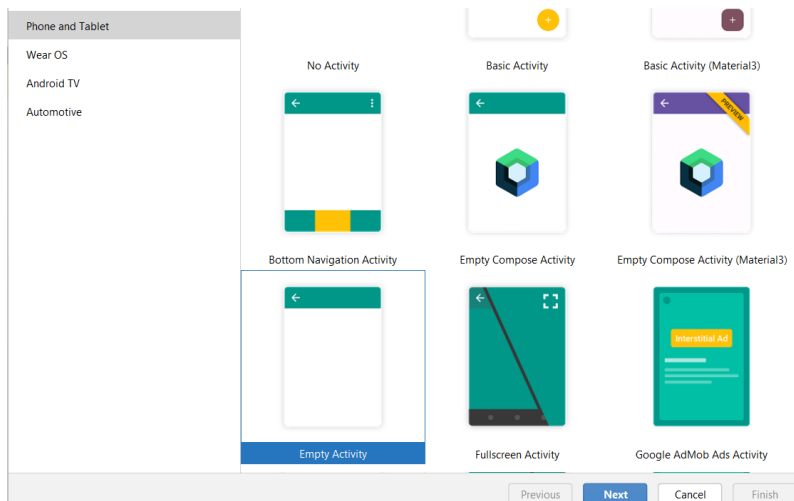


## Thiết kế ứng dụng có khả scan, Pair/unpair thiết bị Bluetooth khả dụng và truyền nhận dữ liệu qua BLE



Đặt tên project là **BLUETOOTHControl** và lưu tại folder **BLUETOOTHControl**

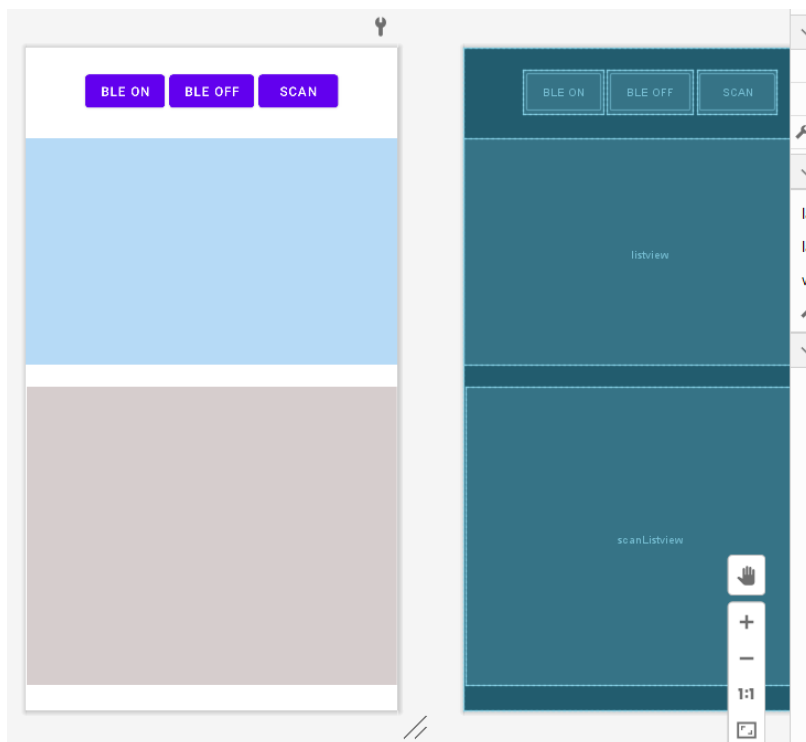
Mở *AndroidManifest.xml* và điền 4 dòng lệnh cấp quyền bluetooth

```
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
<uses-permission android:name="android.permission.BLUETOOTH_SCAN" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.BLUETOOTH_BOND" />
```

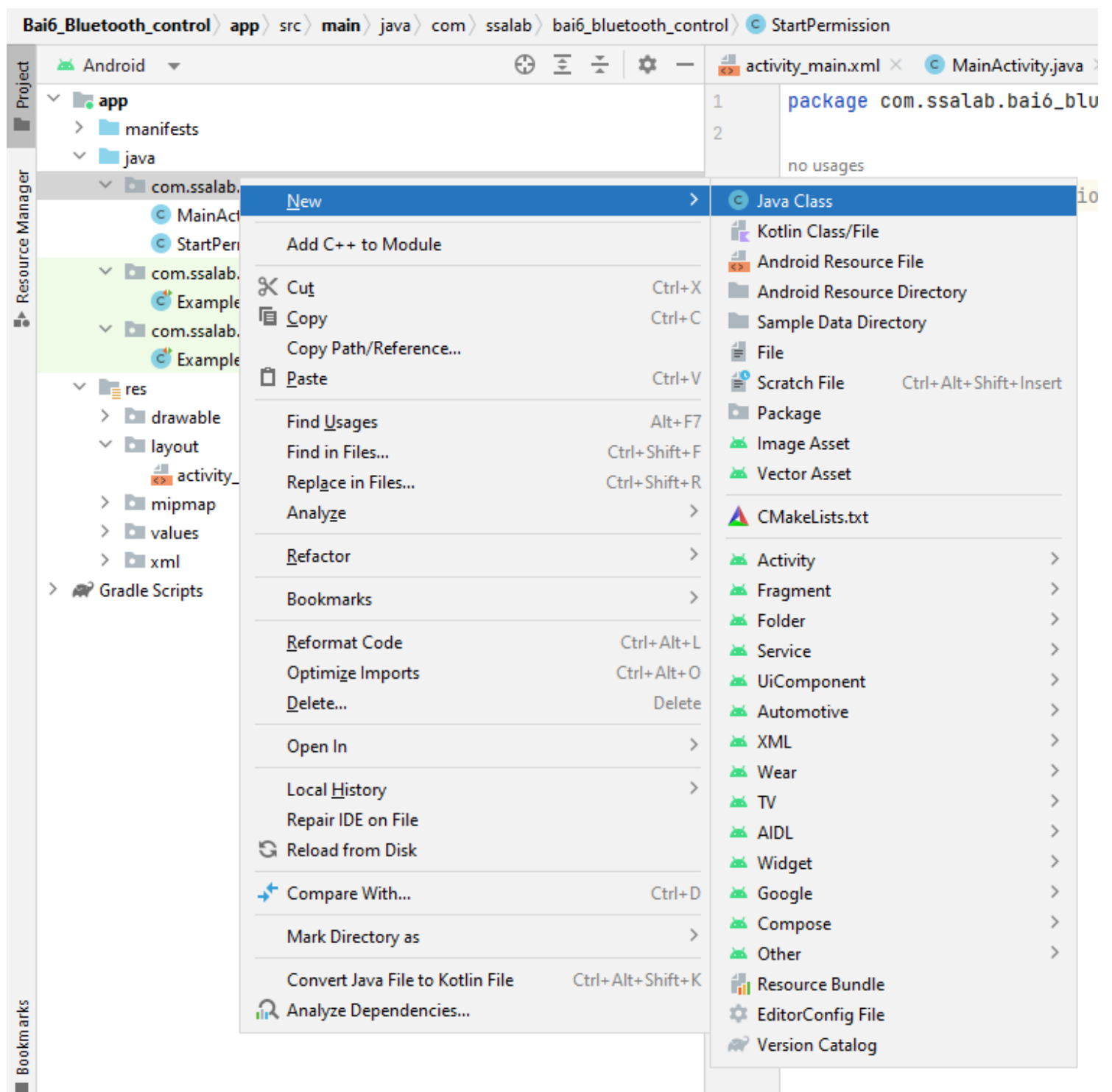
Mở *activity\_main.xml* để thiết kế giao diện với 3 button và 2 listview

