Mainactivity.java

package com.example.smartglovesignlanguageconversionapp;

import androidx.activity.result.ActivityResultCallback;

import androidx.activity.result.ActivityResultLauncher;

import androidx.activity.result.contract.ActivityResultContracts;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import android.annotation.SuppressLint;

import android.app.Activity;

import android.bluetooth.BluetoothAdapter;

import android.bluetooth.BluetoothDevice;

import android.content.Intent;

import android.content.pm.PackageManager;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.TextView;

import android.widget.Toast;

import java.util.Set;

public class MainActivity extends AppCompatActivity {

private static final int *REQUEST\_ENABLE\_BT* = 0;

private static final int *REQUEST\_DISCOVER\_BT* = 1;

TextView mStatusBlueTv, mPairedTv;

ImageView mblueTv;

Button mOnBtn, moffBtn, mDiscoverableBtn, mPairedBtn;

BluetoothAdapter Adapter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

mStatusBlueTv = findViewById(R.id.*statusBluetoothTv*);

mPairedTv = findViewById(R.id.*pairedTv*);

mblueTv = findViewById(R.id.*bluettothIv*);

mOnBtn = findViewById(R.id.*onBtn*);

moffBtn = findViewById(R.id.*offBtn*);

mDiscoverableBtn = findViewById(R.id.*onDiscoverable*);

mPairedBtn = findViewById(R.id.*pairedBtn*);

ActivityResultLauncher<Intent> launcher =

registerForActivityResult(new ActivityResultContracts.StartActivityForResult(),

result -> {

if (result.getResultCode() == Activity.*RESULT\_OK*) {

Intent data = result.getData();

}

});

Adapter = BluetoothAdapter.*getDefaultAdapter*();

if (Adapter == null) {

mStatusBlueTv.setText("BLUETOOTH IS NOT AVAILABLE");

} else {

mStatusBlueTv.setText("BLUETOOTH IS AVAILABLE");

}

if (Adapter.isEnabled()) {

mblueTv.setImageResource(R.drawable.*ic\_action\_on*);

} else {

mblueTv.setImageResource(R.drawable.*ic\_action\_off*);

}

mOnBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

if (!Adapter.isEnabled()) {

showToast("turning on bluetooth");

Intent intent = new Intent(BluetoothAdapter.*ACTION\_REQUEST\_ENABLE*);

launcher.launch(intent);

} else {

showToast("bluetooth already on");

}

}

});

moffBtn.setOnClickListener(new View.OnClickListener() {

@SuppressLint("MissingPermission")

@Override

public void onClick(View view) {

if (Adapter.isEnabled()) {

Adapter.disable();

showToast("turning bluetooth off");

mblueTv.setImageResource(R.drawable.*ic\_action\_off*);

} else {

showToast("bluetooth is already off");

}

}

});

mDiscoverableBtn.setOnClickListener(new View.OnClickListener() {

@SuppressLint("MissingPermission")

@Override

public void onClick(View view) {

if (!Adapter.isDiscovering()) {

showToast("making your device discoverable");

Intent intent = new Intent(BluetoothAdapter.*ACTION\_REQUEST\_DISCOVERABLE*);

launcher.launch(intent);

}

}

});

mPairedBtn.setOnClickListener(new View.OnClickListener() {

@SuppressLint("MissingPermission")

@Override

public void onClick(View view) {

if (Adapter.isEnabled()) {

mPairedTv.setText("paired devices");

@SuppressLint("MissingPermission") Set<BluetoothDevice> devices = Adapter.getBondedDevices();

for (BluetoothDevice device : devices) {

mPairedTv.append("\nDevices" + device.getName() + "," + device);

}

} else {

showToast("turn on bluetooth to get paired devices");

}

}

});

}

@Override

protected void onActivityResult ( int requestCode, int resultCode, Intent data){

switch (requestCode) {

case *REQUEST\_ENABLE\_BT*:

if (resultCode == *RESULT\_OK*) {

mblueTv.setImageResource(R.drawable.*ic\_action\_on*);

showToast("bluetooth is on");

} else {

showToast("couldn't on bluetooth");

}

break;

}

super.onActivityResult(requestCode, resultCode, data);

}

private void showToast(String msg){

Toast.*makeText*(this, msg, Toast.*LENGTH\_SHORT*).show();

}

}