ScannerWedge
Quick Start Guide V1.0

contents

Support Devices List

BarCode	Honeywell N3680 (Serial Port) NewLand EM2096 (Serial Port)

ScannerWedge Introduce

Function

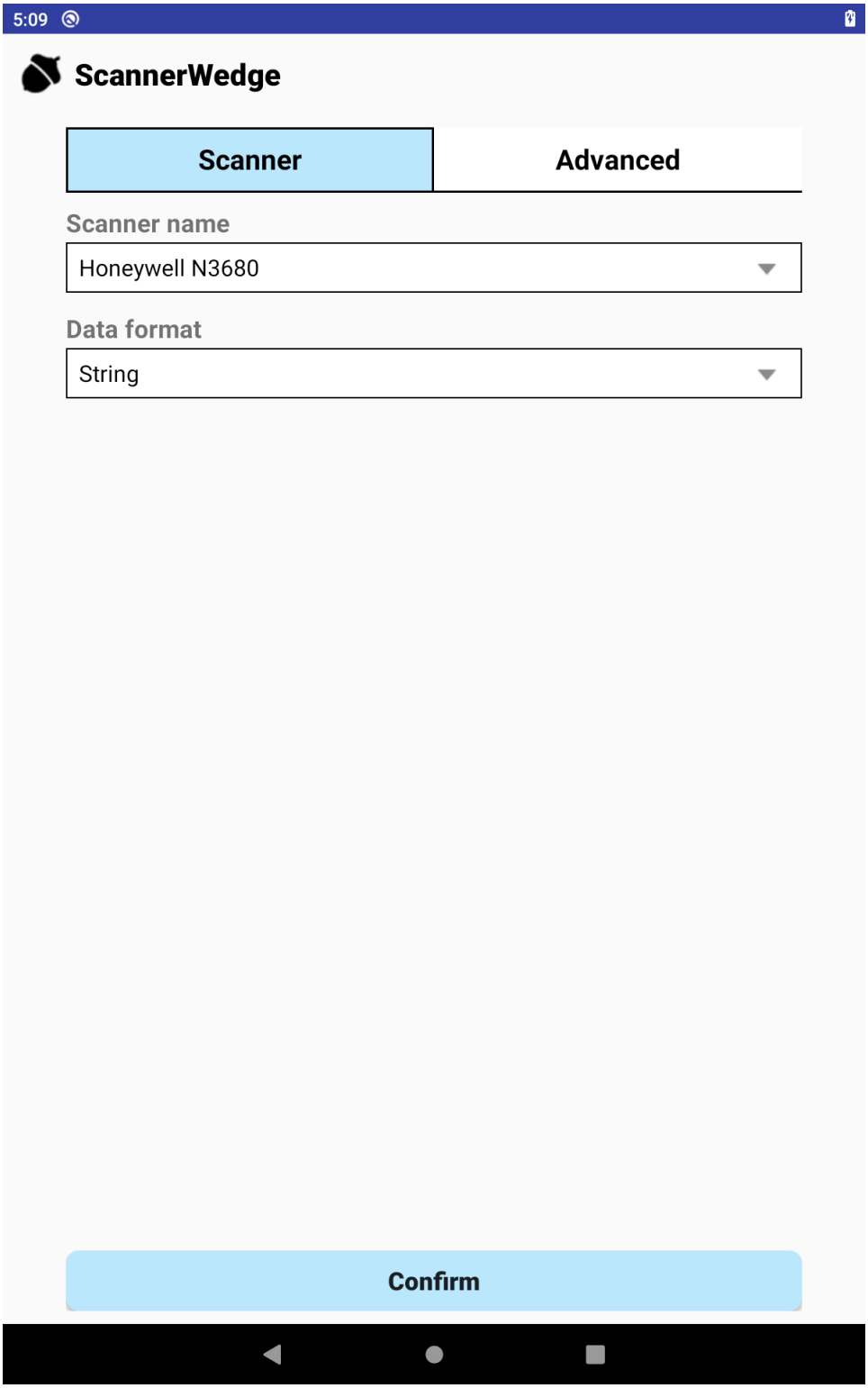
ScannerWedge app is mainly used to simplify the development and use of barcode scanners by customers.