Sử dụng **constraintlayout** hoặc bất kỳ layout nào để thiết kế.

Gọi 3 button đặt id lần lượt là **btON**, **btOFF**, **btScan** (tương ứng text là **BLE ON**, **BLE OFF**, **SCAN**) như hình. 2 listview có ID lần lượt là **listview** và **scanListView**



Tạo 1 class java mới đặt tên là **StartPermission.java**



Tiếp tục lập trình yêu cầu người dùng kích hoạt quyền truy cập các phân quyền bluetooth trong class **StartPermission.java**

```
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

no usages
public class StartPermission {
    no usages
    public static boolean requestBluetoothPermissions(Activity activity)
    {
        boolean needPermission = false;
        if (Build.VERSION.SDK_INT >= 31) {

            if (ContextCompat.checkSelfPermission(activity, android.Manifest.permission.BLUETOOTH_CONNECT) == PackageManager.PERMISSION_DENIED) {
                needPermission = true;
            }

            if (ContextCompat.checkSelfPermission(activity, android.Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_DENIED) {
                needPermission = true;
            }

            if (ContextCompat.checkSelfPermission(activity, android.Manifest.permission.BLUETOOTH_SCAN) == PackageManager.PERMISSION_DENIED) {
                needPermission = true;
            }

            if (needPermission) {
                ActivityCompat.requestPermissions(activity, new String[]{
                    android.Manifest.permission.BLUETOOTH_CONNECT,
                    android.Manifest.permission.ACCESS_FINE_LOCATION,
                    android.Manifest.permission.BLUETOOTH_SCAN
                }, requestCode: 100);
            }
        }
        return needPermission;
    }
}
```

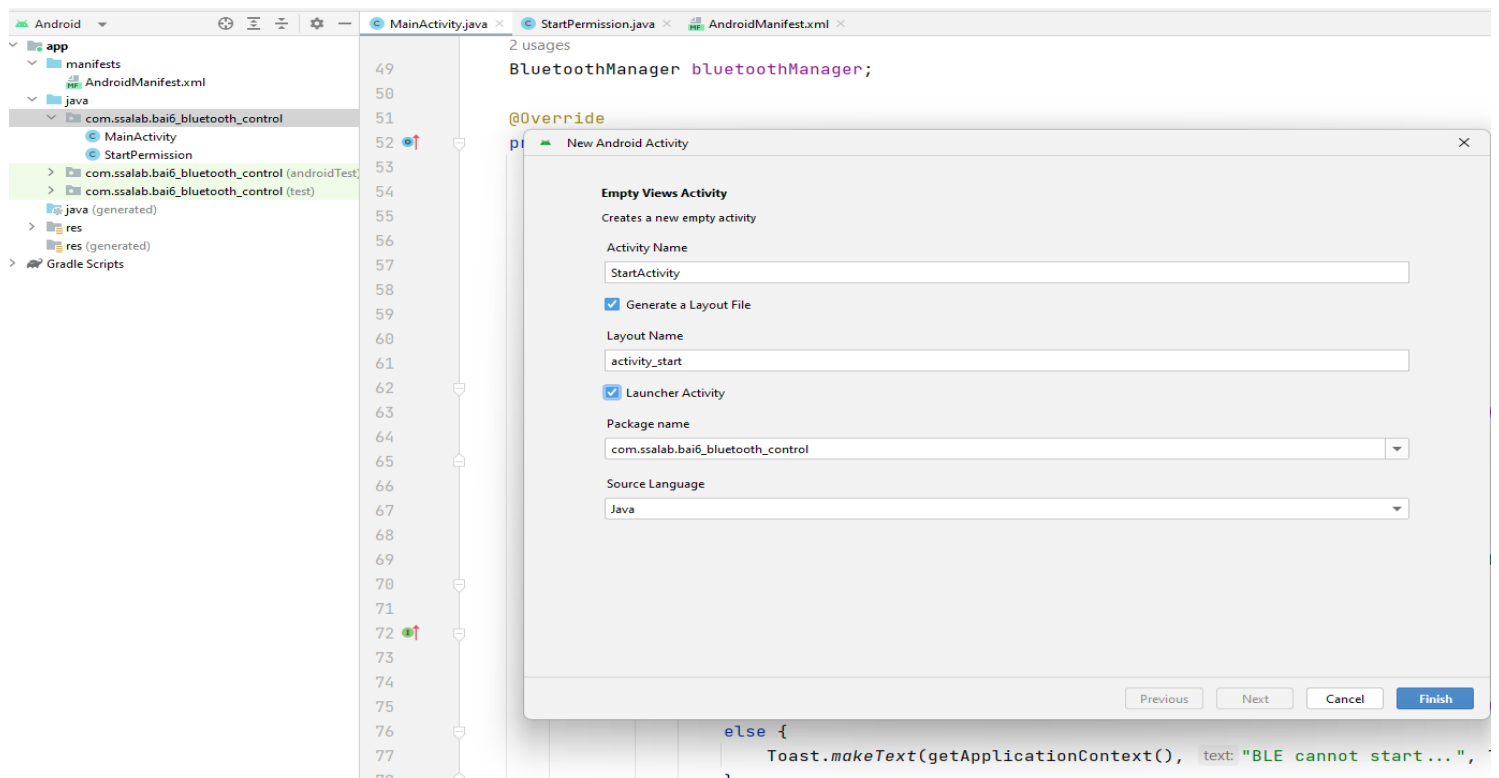
*Lưu ý: Một số máy android đời cao từ API >= 33 trở lên phải bật quyền vị trí*

