**新大陆Android无线数据终端软件开发指南**

**V1.7**

**版本记录**

|  |  |  |
| --- | --- | --- |
| 版本号 | 版本描述 | 发布日期 |
| V1.0 | 初始版本。 | 2016-10-11 |
| V1.0.1 | 扫描超时默认改成3秒 | 2016-11-14 |
| V1.0.2 | 获取扫描结果项中添加码制类型 | 2016-11-24 |
| V1.0.3 | 删除PSAM | 2016-11-30 |
| V1.0.4 | 增加停止扫码的接口 | 2017-03-31 |
| V1.0.5 | 删除备用按键章节 | 2017-08-02 |
| V1.0.6 | 恢复选用按键章节，并修改其内容 | 2017-08-03 |
| V1.1 | 扩展广播设置扫描参数的接口 | 2018-6-13 |
| V1.2 | 增加码制类型参照表 | 2018-6-19 |
| V1.3 | 增加码制数据设置接口 | 2018-7-14 |
| V1.5 | 增加广播输出参数可配置 | 2018-12-11 |
| V1.6 | 增加软件键事件输出 | 2019-08-27 |
| V1.7 | 增加广播设置windows-1521编码格式 | 2019-12-04 |

[一、 开发环境 3](#_Toc26369413)

[二、 获取设备产品识别属性 3](#_Toc26369414)

[三、 扫描服务 3](#_Toc26369415)

[1) 开启扫描 3](#_Toc26369416)

[2) 获取扫描结果 3](#_Toc26369417)

[3) 停止扫描 4](#_Toc26369418)

[4) 修改扫描设置默认值 5](#_Toc26369419)

[四、备用按键 7](#_Toc26369420)

[五、其他接口 8](#_Toc26369421)

[1) 状态栏下拉 8](#_Toc26369422)

[1) 按Home键返回桌面 8](#_Toc26369423)

[2) 设置系统时间 8](#_Toc26369424)

[附件 9](#_Toc26369425)

[码制类型对照表 9](#_Toc26369426)

本手册适用于新大陆**MT65,MT90（Android）**等无线数据终端。

1. 开发环境

设备的二次开发接口采用了android的标准广播方式通信，不需要额外的SDK，应用程序开发环境与原生Android系统的应用开发环境兼容。

1. 获取设备产品识别属性

可通过系统属性：android.os.Build.MODEL获取设备产品型号，应用程序可根据该变量自适应不同产商设备，例如MT65、MT90。

1. 扫描服务
2. 开启扫描

注：当使用物理扫描键来触发扫描时，系统底层会自行监听扫描按键并启动扫描，应用程序无需发送此广播，直接跳过这步操作。

发送一个广播以启动扫描。

* 广播名Action：**nlscan.action.SCANNER\_TRIG**
* Extra扫描超时参数：**SCAN\_TIMEOUT**

（单位为秒，值为int类型，且不超过9秒，默认3秒）

* Extra 扫码类型参数：**SCAN\_TYPE**

(单码、双码类型，值为1：单码，值为2：双码，默认单码；)

注：暂不支持双码

举例1：

Intent intent = new Intent ("nlscan.action.SCANNER\_TRIG");

mContext.sendBroadcast(intent);

举例2：

Intent intent = new Intent ("nlscan.action.SCANNER\_TRIG");

intent.putExtra("SCAN\_TIMEOUT", 4);//单位为秒，值为int类型，且不超过9秒

intent.putExtra("SCAN\_TYPE ", 2);//扫码类型：双码

mContext.sendBroadcast(intent);

说明：如果当前已经启动了扫描，此时如果系统又收到该广播，则表示取消当前正在进行的扫描事务。

注：正常使用按键来触发扫描时，系统底层会启动扫描，应用程序无需再监听扫描按键再开启扫描。

1. 获取扫描结果

* 直接填充模式：当设置该输出模式时，结果会自动填充到焦点状态的输入框。
* 模拟键输出模式：当设置该输出模式时，结果会转换按键键值输出到焦点状态的界面。
* API输出模式：当设置该输出模式时，需要注册广播监听来获取结果

广播名Action：**nlscan.action.SCANNER\_RESULT**

Extra扫描结果1参数：**SCAN\_BARCODE1**

数据类型为：String

Extra扫描结果2参数：**SCAN\_BARCODE2**

数据类型为：String

Extra码制类型：**SCAN\_BARCODE\_TYPE**

数据类型为：int（-1：表示未知类型）

Extra扫码状态参数：**SCAN\_STATE**

(该参数有两个值：fail或ok.)