It integrates the functions of various barcode scanner modules (such as Honeywell N3680，NewLand EM2096). It also provides broadcast trigger interface and floating button trigger mode for barcode scanner. The broadcast trigger interface cannot be closed and configured. It is mainly used for calling by third-party applications. The floating button trigger mode can be configured and closed.

In addition, ScannerWedge has two processing methods for the scanned data. One is to simulate the keyboard input method to directly convert the scanned data into keyboard input data, and the other is to forward the scanned data in the form of broadcast so that the third-party applications can receive the data.

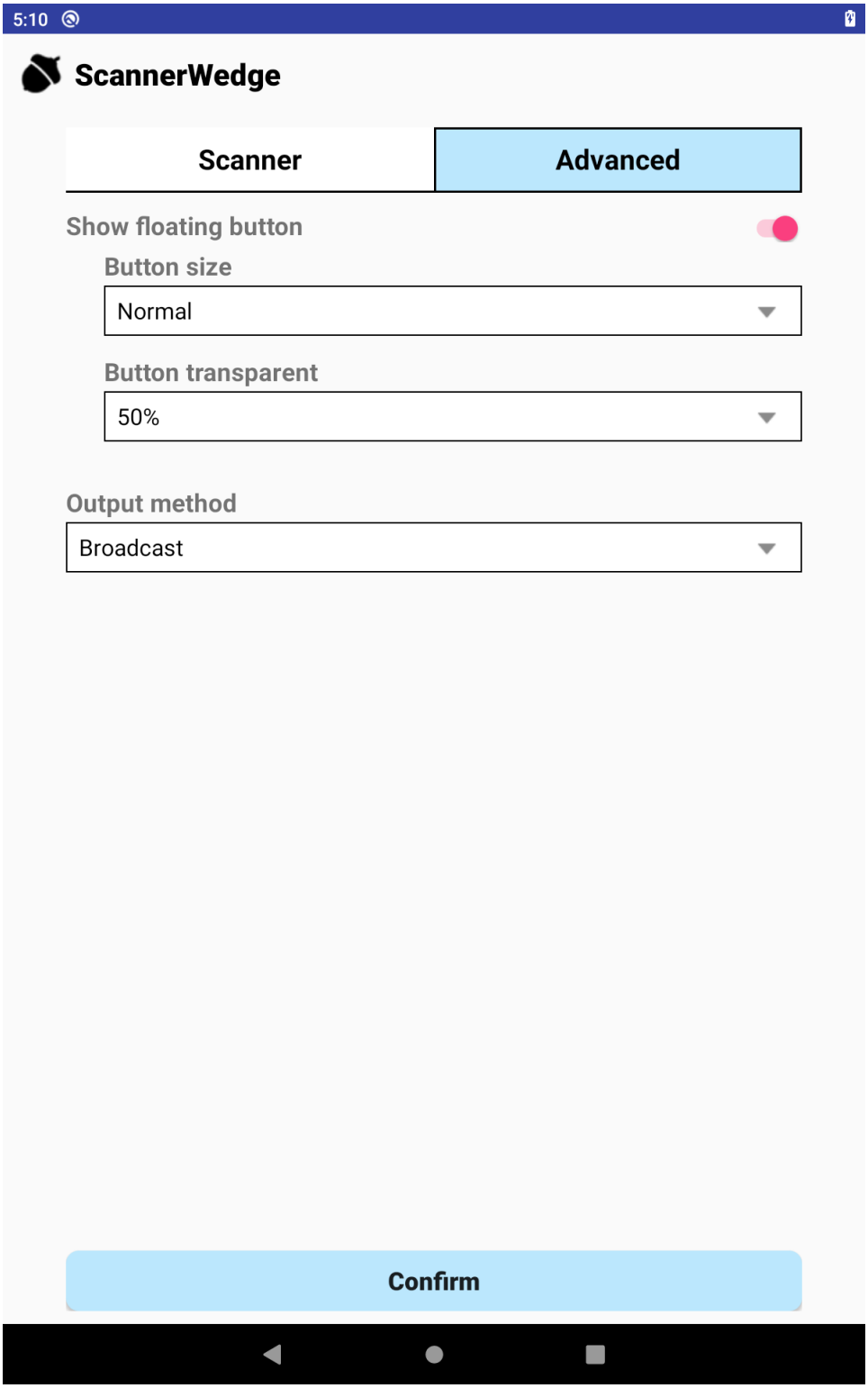
Customers only need to make some simple configuration for ScannerWedge, so they can easily use barcode scanner to collect data.

