Newland Android NQuire
API Handbook

Revision History

Version	Description	Date
V1.0.0	Initial release.	Nov.13th, 2019
V1.0.1	Combination with NQ 350&750&1000 description	July 1st, 2020

Table of Contents

About This Manual

This manual is applicable to Newland Android NQuire 350/750/1000 series customer information terminal (hereinafter referred to **“the terminal”**).

Development Environment

All APIs are built based on standard Android broadcast mechanism, so there is no need for additional SDKs.The terminal application development environment is the same as Android application development environment.

Obtain Product Model Number

To get the product model number, use **ro.build.version.incremental**.

Barcode Scanner

Scan Barcode

To activate the terminal to scan barcode, application should send the following broadcast to the system.

- Broadcast: **nlscan.action.SCANNER_TRIG**

  To trigger the scan engine.

- Extra scan timeout parameter: **SCAN_TIMEOUT** (value: int, 1-9; default value: 3; unit: second)

  To set scan timeout, i.e. the maximum time a scan attempt can last.

- Extra scan type parameter: **SCAN_TYPE** (value: 1 or 2; default value: 1)

  To set scan type: Value = 1, read one barcode during a scan attempt

  Value = 2, read two barcodes during a scan attempt (This feature is **NOT** available)

Example 1:

```
Intent intent = new Intent ("nlscan.action.SCANNER_TRIG");
mContext.sendBroadcast(intent);
```

Example 2:

```
Intent intent = new Intent ("nlscan.action.SCANNER_TRIG");
intent.putExtra("SCAN_TIMEOUT", 4);// SCAN_TIMEOUT value: int, 1-9; unit: second.
intent.putExtra("SCAN_TYPE ", 2);// SCAN_TYPE: read two barcodes during a scan attempt.
mContext.sendBroadcast(intent);
```

Note: When a scan and decode session is in progress, sending the broadcast above will stop the ongoing session.

Get Barcode Data

There are three ways to get barcode data:

1. Fill in EditText directly: Output scanned data at the current cursor position in EditText.

2. Simulate keystroke: Output scanned data to keyboard buffer to simulate keyboard input and get the data at the current cursor position in TextBox.

3. 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

Example:

Register broadcast receiver:

```
mFilter= newIntentFilter("nlscan.action.SCANNER_RESULT");
mContext.registerReceiver(mReceiver, mFilter);
```

Unregister broadcast receiver:

```
mContext.unregisterReceiver(mReceiver);
```

Get barcode data:

```
mReceiver= newBroadcastReceiver() {
        @Override
publicvoidonReceive(Context context, Intent intent) {
            final String scanResult_1=intent.getStringExtra("SCAN_BARCODE1");
            final String scanResult_2=intent.getStringExtra("SCAN_BARCODE2");
            final 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
                }
    }
    };
```

Stop Scanning

Use the broadcast **nlscan.action.STOP_SCAN** to stop an ongoing decode session.

Example:

```
Intent stopIntent = new Intent("nlscan.action.STOP_SCAN");
mContext.sendBroadcast(stopIntent);
```

Change the Scanner Settings

Application can set one or more scanner parameters, such as enable/disable scanner, by sending to the system the broadcast **ACTION_BAR_SCANCFG** which can contain up to 3 parameters. Remarked with "*****" is default value.

Parameter	Type	Description (* indicates default)
EXTRA_SCAN_POWER	INT	Value = 0 Disable scanner = 1 Enable scanner***** Note: When scanner is enabled, it will take some time to initialize during which all scan requests will be ignored.
EXTRA_TRIG_MODE	INT	Value = 0 Level mode = 1 Continuous mode = 2 Pulse mode*****
EXTRA_SCAN_MODE	INT	Value = 1 Fill in EditText directly*****

		= 2 Simulate keystroke = 3 Output via API
EXTRA_SCAN_AUTOENT	INT	Value = 0 Do not add a line feed***** = 1 Add a line feed
EXTRA_SCAN_NOTY_SND	INT	Value = 0 Sound notification off = 1 Sound notification on*****
SCAN_TIMEOUT	LONG	Set decode session timeout (millisecond) Value = 0-9000; default: 3000*****
SCAN_INTERVAL	LONG	Set timeout between decode sessions (millisecond) Value >= 50; default: 50*****
NON_REPEAT_TIMEOUT	LONG	Set reread delay (millisecond) Value = 0 Reread same barcode with no delay***** > 0 Do not allow to reread same barcode before the delay expires
SCAN_PREFIX_ENABLE	INT	Value = 0 Disable prefix = 1 Enable prefix*****
SCAN_SUFFIX_ENABLE	INT	Value = 0 Disable suffix = 1 Enable suffix*****
SCAN_PREFIX	STRING	Set prefix Value = Hexadecimal value of prefix character; default: null***** e.g. 0x61 should be entered as 61.
SCAN_SUFFIX	STRING	Set suffix Value = Hexadecimal value of suffix character; default: null***** e.g. 0x61 should be entered as 61.
SCAN_ENCODE	INT	Character encoding Value = 1 UTF-8 = 2 GBK***** = 3 ISO-8859-1
OUTPUT_RECOVERABLE	BOOLEAN	Value = true Enable overwrite output = false Disable overwrite output*****