数据类型为：String

举例1：

注册广播：

mFilter= newIntentFilter("nlscan.action.SCANNER\_RESULT");

mContext.registerReceiver(mReceiver, mFilter);

注销获取扫描结果的广播

mContext.unregisterReceiver(mReceiver);

获取结果值：

mReceiver= newBroadcastReceiver() {

@Override

publicvoidonReceive(Context context, Intent intent) {

finalString scanResult\_1=intent.getStringExtra("SCAN\_BARCODE1");

finalString scanResult\_2=intent.getStringExtra("SCAN\_BARCODE2");

finalintbarcodeType = intent.getIntExtra("SCAN\_BARCODE\_TYPE", -1); // -1:unknown

finalString scanStatus=intent.getStringExtra("SCAN\_STATE");

if("ok".equals(scanStatus)){

//成功

}else{

//失败如超时等

}

}

};

1. 停止扫描

注：当使用物理扫描键来触发扫描时，系统底层会自行监听扫描按键并按照设定好的扫描模式来停止扫描，应用程序无需发送此广播，直接跳过这步操作。即使用物理扫描键启动读码，只需通过广播获取扫描结果，无需在应用程序中处理启动、停止扫描的操作。

发送一个广播，可以停止正在进行的扫描操作。

* 广播名Action：**nlscan.action.STOP\_SCAN**

例：

Intent stopIntent = new Intent(“nlscan.action.STOP\_SCAN”);

mContext.sendBroadcast(stopIntent);

1. 修改扫描设置默认值

* **修改通用配置**

应用程序可以通过发送广播以配置系统扫描头使能及数据输出模式。其广播名为：ACTION\_BAR\_SCANCFG，该广播最多可带3个参数，应用程序可根据需要一次修改以下一个或多个参数，参数定义如下，标\*为默认值：

|  |  |  |
| --- | --- | --- |
| 参数名 | 参数类型 | 备注 |
| EXTRA\_SCAN\_POWER |  | 值 = 0 表示禁用扫描功能  = 1 表示打开扫描功能\*  说明：当扫描头刚打开的时候需要初始化扫描头，需要一定时间，此时将忽略相关扫描请求 |
| EXTRA\_TRIG\_MODE | INT | 值 = 0 配置扫描头为普通触发模式  = 1 配置扫描头为连续扫描模式  = 2 配置扫描头为超时扫描模式\* |
| EXTRA\_SCAN\_MODE | INT | 值 = 1 ：直接填充模式\*  = 2 ：虚拟按键模式  = 3 ：广播输出模式 |
| EXTRA\_SCAN\_AUTOENT | INT | 值 = 0 关闭自动换行\*  = 1 允许自动换行 |
| EXTRA\_SCAN\_NOTY\_SND | INT | 值 = 0 关闭声音提示  = 1 打开声音提示\* |
| EXTRA\_SCAN\_NOTY\_VIB | INT | 值 = 0 关闭振动提示\*  = 1 打开振动提示 |
| EXTRA\_SCAN\_NOTY\_LED | INT | 值 = 0 关闭指示灯提示  = 1 打开指示灯提示\* |
| SCAN\_TIMEOUT | LONG | 一次扫描超时时间（毫秒）如3秒：3000L |
| SCAN\_INTERVAL | LONG | 连续扫描时，两次扫描的间隔时间（毫秒），值>=50，默认：50（毫秒） |
| TRIGGER\_MODE\_MAIN | INT | 值= 0主扫描键禁止扫描  = 1 主扫描键允许扫描\* |
| TRIGGER\_MODE\_LEFT | INT | 值= 0左侧扫描键禁止扫描  = 1 左侧扫描键允许扫描\* |
| TRIGGER\_MODE\_RIGHT | INT | 值= 0右侧扫描键禁止扫描  = 1 右侧扫描键允许扫描\* |
| TRIGGER\_MODE\_BLACK | INT | 值= 0背面扫描键禁止扫描  = 1 背面扫描键允许扫描\*  （前提支持背扫描键功能） |
| NON\_REPEAT\_TIMEOUT | LONG | 不发送重复条码数据的时间间隔（毫秒），如2秒：2000 |
| SCAN\_PREFIX\_ENABLE | INT | 值= 0前缀禁用  = 1前缀使能\* |
| SCAN\_SUFFIX\_ENABLE | INT | 值= 0后缀禁用  = 1后缀使能\* |
| SCAN\_PREFIX | STRING | 前缀值，16进制表示，如0x61，则传入：”61”  默认：空 |
| SCAN\_SUFFIX | STRING | 后缀值，16进制表示，如0x61，则传入：”61”  默认：空 |
| SCAN\_ENCODE | INT | 编码格式：  值  = 1 UTF-8  = 2 GBK\*  = 3 ISO\_8859\_1  =4 自动  =5 其他  需同时传入  SCAN\_OTHER\_ENCODE的值  =6 windows-1251 |
| OUTPUT\_RECOVERABLE | BOOLEAN | 值= true 使用覆盖输出  = false 禁止覆盖输出\* |
| EXTRA\_OUTPUT\_EDITOR\_ACTION\_ENABLE | INT | 值= 1 输出软件键事件  = 0不输出软件键事件\* |
| EXTRA\_OUTPUT\_EDITOR\_ACTION | INT | 值  =0 IME\_ACTION\_UNSPECIFIED  =1 IME\_ACTION\_NONE,  =2 IME\_ACTION\_GO,  =3 IME\_ACTION\_SEARCH,  =4 IME\_ACTION\_SEND,  =5 IME\_ACTION\_NEXT,  =6 IME\_ACTION\_DONE, \*  =7 IME\_ACTION\_PREVIOUS, |
| BROADCAST\_OUTPUT\_ACTION | STRING | 广播输出参数配置： Action参数 |
| BROADCAST\_OUTPUT\_EXTRA\_KEY\_RESULT\_1 | STRING | 广播输出参数配置： 条码结果1的参数 |
| BROADCAST\_OUTPUT\_EXTRA\_KEY\_RESULT\_2 | STRING | 广播输出参数配置： 条码结果2的参数 |
| BROADCAST\_OUTPUT\_EXTRA\_\_KEY\_BARCODE\_TYPE | STRING | 广播输出参数配置： 码制类型的参数 |