Scanner Menu



Name	Description
Scanner name	The barcode scanner model name. 1. Honeywell N3680 2. NewLand EM2096
Data format	1.String

Advanced Menu



Name	Description
Show floating button	Determines whether to display the floating button
Button size	Size of floating button 1.Small 2.Normal

	3.Large
Button transparent	Transparent of floating button 100% 90% 80% 70% 60% 50% 40% 30% 20% 10% 0%
Output Method	Set the flow direction of the scanned data. 1. Keyboard 2. Broadcast

Keyboard Mode

Keyboard mode is to send the scanned data to the system in the form of analog keyboard input. In this mode, the scanned data will be directly filled into the edit box where the focus is located

Broadcast Mode

Broadcast mode is to send the scanned data to the third-party app in the form of broadcast. If you want to get the data, you need to listen to the "com.advantech.scannerwedge.TRANSFER_DATA" broadcast in the app, and get the data from the extra string "barcode_data" after receiving the broadcast information. Please refer to ScannerWedgeSample app source code for details.

ScannerWedgeSample Introduce

ScannerWedgeSample is an open source sample app, which is mainly used to give users reference on how to trigger a scan through broadcast and how to receive the scanned data through broadcast. The source code address of ScannerWedgeSample app is: https://github.com/AIM-Android/ScannerWedgeSample

ScannerWedge Usage

Device Setting

Please check that the Barcode read has been set to Serial Port mode and decoding session timeout is greater than 10 seconds.

a. Honeywell N3680 module is checked as follows

USB HID

Scan the following code to program the scan engine for USB HID bar code scanners.

USB HID Bar Code Scanner

USB Serial

Scan the following code to program the scan engine to emulate a regular RS232-based COM Port. If you are using a Microsoft® Windows® PC, you will need to download a driver from the Honeywell website (www.honeywellaidc.com). The driver will use the next available COM Port number. Apple® Macintosh computers recognize the scan engine as a USB CDC class device and automatically use a class driver.

USB Serial

b. NewLand EM2096 module is checked as follows

■ **Solution - Step by Step**:

1. Scan below barcodes in sequence, without installing any Virtual COM Driver.
   Enter Setup → **USB COM Port Emulation** → **Exit Setup**

