



Newland

SCANNING MADE SIMPLE

Newland Android Mobile Computer Barcode Scanning SDK Handbook

Revision History

Version	Description	Date
V1.0	Initial release.	August 13, 2015
V1.1	Modified Chapter 3 and Chapter 4.	June 12, 2017
V1.2	Added Chapter 2 and Appendix.	June 19, 2018

Table of Contents

Chapter 1 Overview	1
Chapter 2 Barcode Scanning APIs	1
Chapter 3 Acquiring Scanned Data	2
1. Output via API.....	2
2. Simulate keystroke.....	2
3. Fill in EditText directly	3
Chapter 4 Programming Examples.....	3
1. Development Environment.....	3
2. Adding the SDK to Your Project	3
3. Sample Codes	3
4. Examples of Calling the APIs	5
Appendix.....	6
Symbology ID Number	6

Chapter 1 Overview

This manual is applicable to NLS-MT90 and NLS-MT65 portable data collectors (hereinafter referred to “**the terminal**”). It provides general barcode scanning features and the APIs you need to program them, including:

1. Output mode

Set the means by which users acquire scanned data.

A. Output via API

Application acquires scanned data by receiving specific system broadcasts.

B. Simulate keystroke

Output scanned data to keyboard buffer to simulate keyboard input and get the data into a textbox that has focus.

C. Fill in EditText directly

Output scanned data directly to EditText box that has focus. The difference between this method and keystroke simulation is that the latter does not rely on support of the platform.

2. Scan mode

A. Level mode: Press the scan key to start a decode session which continues until the key is released.

B. Pulse mode: Press the scan key to start a decode session which continues until the decode session timeout expires.

C. Continuous mode: Press the scan key to continuously read barcodes. To suspend or resume scan operation, simply press the scan trigger.

3. Scan trigger

Set the SCAN key and Left/Right SCAN key on the terminal, and background scan key (i.e. the trigger on the pistol grip installed on the terminal) as scan trigger.

4. Prefix & suffix

Set the prefix/suffix which will be appended to scanned data.

5. Good read indicator

Set the good read indicators such as sound, vibration and LED.

Chapter 2 Barcode Scanning APIs

See *Barcode Scanning API Reference*.

Chapter 3 Acquiring Scanned Data

1. Output via API

Application acquires scanned data by registering a broadcast receiver and listening for specific broadcast intents.

- Broadcast: **nlscan.action.SCANNER_RESULT**
To get barcode data.
- Extra scan result 1 parameter: **SCAN_BARCODE1**
To get the data of barcode 1.
Type: String
- Extra scan result 2 parameter: **SCAN_BARCODE2**
To get the data of barcode 2.
Type: String
- Extra symbology ID number parameter: **SCAN_BARCODE_TYPE**
Type: int (-1 indicates failure to get symbology ID Number)
To get the ID number of the barcode scanned (Refer to the “Symbology ID Number” table in Appendix to get the barcode type).
- Extra scan state parameter: **SCAN_STATE** (value: fail or ok)
To get the status of scan operation: Value = fail, operation failed
Value = ok, operation succeeded
Type: String

```
String sValue1 = intent.getStringExtra("SCAN_BARCODE1"); //barcode 1 (string)
String sValue2= intent.getStringExtra("SCAN_BARCODE2"); //barcode 2 (string)
```

```
byte[] bvalue1=intent.getByteArrayExtra("scan_result_one_bytes"); //barcode 1 (byte array)
byte[] bvalue2=intent.getByteArrayExtra("scan_result_two_bytes"); // barcode 2(byte array)
int barcodeType = intent.getIntExtra("SCAN_BARCODE_TYPE", -1); // -1:unknown
```

```
final String scanStatus=intent.getStringExtra("SCAN_STATE");
if("ok".equals(scanStatus)){
    //Success
}else{
    //Failure, e.g. operation timed out
}
```

2. Simulate keystroke

Get scanned data through an application that has input focus. (Use dispatchKeyEvent method).

3. Fill in EditText directly

Get scanned data through EditText that has focus. (Fill scanned data into textbox directly).

Chapter 4 Programming Examples

1. Development Environment

A. Use an appropriate SDK version that is not higher than the terminal's operating system version. For example, for the NLS-MT65 portable data collector running on Android 4.4.2 operating system, it is recommended to use Android SDK 4.0 to 4.4.2.