Tạo 1 activity khởi động để check quyền có tên là **StartActivity.java** và layout tương ứng là



activity\_start.xml

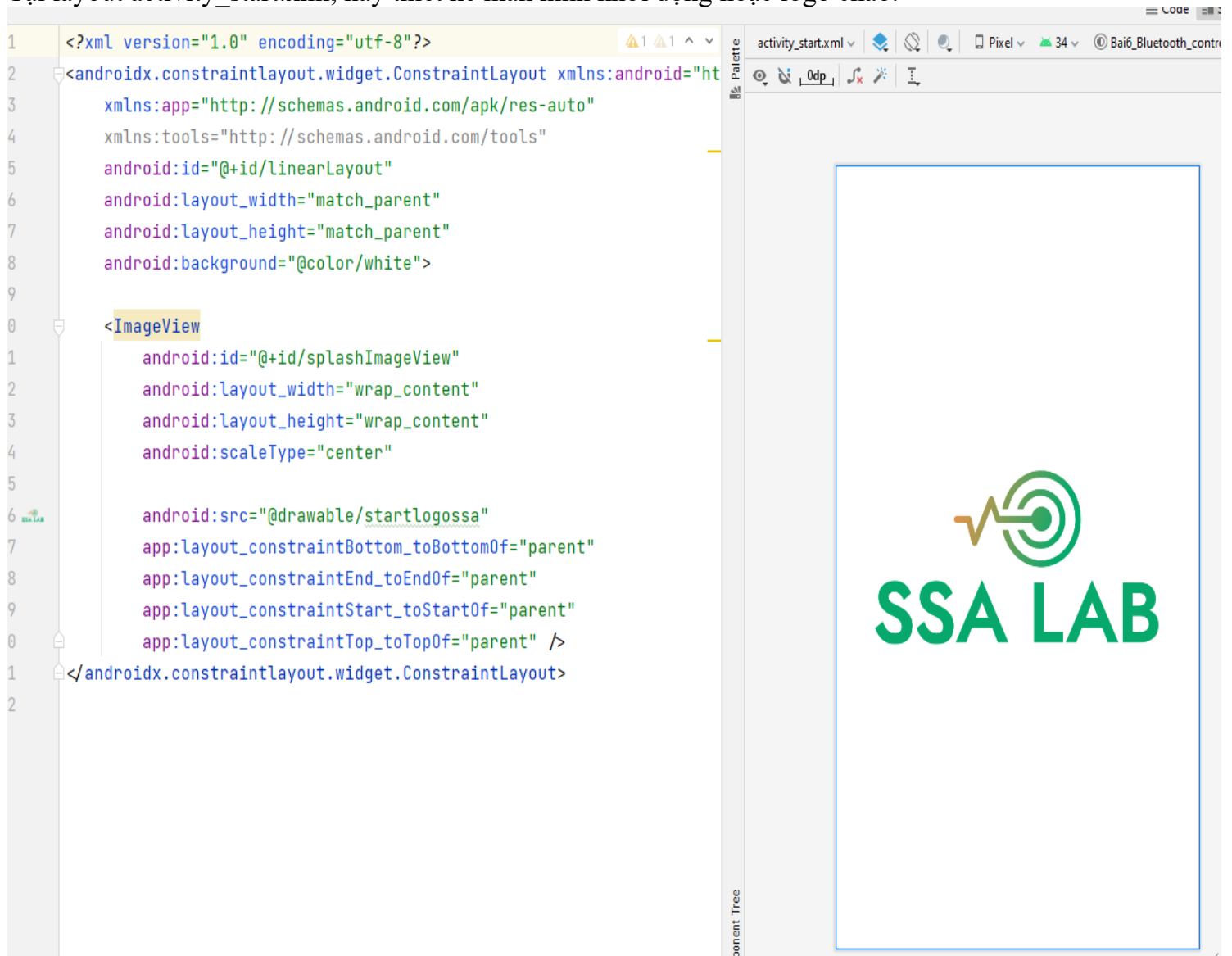
, đồng thời click chọn **launcher activity**:



Tại activity StartActivity.java, lập trình chuyển activity và check quyền ở trong luồng chính (oncreate) của StartActivity.java:

```
boolean start = StartPermission.requestBluetoothPermissions( activity: this);
if(start)
{
    new AlertDialog.Builder( context: StartActivity.this)
        .setTitle("Request Bluetooth Permissions")
        .setMessage("Checked Permission?")
        .setPositiveButton( text: "Yes", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                Intent mainIntent = new Intent( packageContext: StartActivity.this, MainActivity.class);
                startActivity(mainIntent);
                finish(); // Đóng màn hình khởi động
            }
        })
        .setNegativeButton( text: "No", listener: null)
        .show();
}
else
{
    new Handler().postDelayed(new Runnable() {
        @Override
        public void run() {
            // Chuyển đến màn hình chính sau khi kết thúc thời gian hiển thị
            Intent mainIntent = new Intent( packageContext: StartActivity.this, MainActivity.class);
            startActivity(mainIntent);
            finish(); // Đóng màn hình khởi động
        }
    }, delayMillis: 1000); // Thời gian hiển thị (ms)
}
Toast.makeText(getApplicationContext(), text: "Started", Toast.LENGTH_SHORT).show();
```