Enter Setup

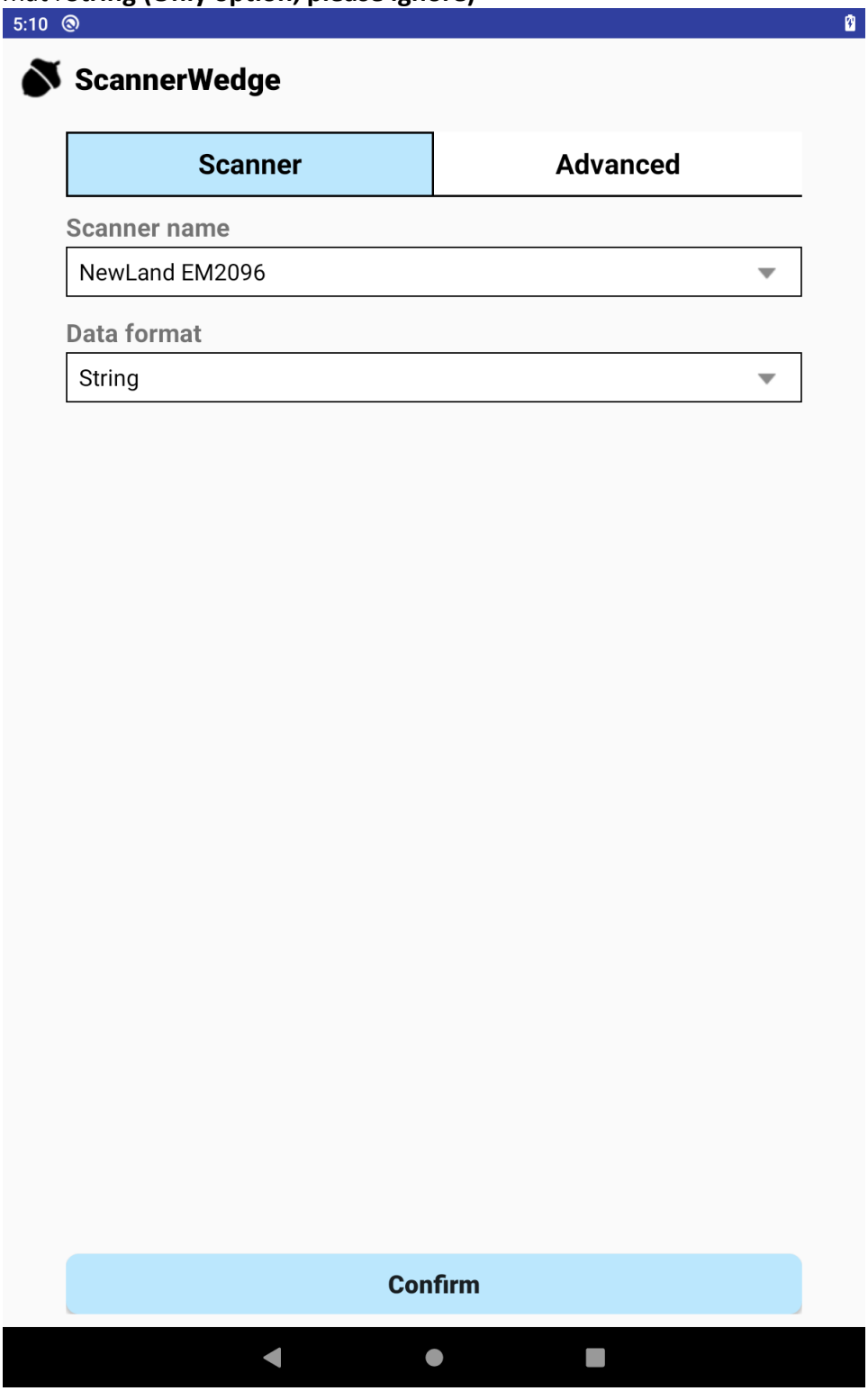
USB COM Port Emulation

** **Exit Setup**

ScannerWedge Setting

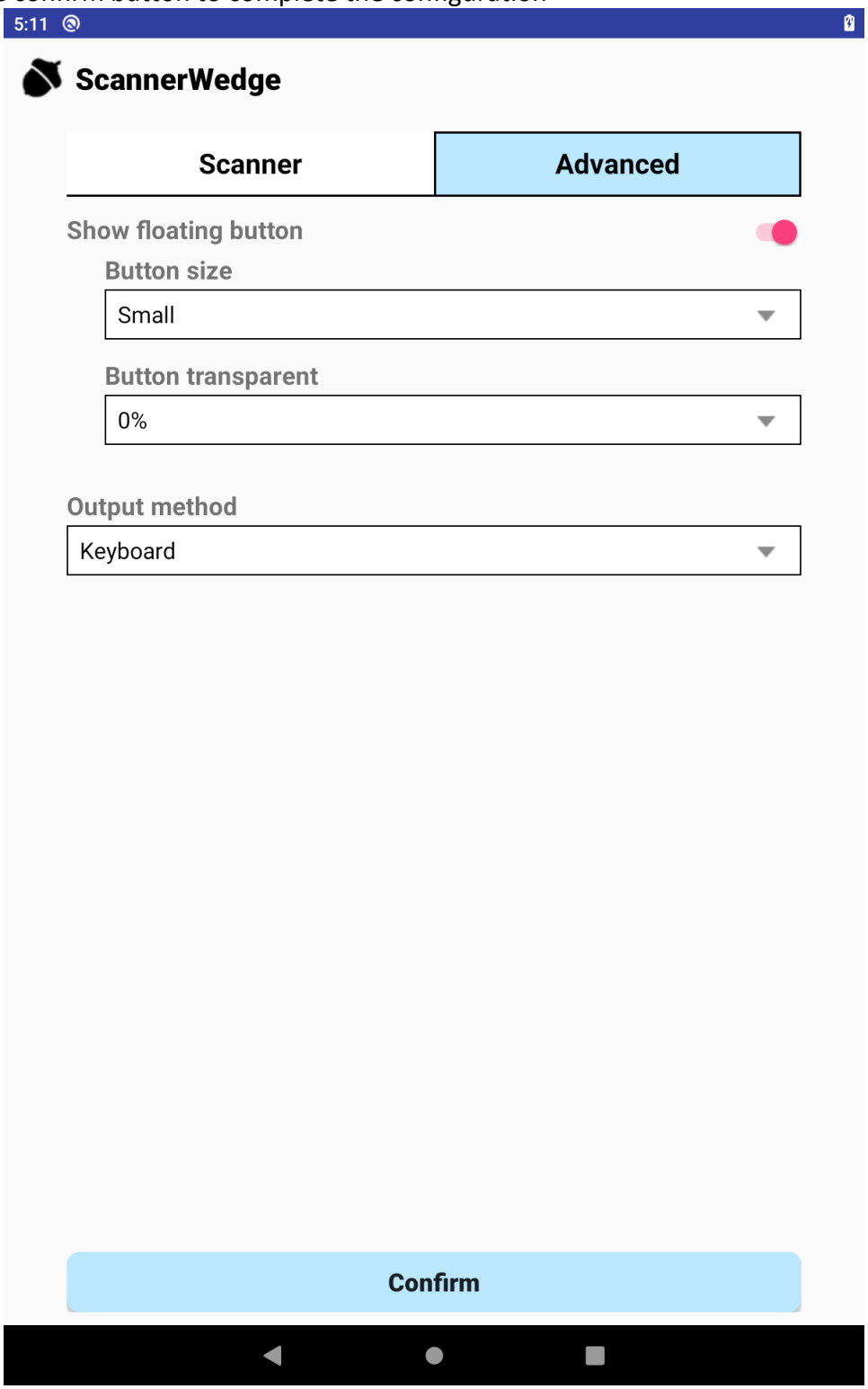
Run ScannerWedge app and make some configuration

1. Scanner name : **Select the correct scanner**
2. Data format : **String (Only option, please ignore)**



3. Show floating button: **Select whether to turn it on according to your own needs. If enabled, you can continue to configure the following parameters**
    1. Button size: **Normal**
    2. Button transparent: **50%**
4. Output method: **Keyboard**

5. Complete configuration(This step is indispensable)
   Click the confirm button to complete the configuration



Trigger Scan

There are three ways to trigger a scan

a. Physical key trigger
   Press the physical key.

b. Broadcast trigger
You can add the following code to the app to send a broadcast and trigger a scan