Example 1: Disable scanner

```
Intent intent = new Intent ("ACTION_BAR_SCANCFG");
intent.putExtra("EXTRA_SCAN_POWER", 0);
mContext.sendBroadcast(intent);
```

Example 2: Output via API, add a line feed

```
Intent intent = new Intent ("ACTION_BAR_SCANCFG");
intent.putExtra("EXTRA_SCAN_MODE", 3);
intent.putExtra("EXTRA_SCAN_AUTOENT", 1);
mContext.sendBroadcast(intent);
```

Other APIs

Press the Home Key to Switch to Desktop

To enable/disable the feature of switching to desktop by pressing the Home key, application should send to the system the broadcast nlscan.action.HOMEKEY_SWITCH_STATE with the value of Extra parameter ENABLE set to be true/false.

Example: Disable the feature of switching to desktop by pressing the Home key

```
Intent intent = new Intent("nlscan.action.HOMEKEY_SWITCH_STATE");
intent.putExtra("ENABLE", false);
context.sendBroadcast(intent);
```

Add Physical Button Control Interface, Enable or Disable the Physical Button. (Not for NQuire 350)

When you use the adb shell or ssh connection tool to enter the shell mode, you can directly enable or disable the physical button node as follows:

echo 2 >/sys/devices/virtual/misc/key_ctl/key_ctl Forbidden

echo 1 >/sys/devices/virtual/misc/key_ctl/key_ctl Permissible

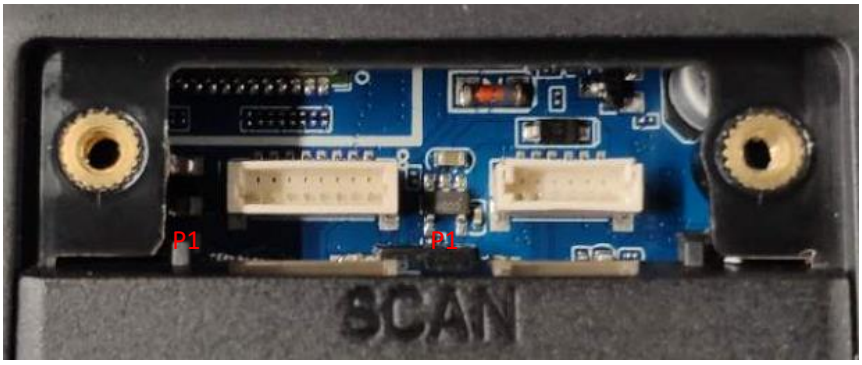
Example

```
Intent intent = new Intent();
intent.putExtra("isShow", value);int value; //value=1 enable the button，value=2 disable the button.
intent.setAction("physical.button.manager"); //send the broadcast.
sendBroadcast(intent);
```

API Interface Description

(Mainly related to set GPIO port, enable or disable physical button, obtain IR value, and set the working mode of the scanner)

(NQuire1000)



(NQuire750)

PIN1	GPIO_IN	GPIO0_31		PIN1	GND
PIN2	GPIO_IN	GPIO0_30		PIN2	RS232-RX
PIN3	GPIO_OUT	GPIO0_29		PIN3	RS232-TX
PIN4	GPIO_OUT	GPIO0_15		PIN4	GND
PIN5	GND_IN			PIN5	VCC_OUT_5V
PIN6	GND_IN			PIN6	VCC_OUT_5V
PIN7	NC			备注：UART_EN=GPIO1_01	
PIN8	VCC_IN				

Control method

a.Import jar

First quote Nquire_interface_api.jar and then import the NQManager class into your code. Namely, **import com.android.nq1000.NQManager**.

b. Instructions for the NQManager class

API: setdoorThreshold(String value);
API Description: Set the GPIO port status.
Parameter Description:
The current parameters that can be set are:
150 (GPIO0_15 pull high level)
151 (GPIO0_15 pull low level)
290 (GPIO0_29 pull high level)
291 (GPIO0_29 pull low level)
1011 (uart serial port enable)
1010 (uart serial port disable)
Return value type: boolean
Return value description:
True: set successfully

00: GPIO0_30 input high level, GPIO0_31 input high level
01: GPIO0_30 input high level, GPIO0_31 input low level
10: GPIO0_30 input low level, GPIO0_31 input high level
11: GPIO0_30 input low level, GPIO0_31 input low level

False: setting failed.