Tại layout activity\_start.xml, hãy thiết kế màn hình khởi động hoặc logo chào:



### Mở MainActivity.java

- Import thư viện

```
import androidx.activity.result.ActivityResult;
import androidx.activity.result.ActivityResultCallback;
import androidx.activity.result.ActivityResultLauncher;
import androidx.activity.result.contract.ActivityResultContracts;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothManager;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;

import java.io.OutputStream;
import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
```

## - Khai báo đối tượng

```
8 usages
ListView listView, scanListview;
2 usages
Button btON, btOFF, btSCAN;
2 usages
Intent intOnBLE;
2 usages
ActivityResultLauncher<Intent> EnableBLE;
9 usages
ArrayList<String> stringArrayList = new ArrayList<>();
3 usages
ArrayAdapter<String> arrayAdapter1;
12 usages
BluetoothAdapter bluetoothAdapter;
2 usages
BluetoothManager bluetoothManager;
```

## - Gán đối tượng trong hàm onCreate

```
btON = findViewById(R.id.btON);
btOFF = findViewById(R.id.btOFF);
btSCAN = findViewById(R.id.btSCAN);
bluetoothManager = (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
listView = findViewById(R.id.listview);
scanListview = findViewById(R.id.scanListview);
boolean start = StartPermission.requestBluetoothPermissions(activity: this);
if (start)
{
    Toast.makeText(getApplicationContext(), text: "Hãy khởi động lại APP...", Toast.LENGTH_LONG).show();
    return;
}
```

## - Kiểm tra và thiết lập quyền truy cập bluetooth. Lưu ý: thiết lập API>31 sau đó dùng **intent** kích hoạt Bluetooth

```
if (Build.VERSION.SDK_INT ≥ 31) bluetoothAdapter = bluetoothManager.getAdapter();
else bluetoothAdapter = bluetoothAdapter.getDefaultAdapter();
intOnBLE = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
EnableBLE = registerForActivityResult(new ActivityResultContracts.StartActivityForResult(),
    new ActivityResultCallback<ActivityResult>() {
        @Override
        public void onActivityResult(ActivityResult result) {
            int resultcode = result.getResultCode();
            if (resultcode == RESULT_OK)
                Toast.makeText(getApplicationContext(), text: "BLE started", Toast.LENGTH_SHORT).show();
            else {
                Toast.makeText(getApplicationContext(), text: "BLE cannot start...", Toast.LENGTH_LONG).show();
            }
        }
    });
```



- Vẫn trong onCreate, kích hoạt sự kiện click button bật Bluetooth

```
btON.setOnClickListener(new View.OnClickListener() {
    @SuppressWarnings("MissingPermission")
    @Override
    public void onClick(View v) {
        if (bluetoothAdapter == null) {
            Toast.makeText(getApplicationContext(), text: "Bluetooth not...", Toast.LENGTH_LONG).show();
        } else {
            if (!bluetoothAdapter.isEnabled()) {
                Toast.makeText(getApplicationContext(), text: "Bluetooth starting", Toast.LENGTH_LONG).show();
                EnableBLE.launch(intOnBLE);
            } else {
                Toast.makeText(getApplicationContext(), text: "Bluetooth started", Toast.LENGTH_SHORT).show();
            }
        }
    }
});
```

- Tiếp tục với sự kiện click off bluetooth

```
btOFF.setOnClickListener(new View.OnClickListener() {
    @SuppressWarnings("MissingPermission")
    @Override
    public void onClick(View v) {
        if(!bluetoothAdapter.isEnabled())
        {
            Toast.makeText(getApplicationContext(), text: "Bluetooth stopped", Toast.LENGTH_LONG).show();
        }
        else
        {
            if(Build.VERSION.SDK_INT ≥ 33)
            {
                Intent disableBLE = new Intent(Settings.ACTION_BLUETOOTH_SETTINGS);
                startActivity(disableBLE);
            }
            else bluetoothAdapter.disable();
            Toast.makeText(getApplicationContext(), text: "Bluetooth starting", Toast.LENGTH_LONG).show();
        }
    }
});
```

⇒ Đến bước này phải kích hoạt được on/off BLE là được.

- Tiếp tục với sự kiện click scan và hiển thị bluetooth. Viết chương trình click button **SCAN**

```
btSCAN.setOnClickListener(new View.OnClickListener() {
    @SuppressWarnings("MissingPermission")
    @Override
    public void onClick(View view) {
        bluetoothAdapter.startDiscovery();
        listView.setAdapter(null);
        scanListView.setAdapter(null);
        stringArrayList.clear();
        showBLEconnect();
    }
});
```



- Ngoài onCreate(), xây dựng hàm showBLEconnect hiển thị lên listview và scanlistview

```
usage
@SuppressLint("MissingPermission")
private void showBLEconnect() {
    @SuppressLint("MissingPermission") Set<BluetoothDevice> pairedDevices = bluetoothAdapter.getBondedDevices();
    String[] str = new String[pairedDevices.size()];
    // Tạo một ArrayList mới để lưu trữ danh sách các thiết bị Bluetooth đã kết nối
    ArrayList<String> connectedDevices = new ArrayList<>();
    arrayAdapter1 = new ArrayAdapter<String>(getApplicationContext(), android.R.layout.simple_list_item_1, stringArrayList);
    int index = 0;
    if (pairedDevices.size() >= 0) {
        try {
            for (BluetoothDevice device : pairedDevices) {
                str[index] = device.getName() + "\n" + device.getAddress();
                connectedDevices.add(str[index]); // Thêm vào danh sách các thiết bị đã kết nối
                index++;
            }
            // Loại bỏ các phần tử đã tồn tại trong connectedDevices khỏi stringArrayList
            stringArrayList.removeAll(connectedDevices);
            ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(getApplicationContext(), android.R.layout.simple_list_item_1, str);
            listView.setAdapter(arrayAdapter);
        } catch (Exception exception) {
        }
    }
    scanListView.setAdapter(arrayAdapter1);
}
```