举例1：禁用扫描功能

Intent intent = new Intent ("ACTION\_BAR\_SCANCFG");

intent.putExtra("EXTRA\_SCAN\_POWER", 0);

mContext.sendBroadcast(intent);

举例2：设置扫描为“广播输出模式”，同时输出“自动换行”

Intent intent = new Intent ("ACTION\_BAR\_SCANCFG");

intent.putExtra("EXTRA\_SCAN\_MODE", 3);

intent.putExtra("EXTRA\_SCAN\_AUTOENT", 1);

mContext.sendBroadcast(intent);

* **修改码制配置**

应用程序可以通过发送广播以配置扫描头码制与相关参数使能（参数值详情随不同的识读头而不同，参考识读头的参数值手册）

其广播名为：ACTION\_BARCODE\_CFG

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名 |  | 参数类型 | 备注 |
| CODE\_ID |  | STRING | 码制ID |
| PROPERTY |  | STRING | 码制属性ID |
| VALUE |  | STRING | 码制属性的设置值 |

举例1：打开EAN-8码制的”发送校验”功能

Intent intent = new Intent ("ACTION\_BARCODE\_CFG");

intent.putExtra("CODE\_ID", “EAN8”);

intent.putExtra("PROPERTY", “TrsmtChkChar”);

intent.putExtra("VALUE", “1”);//1 开, 0 关

mContext.sendBroadcast(intent);

四、备用按键

* + MT65型号备用按键：F1、F2、F3、F4
  + MT90型号备用按键：F6

应用程序可以根据用户需要，自行定义备用按键的功能。

举例：处理备用按键按下事件

publicbooleanonKeyDown(intkeyCode, KeyEvent event) {

switch (keyCode)

{

case KeyEvent.KEYCODE\_F6:

showInfo("F6键按下\n");

break;

}

return super.onKeyDown(keyCode,event);

}

举例：处理备用按键弹起事件

publicbooleanonKeyUp(intkeyCode, KeyEvent event) {

switch (keyCode)

{

case KeyEvent.KEYCODE\_F6:

showInfo("F6键放开\n");

break;

}

returnsuper.onKeyDown(keyCode, event);

}

五、其他接口

1. 状态栏下拉

设置状态栏是否可以下拉的api接口，应用层通过向系统发送广播，其中广播的内容为：ENABLE为true时，状态栏可以下拉，ENABLE为false时，状态栏禁止下拉。

举例：状态栏禁止下拉

Intent intent = new Intent("nlscan.action.STATUSBAR\_SWITCH\_STATE");

intent.putExtra("ENABLE", false);

context.sendBroadcast(intent);

1. 按Home键返回桌面

设置HOME键短按是否可以回到桌面，应用层通过向系统发送广播，其中广播的extra内容为：ENABLE为true时，按HOME键可以回到桌面，ENABLE为false时，按HOME键禁止回到桌面。

举例：按HOME键禁止回到桌面

Intent intent = new Intent("nlscan.action.HOMEKEY\_SWITCH\_STATE");

intent.putExtra("ENABLE", false);

context.sendBroadcast(intent);

1. 设置系统时间

根据时间的毫秒数来设置系统时间，应用层通过向系统发送广播，其中广播的extra的参数名为TIME\_MS，参数类型为string，表示系统时间毫秒数。

举例：

public long getTimeMillis(){  
 Calendar c = Calendar.getInstance();  
 c.set(2016, 0, 1, 0,0,0);  
 return c.getTimeInMillis();  
 }

 Intent it = new Intent("nlscan.action.SET\_TIME");

 long mills = getTimeMillis();

 it.putExtra("TIME\_MS", String.valueOf(mills));

 mContext.sendBroadcast(it);

**附件**

码制类型对照表

|  |  |
| --- | --- |
| **Code** | **name** |
| 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 |