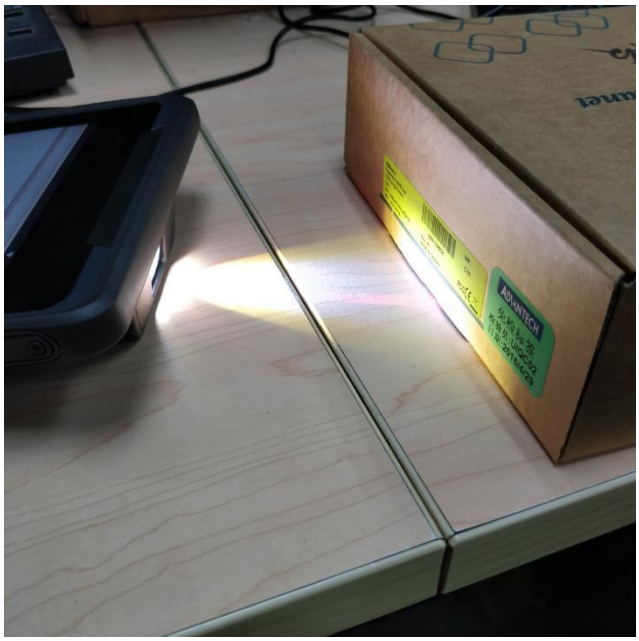
```
1.  public class MainActivity extends AppCompatActivity {
2.      private static final String ACTION_TRIGGER_SCAN = "com.advantech.scannerwedge.TRIGGE
    R_SCAN";
3.
4.      @Override
5.      protected void onCreate(Bundle savedInstanceState) {
6.          super.onCreate(savedInstanceState);
7.          setContentView(R.layout.activity_main);
8.
9.          Button mBtnTrigger = findViewById(R.id.btn_trigger);
10.         mBtnTrigger.setOnClickListener(new View.OnClickListener() {
11.             @Override
12.             public void onClick(View view) {
13.                 Intent intent = new Intent(ACTION_TRIGGER_SCAN);
14.                 intent.setFlags(Intent.FLAG_INCLUDE_STOPPED_PACKAGES);
15.                 sendBroadcast(intent);
16.             }
17.         });
18.     }
19.
```

```
20. }
```