B. It is recommended to use Eclipse 3.7.2 or later version when developing apps for the NLS-MT65.

2. Adding the SDK to Your Project

Copy the SDK file nlscan_master_sdk.jar to the libs directory of the project.

3. Sample Codes

Here are some sample codes from the ScanSettingTest program.

1. Acquire ScanManager during initialization

```
private ScanManager mScanMgr;
@Override
protected void onCreate(Bundle savedInstanceState) {
.....
    mScanMgr=ScanManager.getInstance();
    .....
}
```

2. Acquire current output mode with the getScanSettings function

```
Map<String, String> settings=mScanMgr.getScanSettings();
String sOutputMode=settings.get(ScanSettings.Global.OUT_PUT_MODE); //Acquire
current output mode
```

3. Set the output mode with the setScanSettings function

```
int outputMode=-1;

switch (checkedId) {
```

```

//Output via API
case R.id.radio_broadcast:
    outputMode=ScanSettings.Global.VALUE_OUT_PUT_MODE_BROADCAST;
    break;

//Simulate keystroke
case R.id.radio_emulate_key:
    outputMode=ScanSettings.Global.VALUE_OUT_PUT_MODE_EMULATE_KEY;
    break;

//Fill in EditText directly
case R.id.radio_fill:
    outputMode=ScanSettings.Global.VALUE_OUT_PUT_MODE_FILLING;
    break;
}

if(outputMode!=-1)
    mScanMgr.setOutputMode(outputMode);

```

4. Acquire scanned data by receiving system broadcasts (when Output via API is enabled)

```

IntentFilter intFilter=new IntentFilter("nlscan.action.SCANNER_RESULT");
registerReceiver(mResultReceiver, intFilter);

.....

/**
 * Get the scanned data via broadcast intent when Output via API is enabled
 */
private BroadcastReceiver mResultReceiver=new BroadcastReceiver() {

    @Override
    public void onReceive(Context context, Intent intent) {
        String action=intent.getAction();
        if("nlscan.action.SCANNER_RESULT".equals(action)){
            String svalue1 = intent.getStringExtra("SCAN_BARCODE1");
            String svalue2 = intent.getStringExtra("SCAN_BARCODE2");
            byte[] bvalue1=intent.getByteArrayExtra(ScanManager.EXTRA_SCAN_RESULT_ONE_BYTES);
            byte[] bvalue2=intent.getByteArrayExtra(ScanManager.EXTRA_SCAN_RESULT_TWO_BYTES);
            int state = intent.getStringExtra("SCAN_STATE");
            try {
                if (TextUtils.isEmpty(svalue1) && bvalue1 != null)
                    svalue1 = new String(bvalue1, "GBK");
                if (TextUtils.isEmpty(svalue2) && bvalue2 != null)
                    svalue2 = new String(bvalue2, "GBK");
                tv_broadcast_result.setText(svalue1);
            }
        }
    }
}

```

```

        } catch (Exception e) {
            e.printStackTrace();
            tv_broadcast_result.setText("Decode failed.");
        }
    }
}
}
}