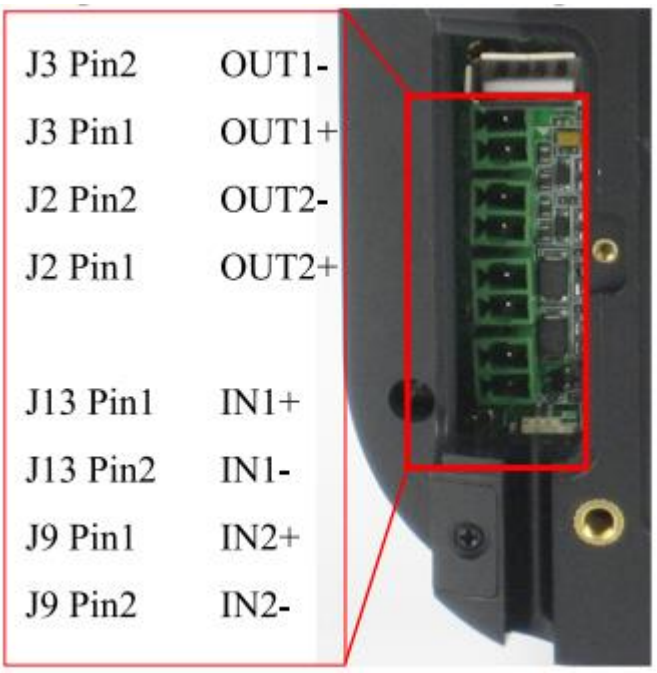
API: getdoorData();
API Description: Read status information for GPIO0_30 and GPIO0_31.
The return value is a 2-digit number, where GPIO0_30 is the tens place and GPIO0_31 is the one place.
Parameter description: null
Return value type: String
Return value description:
Possible return values are:

API: getIrData();
API Description: Get the IR sensor value.
Parameter description: null
Return value type: String
Return value description: IR value

API: setThreshold(string) ;
API Description: Set the working mode of scan head.
Parameter Description:
The current parameters that can be set are:
Barscanonn: initialize the power to the scan head.
Barscanpoweronn: power supply to the scan head power pin.
Barscanpoweroffn: power off to the scan head power pin.

Barscantrgonn: power supply to the scan head trig pin.
Barscantrgoffn: power off to the scan head trig pin.
Return value type: boolean
Return value description:
True: set successfully
False: setting failed

NQ350 GPIO port pinout :



(NQuire350)

API: setdoorThreshold(String value);
API Description: Set the GPIO port status.
Parameter description：
The current parameters that can be set are:
160（GPIO0_16 pull low level）
161（GPIO0_16 pull high level）
290（GPIO0_29 pull low level）
291（GPIO0_29 pull high level）
1011（uart serial port enable）
1010（uart serial port disable）

2170（gpio2_17 pull low level）
2171（gpio2_17 pull high level）
2230（gpio2_30 pull low level）
2231（gpio2_31 pull high level）
The above mentioned gpio pull high/low level are paremeter under output mode of the CPU gpio port. The specific level on the external hardware interface depends on whether there is a reverse on hardware interface. The corresponding relationship of the gpio port is : A--0 B—8 C--16 D—24. For example, gpio3_21, namely gpio3_c5. For details, please refer to the gpio comparison table

Gate1	GPIO1_A4	GPIO1_04
Gate2	GPIO1_A5	GPIO1_05
Gate3	GPIO2_C3	GPIO1_19
Gate4	GPIO1_A6	GPIO1_06

Return value type：boolean
Return value description:
true：set successfully
false：setting failed
API：getdoorData()；
API Description：Read status information for GPIO1_04 and GPIO1_05.
The return value is a 2-digit number, where GPIO1_04 is the tens place and GPIO1_05 is the one place.

Parameter description：null
Return value type：String
Return value description:：
Possible return values are：
Ten place/one place
00：GPIO1_04 input low level，GPIO1_05 input low level
01：GPIO1_04 input low level，GPIO1_05 input high level
10：GPIO1_04 input high level，GPIO1_05 input low level
11：GPIO1_04 input high level，GPIO1_05 input high level
The above mentioned gpio input high/low level are refer to the parameter at the CPU gpio port. The specific level on the external hardware interface depends on whether there is a reverse on the hardware interface.
API：ifKeysEnabled()；
API Description： Judge whether the physical button is on enable status.
Parameter description：null
Return value type：boolean
Return value description： Ture for physical button enabled, failed for setting failed.
API：enableKeys(boolean enable)；
API Description： Set physical buttons to be valid or invalid
Note: When the physical button is set to be invalid, the Volume+, Volume-, Back keys on systemUI will also be

blocked

Parameter Description: true for physical button is valid, false for invalid
Return value type：boolean
Return value description:
true：true for setting successfully
false：false for setting failed.