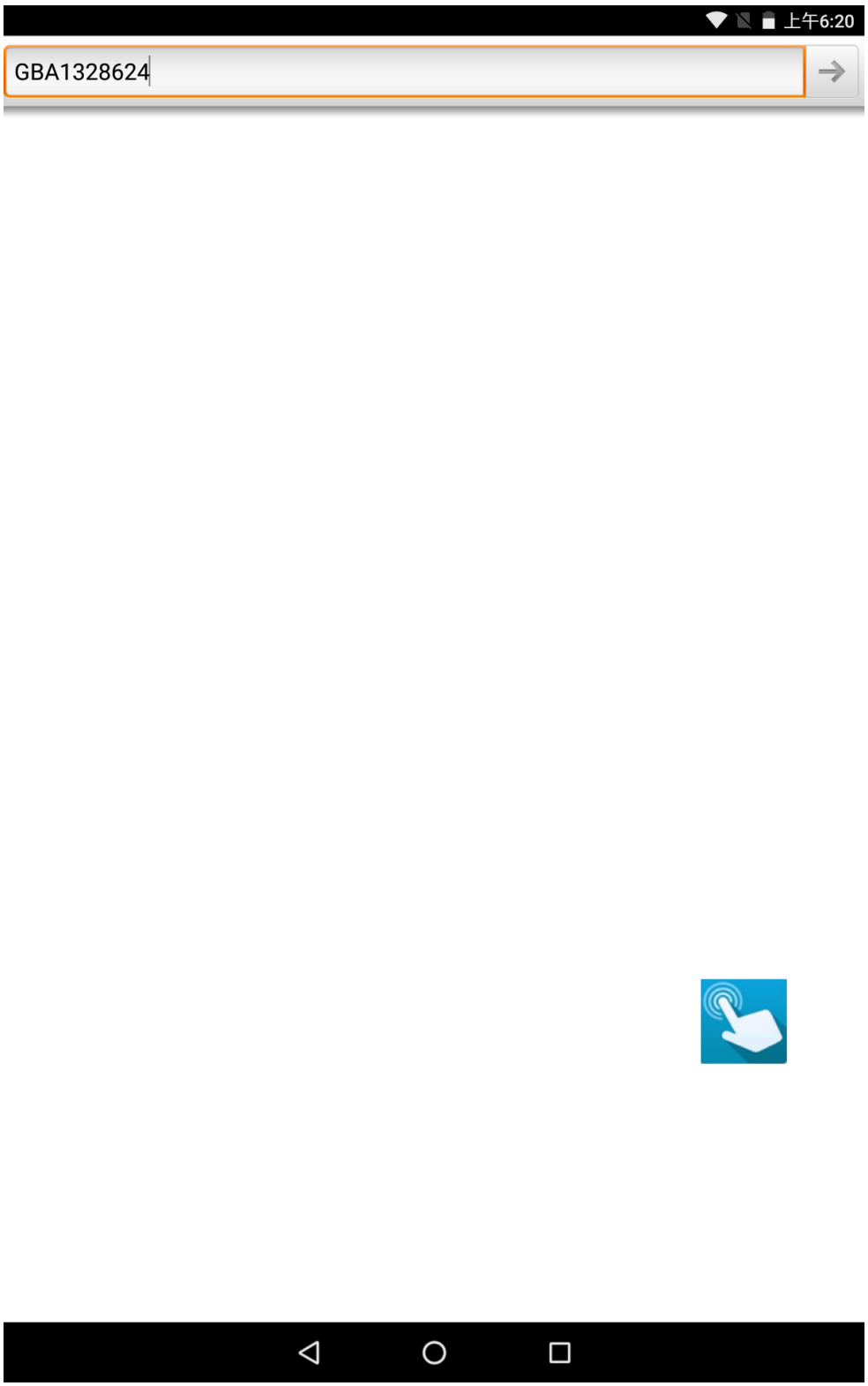
c. Floating button trigger

If you turn on the floating button, you can see the floating button shown below in the UI. Click it to trigger scanning