- Ngoài onCreate(), xây dựng hàm scan BLE bằng cách phát quảng bá:

```
BroadcastReceiver broadcastReceiver = new BroadcastReceiver() {
    @SuppressLint("MissingPermission")
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            stringArrayList.add(device.getName() + "\n" + device.getAddress());
            // Khai báo HashSet để theo dõi các phần tử đã thêm vào danh sách
            HashSet<String> uniqueItems = new HashSet<>();
            for (String item : stringArrayList) {
                // Thêm phần tử vào HashSet, nếu nó chưa tồn tại
                uniqueItems.add(item);
            }
            // Xóa toàn bộ phần tử trong stringArrayList
            stringArrayList.clear();
            // Thêm lại các phần tử đã loại bỏ trùng lặp từ HashSet vào stringArrayList
            stringArrayList.addAll(uniqueItems);
            ArrayAdapter<String> adapter = (ArrayAdapter<String>) listView.getAdapter();
            if (adapter != null) {
                for (int i = 0; i < adapter.getCount(); i++) {
                    if (adapter.getItem(i).equals(device.getName() + "\n" + device.getAddress())) // LOẠI BỎ PHẦN TỬ ĐÃ PAIRED
                    {
                        stringArrayList.remove(device.getName() + "\n" + device.getAddress());
                    }
                }
            }
            arrayAdapter1.notifyDataSetChanged();
        }
    }
};
```

- Trong `OnCreate()`, dùng **intentFilter** để kích hoạt quét và hiển thị device BLE được tìm thấy:

```
IntentFilter intentFilter = new IntentFilter(BluetoothDevice.ACTION_FOUND);  
registerReceiver(broadcastReceiver, intentFilter);
```

Ở Ngoài luồng chính trong `MainActivity.java`, lập trình hủy quảng báo khi app thoát:

```
@Override  
protected void onDestroy() {  
    super.onDestroy();  
    unregisterReceiver(broadcastReceiver);  
}
```

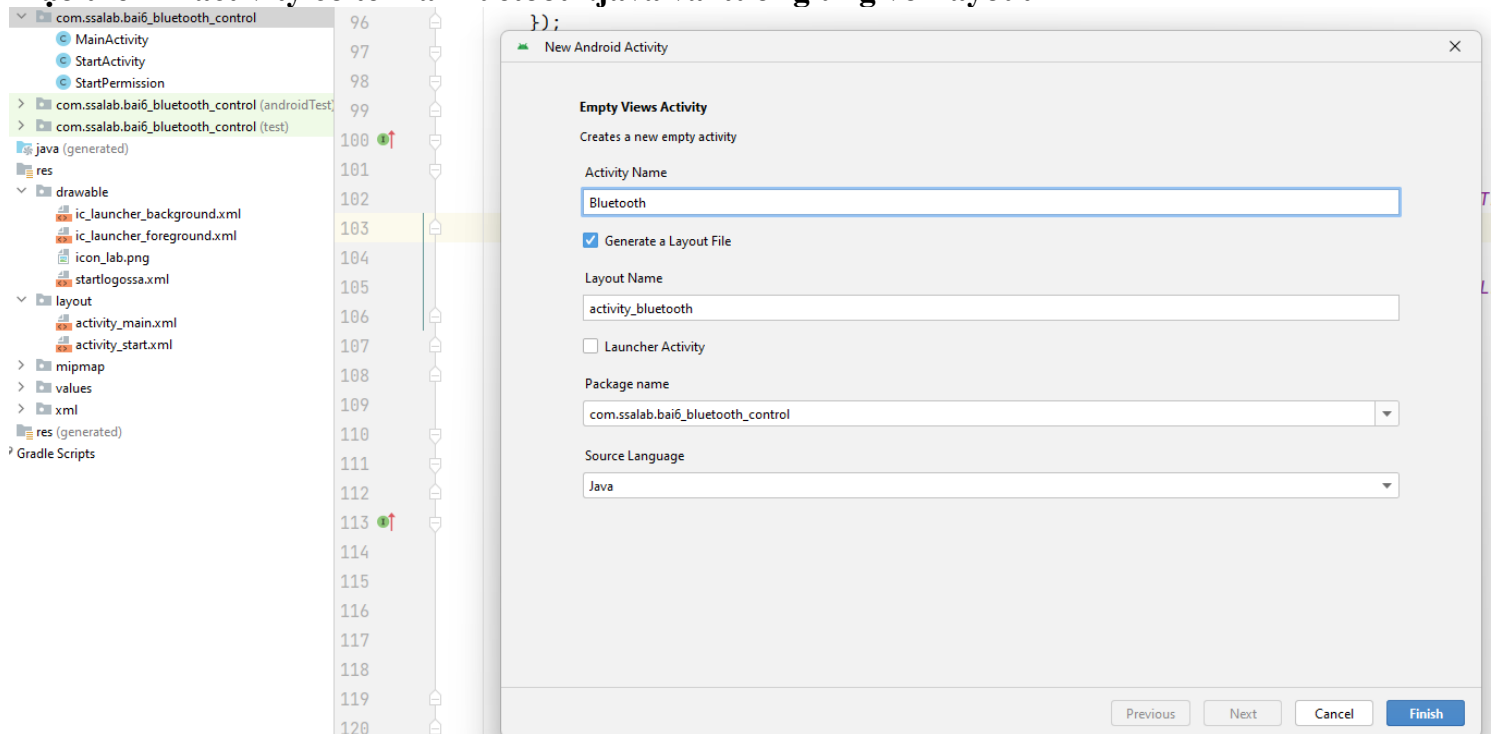
Đến bước này chạy thử thì phải hiển thị được list thiết bị đã pair và chưa pair sau khi nhấn Scan thì sang Phần II.