Enable and Disable the USB Port with Api

Use the broadcast **private String USB_DISABLE = "com.usb.disable"** to disable the USB .

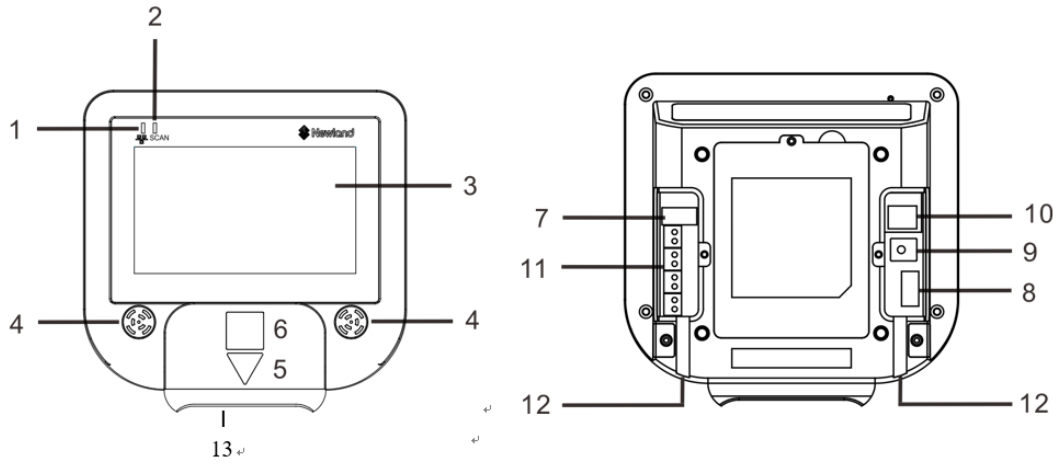
When sending the message, it should take the value for “isOpen”. The value is 0 or 1.

```
Intent intent = new Intent();
If(isChecked){
intent.putExtra("isOpen", 0);//The value is 0, usb is disabled.
}else {
intent.putExtra("isOpen", 1);//The vaule is 1,usb is enable.
}
intent.setAction(USB_DISABLE); //send the broadcast
SendBroadcast(intent);
```

The state of the switch is judged via following: int isOpen =Settings.System.getInt(getContentResolver(), "isOpen", -1);
When “isOpen” = 0, button state is on – disabled status.
When “isOpen” = 1, the button state is off – enable status.

PS： For the NQ350, only the USB interface marked with "7" in the below figure is valid.

Product Outline



1	Network LED	**2**	Good Read LED
3	LCD (Touch) Display	**4**	Speaker
5	“Where to Scan Barcode” Arrow	**6**	RFID Antenna
7	USB Host/Slave Port	**8**	USB Host Port
9	Power Jack	**10**	Ethernet Port
11	GPIO Connectors	**12**	Cable Trough
13	Barcode Scanner		

Hide the Navigation Bar and Status Bar

Open and hide the Navigation bar:

```
Intent intent = new Intent();
intent.putExtra("isOpen", value); //1 is to display navbar, 0 is to hide the navbar.
intent.setAction("com.android.navibar"); //Send the broadcast.
sendBroadcast(intent);
```

Open and hide the the drop-down status bar:

```
Intent intent = new Intent();
intent.putExtra("isHide", value); //1 to hide the status bar, 0 is to display the status bar.
intent.setAction("com.hide.statusbar"); //send the broadcast.
sendBroadcast(intent);
```

Start up the App

```
Intent intent= new Intent();
intent.putExtra("pkg", "com.newland.quicksetting"); //strting1 is the package name.
intent.putExtra("cls", "com.newland.quicksetting.MainActivity");// string2 is the fully-qualified class name.
intent.setAction("start.up.application"); //send the broadcast.
sendBroadcast(intent);
```

The following methods can be used to set up the startup App.

Enable or Disable to Pop Up the Soft Keyboard （Not for NQuire 350 and 750）

The following methods can be used to Enable or disable to pop up the soft keyboard.

```
Intent intent = new Intent();
intent.putExtra("isHide", value); //1 to disable popup, 0 to enable popup soft keyboard.
intent.setAction("com.softinput.manager "); //Send the broadcast
sendBroadcast(intent);
```

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