```

4. Examples of Calling the APIs

```

ScanManager sm = ScanManager.getInstance() ;
sm.setScanEnable(true); //Enable scanner
sm.startScan(); //Start scanning barcode
sm.stopScan (); //Stop scanning barcode
sm.enableBeep(); //Turn good read beep on
sm.disableBeep(); //Turn good read beep off
sm.setOutputMode(ScanSettings.Global.VALUE_OUTPUT_MODE_BROADCAST); //Enable the
Output via API mode
sm.setOutputMode(ScanSettings.Global.VALUE_OUTPUT_MODE_EMULATE_KEY); //Enable the
Simulate keystroke mode
sm.setOutputMode(ScanSettings.Global.VALUE_OUT_PUT_MODE_FILLING); //Enable the Fill in
EditText directly mode
sm.setScanMode(ScanSettings.Global.VALUE_SCAN_MODE_DOWN_UP); //Enable the Level
scan mode
sm.setScanMode(ScanSettings.Global.VALUE_SCAN_MODE_TIME_OUT); //Enable the Pulse
scan mode
sm.setTriggerEnable(ScanSettings.Global.TRIGGER_MODE_MAIN,
ScanSettings.Global.VALUE_TRIGGER_ON); //Enable the SCAN key on front panel as scan trigger
sm.setTriggerEnable(ScanSettings.Global.TRIGGER_MODE_LEFT,
ScanSettings.Global.VALUE_TRIGGER_ON); //Enable the Left SCAN key as scan trigger
sm.setTriggerEnable(ScanSettings.Global.TRIGGER_MODE_RIGHT,
ScanSettings.Global.VALUE_TRIGGER_ON); //Enable the Right SCAN key as scan trigger

sm.setTriggerEnable(ScanSettings.Global.TRIGGER_MODE_MAIN,
ScanSettings.Global.VALUE_TRIGGER_OFF); //Disable the SCAN key on front panel as scan trigger
sm.setTriggerEnable(ScanSettings.Global.TRIGGER_MODE_LEFT,
ScanSettings.Global.VALUE_TRIGGER_ OFF); //Disable the Left SCAN key as scan trigger
sm.setTriggerEnable(ScanSettings.Global.TRIGGER_MODE_RIGHT,ScanSettings.Global.VALUE
_TRIGGER_ OFF); //Disable the Right SCAN key as scan trigger

sm.setScanTimeout(7000); //Set the decode session timeout to 7 seconds

```



```

sm.setPrefix("41"); //Set the prefix to "A" (0x41)
sm.setSuffix ("41"); //Set the suffix to "A" (0x41)
sm.setScanPromptVibrateEnable(true); //Turn on vibration notification for good read
sm.setScanPromptVibrateEnable(false); //Turn off vibration notification for good read
sm.setAutoNewLineEnable(true); //Add a newline after scanned data
sm.setCodeParam("Code128","Enable","1"); //Enable Code128
sm.setEmulateOutputIntervalTime(50); //Set the delay between emulated keystrokes to 50ms

Map<String,String> replaceMap = new HashMap<String,String>();
replace.put("x61","x62"); //Replace "a" with "b"
sm.setReplaceContent(replaceMap);

sm.setOutputRecoverable(true); //Overwrite output (Precondition: Simulate keystroke mode or Fill in
EditText directly mode is enabled)
sm.setSubResutLimits(new int[]{1,9}); //Output the first through ninth characters of raw data

```

Appendix

Symbology ID Number

ID Number	Symbology
0	ZASETUP
1	SETUP128
2	CODE128
3	UCCEAN128
4	AIM128
5	GS1_128
6	ISBT128
7	EAN8
8	EAN13
9	UPCE
10	UPCA
11	ISBN
12	ISSN
13	CODE39
14	CODE93
15	93I
16	CODABAR
17	ITF

18	ITF6
19	ITF14
20	DPLEITCODE
21	DPIDENTCODE
22	CHNPOST25
23	STANDARD25
23	IATA25
24	MATRIX25
25	INDUSTRIAL25
26	COOP25
27	CODE11
28	MSIPLESSEY
29	PLESSEY
30	RSS14
31	RSSLIMITED
32	RSSEXPANDED
33	TELEPEN
34	CHANNELCODE
35	CODE32
36	CODEZ
37	CODABLOCKF
38	CODABLOCKA
39	CODE49
40	CODE16K
41	HIBC128
42	HIBC39
43	RSSFAMILY
44	TriopticCODE39
45	UPC_E1
256	PDF417
257	MICROPDF
258	QRCODE
259	MICROQR
260	AZTEC
261	DATAMATRIX
262	MAXICODE
263	CSCODE
264	GRIDMATRIX
265	EARMARK
266	VERICODE
267	CCA
268	CCB
269	CCC

270	COMPOSITE
271	HIBCAZT
272	HIBCDM
273	HIBCMICROPDF
274	HIBCQR
512	POSTNET
513	ONECODE
514	RM4SCC
515	PLANET
516	KIX
517	APCUSTOM
518	APREDIRECT
519	APREPLYPAID
520	APROUTING
768	NUMOCRB
769	PASSPORT
770	TD1
2048	PRIVATE
2049	ZZCODE
65535	UNKNOWN



Newland

SCANNING MADE SIMPLE

Newland EMEA HQ
+31 (0) 345 87 00 33
info@newland-id.com
newland-id.com

Need more info? Contact us or one of
our partners at newland-id.com/partners