## Phần II: Tương tác ghép đôi và bỏ ghép đôi



activity\_bluetooth.xml

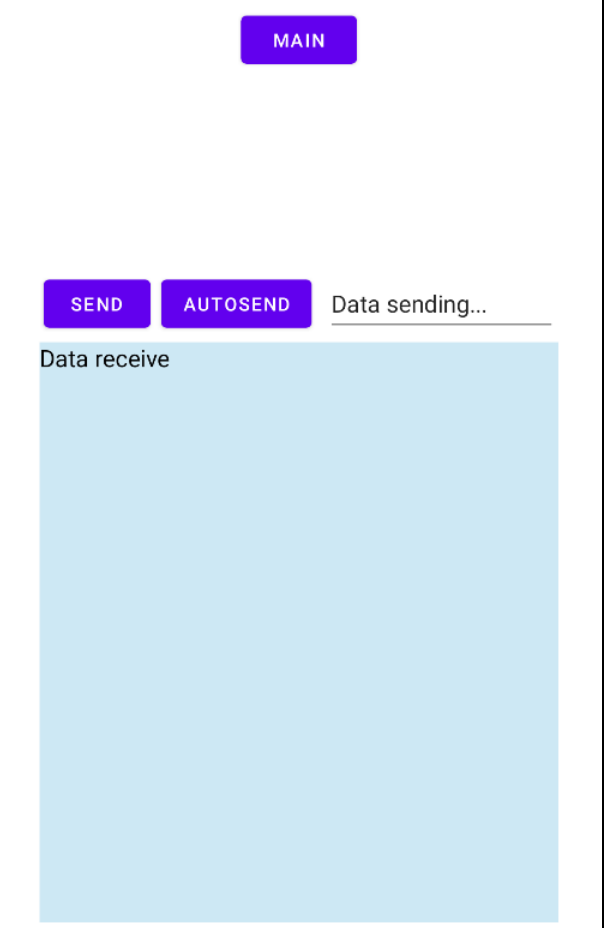
**Tạo thêm 1 activity có tên là Bluetooth.java và tương ứng với layout**



activity\_bluetooth.xml

Thiết kế giao diện

như hình:

	<p>Thiết lập giao diện với các button, edittext và textview có ID sau:</p> <ul style="list-style-type: none"> <li>- Nút Main: btBack</li> <li>- Nút Send: btSend</li> <li>- Nút AUTOSEND: btAuto</li> <li>- Edittext (mặc định hiển thị “Data sending”) : edSend</li> <li>- Textview (mặc định hiển thị “Data receive”): tvReceive</li> </ul>
--	---

Quay trở lại MainActivity.java, lập trình Thao tác các phần tử hiển thị trên **listview** chứa các thiết bị đã paired, click để chuyển sang **Bluetooth.java**

```

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @SuppressWarnings("MissingPermission")
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, int position, long l) {
        // Lấy adapter từ ListView
        ArrayAdapter<String> adapter = (ArrayAdapter<String>) listView.getAdapter();

        // Lấy danh sách các mục từ adapter
        List<String> listItems = new ArrayList<>();
        for (int i = 0; i < adapter.getCount(); i++) {
            listItems.add(adapter.getItem(i));
        }
        String selectedItem = listItems.get(position);
        String[] parts = selectedItem.split(regex: "\\n");
        if (parts.length >= 2) {
            String deviceName = parts[0];
            String deviceAddress = parts[1];
            Intent intent = new Intent(packageContext: MainActivity.this, Bluetooth.class);
            intent.putExtra(name: "DEVICE_ADDRESS", deviceAddress);
            startActivity(intent);
        }
    }
});
scanListView.setAdapter(arrayAdapter1);

```

Lập trình nhấn giữ 1 vị trí trên listview để bỏ pair. Đầu tiên, lập trình hàm unpair ở ngoài luồng chính trong MainActivity.java:

```
1 usage
private void unpairDevice(BluetoothDevice device) {
    try {
        Method method = device.getClass().getMethod("removeBond", (Class[]) null);
        method.invoke(device, (Object[]) null);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Quay trở lại luồng chính, lập trình nhấn giữ 1 vị trí trên listview để bỏ pair:

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    1 usage
    private static final long LONG_PRESS_DURATION = 1500; // Thời gian giữ lâu (đơn vị mili giây)
    1 usage
    private Handler handler = new Handler();
    3 usages
    private boolean isLongPress = false;
    @Override
    public boolean onItemClick(AdapterView<?> adapterView, View view, int position, long l) {
        isLongPress = true;
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                if (isLongPress) {
                    ArrayAdapter<String> adapter = (ArrayAdapter<String>) listView.getAdapter();
                    List<String> listItems = new ArrayList<>();
                    for (int i = 0; i < adapter.getCount(); i++) {
                        listItems.add(adapter.getItem(i));
                    }
                    String selectedItem = listItems.get(position);
                    String[] parts = selectedItem.split("regex: \"\\n\"");
                    if (parts.length >= 2) {
                        String deviceName = parts[0];
                        String deviceAddress = parts[1];
                        BluetoothDevice selectedDevice = bluetoothAdapter.getRemoteDevice(deviceAddress);
                        // Hiển thị dialog hỏi người dùng có muốn bỏ pair không
                        new AlertDialog.Builder(context MainActivity.this)
                            .setTitle("Confirm")
                            .setMessage(deviceName + " unpair?")
                            .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                                @Override
                                public void onClick(DialogInterface dialogInterface, int i) {
                                    unpairDevice(selectedDevice);
                                    Toast.makeText(getApplicationContext(), deviceName + "\\n" + deviceAddress + " unpaired", Toast.LENGTH_SHORT).show();
                                }
                            }).setNegativeButton("No", listener: null).show();
                    }
                    isLongPress = false; // Đặt lại trạng thái long press
                }}, LONG_PRESS_DURATION);
        return true; // Trả về true để ngăn sự kiện click ngắn xảy ra sau khi long press
    }
});
```

