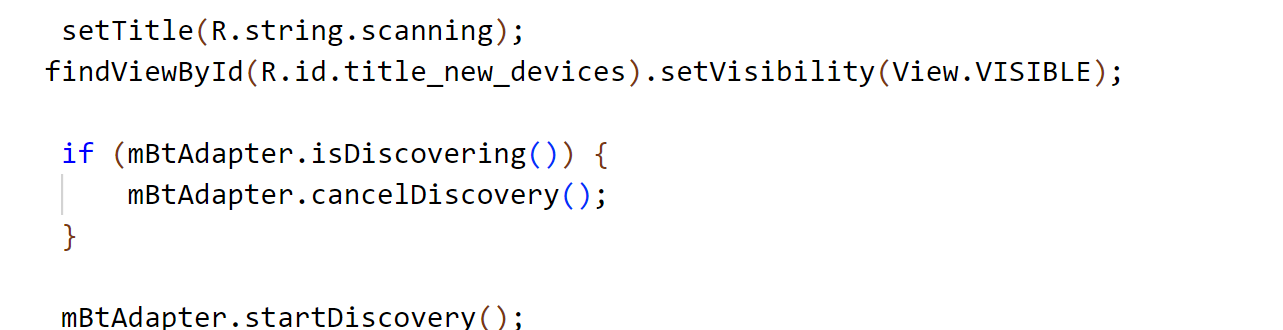
<2>In protected void onDestroy()



When the device list is closed, cancel the discovery and unregister the receiver.

<3> In private void doDiscovery()

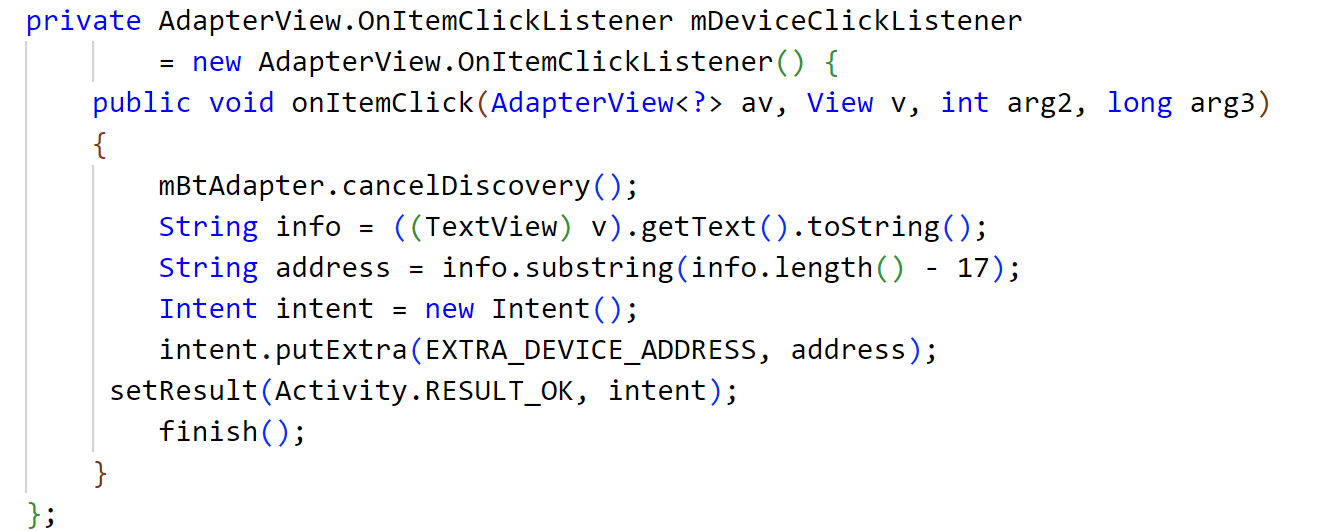
Searching function, and reveal the result in UI interface



When the discovery is started, funtion setTitle() can reveal "scanning" to inform users. If new devices are found（R.id.title\_new\_devices）, their names will be put into device list and set visible.