Barcode reader Startup



Select mouse cursor output location, scan bar code items
ex : Search

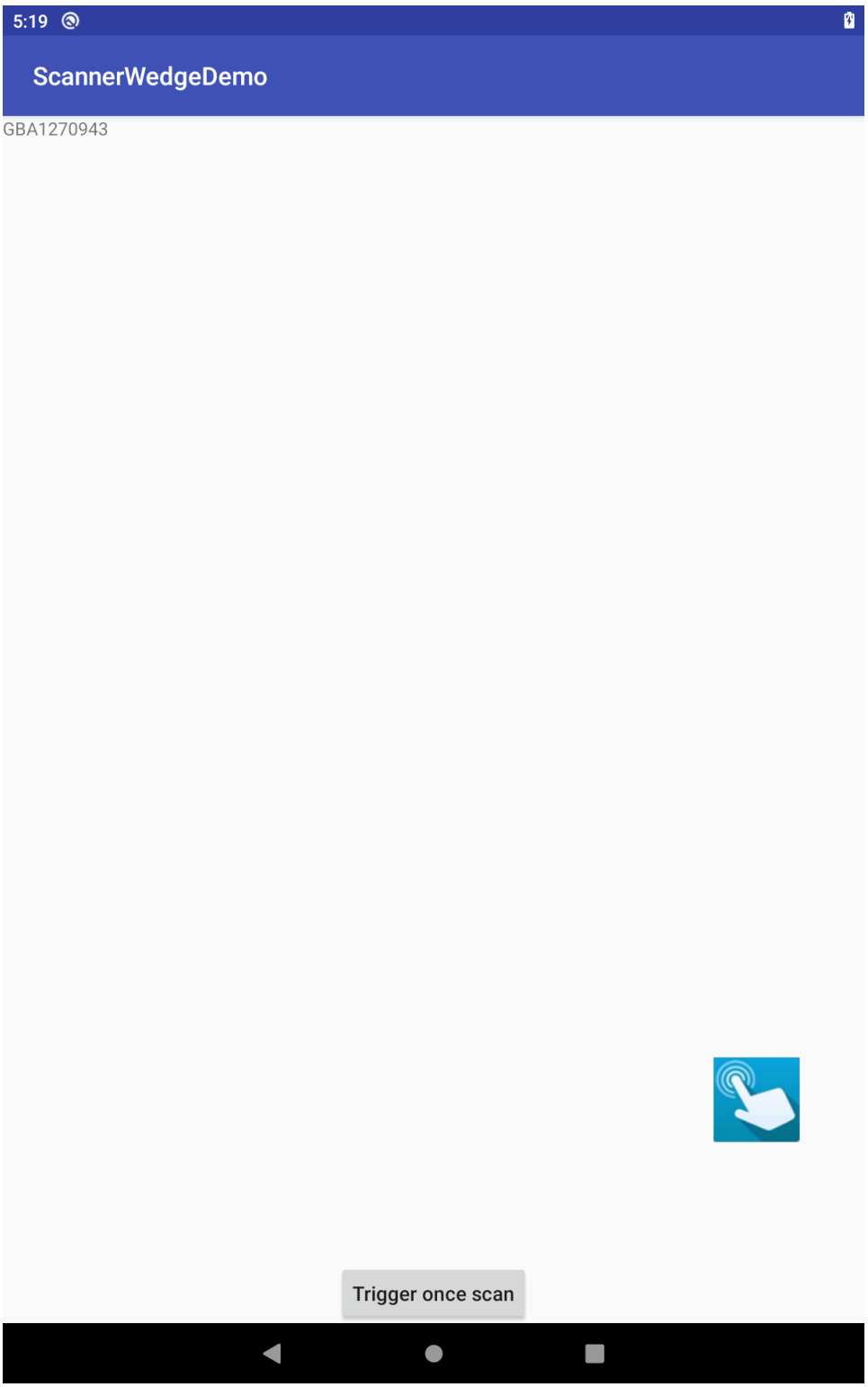


If broadcast mode is selected in “Advance->Output method”. You can add the following code to your app to receive data.

```
1.  public class MainActivity extends AppCompatActivity {
2.      private static final String ACTION_TRANSFER_DATA = "com.advantech.scannerwedge.TRANS
    FER_DATA";
3.
4.      private TextView textView;
5.      BarCodeDataBroadcastReceiver barCodeDataBroadcastReceiver;
6.
7.      @Override
```

```
8.      protected void onCreate(Bundle savedInstanceState) {
9.          super.onCreate(savedInstanceState);
10.         setContentView(R.layout.activity_main);
11.         textView = findViewById(R.id.textview);
12.
13.         IntentFilter filter = new IntentFilter();
14.         filter.addAction(ACTION_TRANSFER_DATA);
15.         barCodeDataBroadcastReceiver = new BarCodeDataBroadcastReceiver();
16.         registerReceiver(barCodeDataBroadcastReceiver, filter);
17.
18.     }
19.
20.     private class BarCodeDataBroadcastReceiver extends BroadcastReceiver {
21.
22.         @Override
23.         public void onReceive(Context context, Intent intent) {
24.             String barcodeData = intent.getStringExtra("barcode_data");
25.             if (barcodeData != null) {
26.                 textView.append(barcodeData + "\n");
27.             }
28.         }
29.     }
30.
31.     @Override
32.     protected void onDestroy() {
33.         unregisterReceiver(barCodeDataBroadcastReceiver);
34.         super.onDestroy();
35.     }
36.
37. }
```

For example, the data received with ScannerWedgeSample app is as follows



Note

ScannerWedge app needs system signature, so please sign the app before installation