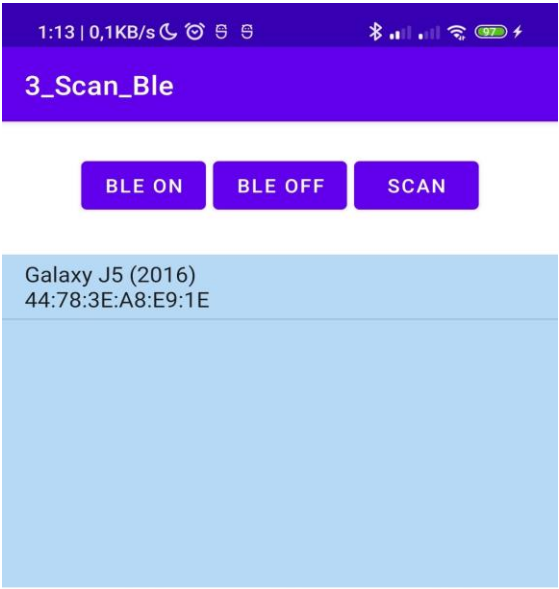
- Thao tác các phần tử hiển thị trên **ScanListView** chứa các thiết bị chưa pair để thực hiện ghép đôi

```
scanListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, int position, long l) {
        String selectedItem = stringArrayList.get(position);
        String[] parts = selectedItem.split( regex: "\\n");
        if (parts.length ≥ 2) {
            String deviceName = parts[0];
            String deviceAddress = parts[1];
            BluetoothDevice selectedDevice = null;
            Set<BluetoothDevice> pairedDevices = bluetoothAdapter.getBondedDevices();
            boolean foundInPairedDevices = false;
            for (BluetoothDevice device : pairedDevices) {
                if (device.getName().equals(deviceName) && device.getAddress().equals(deviceAddress)) {
                    foundInPairedDevices = true;
                    break;}}
            if (!foundInPairedDevices) {
                // Thiết bị chưa được pair, thực hiện pair
                try {
                    selectedDevice = bluetoothAdapter.getRemoteDevice(deviceAddress);
                    if (selectedDevice ≠ null) {
                        BluetoothDevice finalSelectedDevice = selectedDevice;
                        new AlertDialog.Builder( context: MainActivity.this)
                            .setTitle("Confirm")
                            .setMessage(deviceName + " can be pair???)
                            .setPositiveButton( text: "Yes", new DialogInterface.OnClickListener() {
                                @Override
                                public void onClick(DialogInterface dialogInterface, int i) {
                                    @SuppressWarnings("MissingPermission") boolean paired = finalSelectedDevice.createBond();
                                    if (!paired) {
                                        Toast.makeText(getApplicationContext(), text: deviceName + "\\n" + deviceAddress + " unpaired", Toast.LENGTH_SHORT).show();
                                    }
                                }
                            })
                            .setNegativeButton( text: "No", listener: null).show();
                    } else {
                        Toast.makeText(getApplicationContext(), text: "Cannot find " + deviceName, Toast.LENGTH_SHORT).show();
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                    Toast.makeText(getApplicationContext(), text: "Error paired" + deviceName, Toast.LENGTH_SHORT).show();
                }
            } else {
                // Thiết bị đã được pair
                Toast.makeText(getApplicationContext(), text: deviceName + " paired", Toast.LENGTH_SHORT).show();
            }
        }
    }
});
```

Tiếp tục lập trình hàm auto refresh list thiết bị ở trong luồng chính MainActivity.java:

```
Handler autoRefreshHandler = new Handler();
Runnable autoRefreshRunnable = new Runnable() {
    @Override
    public void run() {
        showBLEconnect();
        // Lên lịch để tự động làm mới lại sau khoảng thời gian
        autoRefreshHandler.postDelayed( r: this, delayMillis: 1000);
    }
};
autoRefreshHandler.postDelayed(autoRefreshRunnable, delayMillis: 1000);
```

Kết quả





### Phần III: Lập trình truyền nhận qua bluetooth:

Trong Bluetooth.java, khai báo các thư viện:

```
import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothManager;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.nio.charset.StandardCharsets;
import java.util.LinkedList;
import java.util.UUID;
```

Tiếp theo, lập trình khai báo các đối tượng trong Bluetooth.java:

```
private Button btBack, btSend, btAuto;
1 usage
EditText edSend;
2 usages
TextView tvReceive;
4 usages
boolean autoSend = false;
4 usages
BluetoothAdapter bluetoothAdapter;
5 usages
BluetoothSocket bluetoothSocket;
2 usages
BluetoothManager bluetoothManager;
2 usages
private BluetoothDevice esp32Device;
4 usages
private ConnectedThread connectedThread;
3 usages
private Handler handler = new Handler();
3 usages
private Runnable autoSendRunnable;
1 usage
private static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB"); // UUID của dịch vụ Serial Port Profile (SPP)
2 usages
String deviceAddress;
4 usages
private LinkedList<String> receivedDataList = new LinkedList<>();
```

Trong onCreate của Bluetooth.java, khai báo các nút nhấn và tương tác:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_bluetooth);
    StartPermission.requestBluetoothPermissions(activity: this);
    btBack = findViewById(R.id.btBack);
    btSend = findViewById(R.id.btSend);
    btAuto = findViewById(R.id.btAuto);
    edSend = findViewById(R.id.edSend);
    tvReceive = findViewById(R.id.tvReceive);
    bluetoothManager = (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
    if (Build.VERSION.SDK_INT ≥ 31) bluetoothAdapter = bluetoothManager.getAdapter();
    else bluetoothAdapter = bluetoothAdapter.getDefaultAdapter();
}
```

Tiếp theo vẫn trong luồng chính của **Bluetooth.java**, hãy lập trình nhận địa chỉ của thiết bị bluetooth từ mainactivity truyền sang.