<4>In private AdapterView.OnItemClickListener mDeviceClickListener

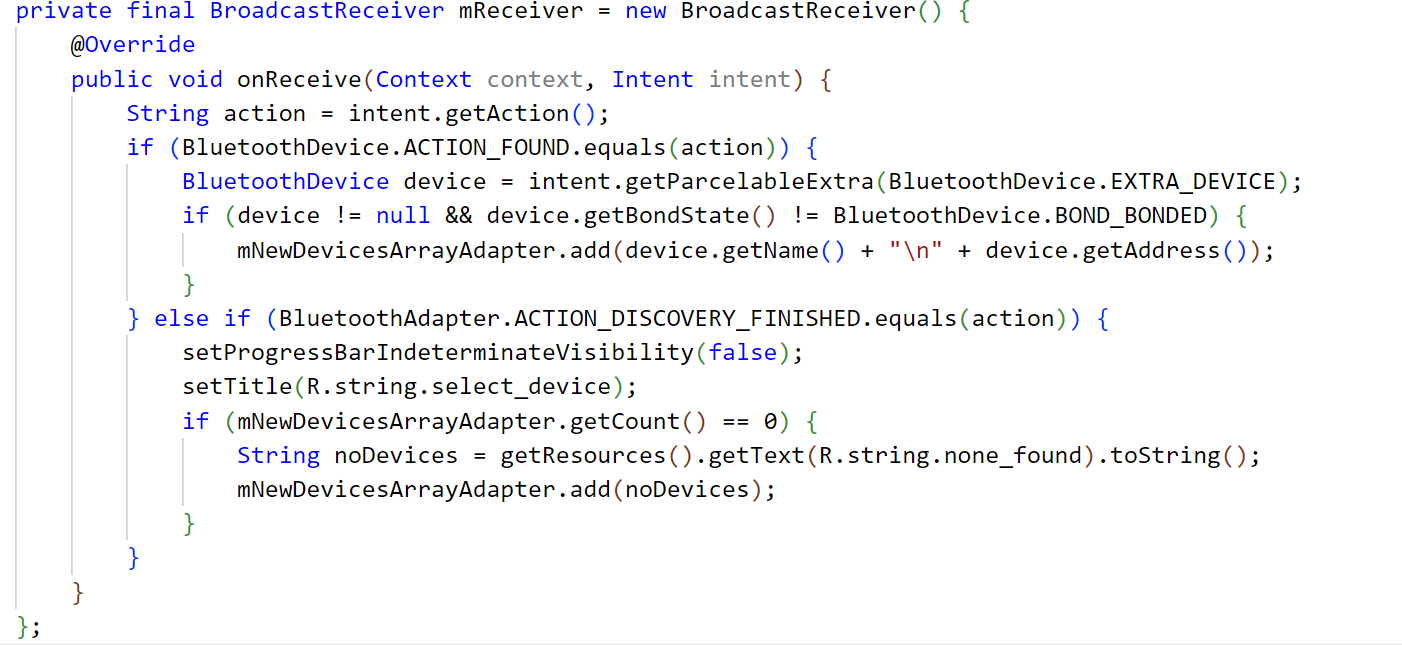
Set ONItemClickListener()



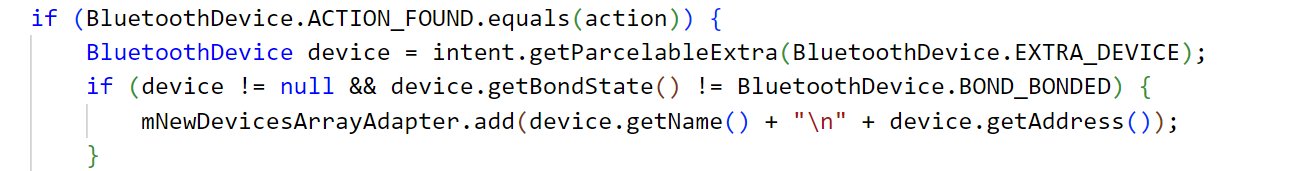
When the item in device list is clicked, cancelDiscover() will be called to stop discovering devices. String address saves the MAC address, which is the last 17 numbers of String info, that's why address =info.substring(info.length()-17). Finally, send intent containg MAC address to setResult().

<5>In private final BroadcastReceiver mReceiver

mReceiver listens to the states and update itself when states changes

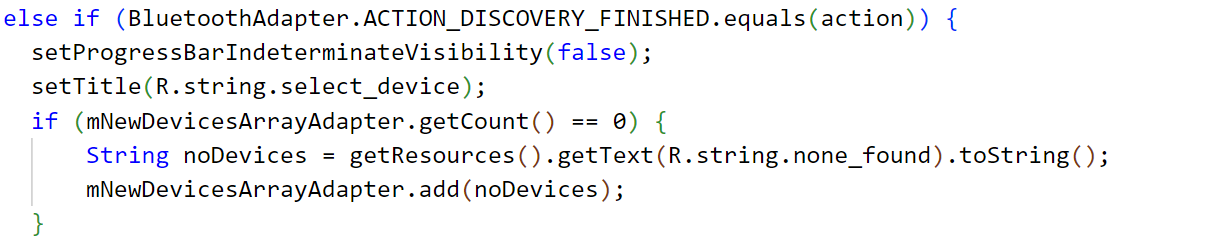


Override onReceive()，call intent.getAction() to listen to the broadcast.



intent.getParcelableExtra() can put the object into Parcel Object, and restore it if necessary, thus implimenting passing objects between two activities. device.getBondedState() returns the state of the bond, when it is bonding, returns BOND\_BONDGIND, when successfully bonded, reutnr BOND\_BONDED, no device to be bonded, return BOND\_NONE. When the devices has never been bonded, which means getBondState()!=BOND\_BONDED, add that device to mNewDevicesArrayAdapter.

Finish the Discovery



There are five constant values in Android bluetooth actions，they are ACTION\_DISCOVERY\_STARTED，ACTION\_DISCOVERY\_FINISHED，ACTION\_LOCAL\_NAME\_CHANGED，ACTION\_SCAN\_MODE\_CHANGED，ACTION\_STATE\_CHANGED.When BluetoothAdapter.ACTION\_DISCOVERY\_FINISHED.equals(action) is true，bluetooth adapter finished its searching,and entitle the found names.If there is no objects in the adapter，entitle “NO Device”.

**4. Result analysis**

4.1 Work Flow on real phone

4.2 Conclusions & Inferences

(1) The disastrous failure of team leader

**5. Task Allocation Among Team Members**

|  |  |  |
| --- | --- | --- |
| Name | ID | Distribution |
| 谈世翔 | 202130430232 | Student Screen |
| 王侯 | 202130430249 | Project Designer、Bluetooth Part |
| 陆俊安 | 202130430195 | Register Screen |
| 叶昊林 | 202130430300 | Monitor Screen |