```
// Lấy địa chỉ Bluetooth từ Intent
deviceAddress = getIntent().getStringExtra(name: "DEVICE_ADDRESS");
esp32Device = bluetoothAdapter.getRemoteDevice(deviceAddress);
```

Ở ngoài luồng chính trong **Bluetooth.java**, tạo hàm hủy Bluetooth khi thoát activity:

```
@Override
protected void onDestroy() {
    super.onDestroy();
    if (bluetoothSocket ≠ null) {
        try {
            bluetoothSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Ở ngoài luồng chính trong Bluetooth.java, lập trình class **ConnectedThread** định nghĩa phương thức truyền bluetooth. Cụ thể trong class **ConnectedThread**, tạo **ConnectedThread**(BluetoothSocket socket) :

```
private class ConnectedThread extends Thread {
    1 usage
    private final InputStream inputStream;
    1 usage
    private final OutputStream outputStream;
    no usages
    ConnectedThread(BluetoothSocket socket) {
        InputStream tempIn = null;
        OutputStream tempOut = null;
        try {
            tempIn = socket.getInputStream();
            tempOut = socket.getOutputStream();
        } catch (IOException e) {
            e.printStackTrace();
        }
        inputStream = tempIn;
        outputStream = tempOut;
    }
}
```

Vẫn trong class **ConnectedThread**, tạo hàm **addReceivedDataToList ()** và **Write (String data)**

```

1 usage
private void addReceivedDataToList(String data) {
    // Kiểm tra xem danh sách có quá 10 phần tử không
    if (receivedDataList.size() ≥ 10) {
        // Nếu có, loại bỏ phần tử đầu tiên để giữ cho danh sách không vượt quá 10 phần tử
        receivedDataList.remove(index: 0);
    }
    // Thêm dòng dữ liệu mới vào danh sách
    receivedDataList.add(data);
}

```

```

no usages
void write(String data) {
    byte[] msgBuffer = data.getBytes();
    try {
        outputStream.write(msgBuffer);
    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

Vẫn trong class **ConnectedThread**, tạo hàm **handleReceivedData ()**:

1 usage

```

private void handleReceivedData(String receivedData) {

    addReceivedDataToList(receivedData);
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            // Hiển thị dữ liệu từ danh sách lên TextView
            StringBuilder displayText = new StringBuilder();
            for (String data : receivedDataList) {
                displayText.append(data).append("\n");
            }
            tvReceive.setText(displayText.toString());
        }
    });
}

```

Vẫn trong class **ConnectedThread**, tạo hàm **Run ()** :

```

public void run() {
    try {
        InputStreamReader reader = new InputStreamReader(inputStream, StandardCharsets.UTF_8);
        StringBuilder receivedDataBuilder = new StringBuilder();
        char[] buffer = new char[1024];
        int bytesRead;
        while (true) {
            bytesRead = reader.read(buffer);
            if (bytesRead == -1) {
                // Kết thúc dữ liệu, có thể xử lý dữ liệu ở đây nếu cần
                break;
            }
            receivedDataBuilder.append(buffer, offset: 0, bytesRead);
            // Kiểm tra xem có ký tự kết thúc (ví dụ: '$' hoặc ' ') trong dữ liệu nhận được không
            String receivedData = receivedDataBuilder.toString();
            int newlineIndex = receivedData.indexOf('$');
            // Kiểm tra thêm cho ký tự khoảng trắng
            if (newlineIndex == -1) newlineIndex = receivedData.indexOf(' ');
            if (newlineIndex == -1) newlineIndex = receivedData.indexOf('\n');
            if (newlineIndex == -1) newlineIndex = receivedData.indexOf('\r');

            if (newlineIndex != -1) {
                // Nếu có ký tự kết thúc, xử lý dữ liệu trước khi xuống dòng
                handleReceivedData(receivedData.substring(0, newlineIndex));
                // Xóa dữ liệu đã xử lý khỏi StringBuilder
                receivedDataBuilder.delete(0, newlineIndex + 1);
            }
        }
    } catch (IOException e) { e.printStackTrace(); }
}

```

Ngoài luồng chính, bổ sung khai báo đối tượng

```

TextView tvReceive;
1 usage
boolean autoSend = false;
4 usages
BluetoothAdapter bluetoothAdapter;
5 usages
BluetoothSocket bluetoothSocket;
2 usages
BluetoothManager bluetoothManager;
2 usages
private BluetoothDevice esp32Device;
1 usage
private Handler handler = new Handler();
1 usage
private Runnable autoSendRunnable;
1 usage
private static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB"); // UUID của dịch vụ Serial Port Profile (SPP)
2 usages
String deviceAddress;
4 usages
private LinkedList<String> receivedDataList = new LinkedList<>();
3 usages
private ConnectedThread connectedThread;

```

Ở Trong luồng chính trong Bluetooth.java, lập trình kết nối Bluetooth:

```
try {
    bluetoothSocket = esp32Device.createRfcommSocketToServiceRecord(MY_UUID);
    bluetoothSocket.connect();
    connectedThread = new ConnectedThread(bluetoothSocket);
    connectedThread.start();
} catch (IOException e) {
    e.printStackTrace();
    Toast.makeText(context, this, text: "Failed to connect to Bluetooth device", Toast.LENGTH_SHORT).show();
    finish();
}
```

Tiếp tục trong luồng chính trong Bluetooth.java, lập trình kết nối các nút nhấn:

```
btBack.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        finish(); // Kết thúc Activity và quay lại cửa sổ chính
    }
});
```

Lập trình nút nhấn send và nhận dữ liệu.

```
btSend.setOnClickListener(new View.OnClickListener() {
    @SuppressWarnings("MissingPermission")
    @Override
    public void onClick(View view) {
        handler.removeCallbacks(autoSendRunnable);
        autoSend = false;
        connectedThread.write(data: "YourDataToSend" + "\n");
    }
});
```

Lập trình nút nhấn send dữ liệu liên tục:

```
int timeDelay = 100;
autoSendRunnable = new Runnable() {
    @Override
    public void run() {
        if(autoSend)
        { // Gửi dữ liệu tự động
            connectedThread.write(data: "AutoToSend" + "\n");
            // Lập lịch chạy lại Runnable sau 100ms
            handler.postDelayed(this, timeDelay);
        }
    }
};

btAuto.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        autoSend = !autoSend;
        // Bắt đầu lập lịch chạy Runnable sau 100ms
        handler.postDelayed(autoSendRunnable, timeDelay);
    }
});
```

