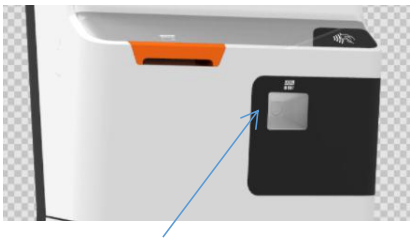
扫码器使用指南

商米扫码+键盘组件用于解决扫码枪和输入法之间互相干扰的问题。
以下分别介绍 USB 扫码器与串口扫码器：

USB 扫码器



串口扫码器(尺寸小很多)

一、USB 扫码器：

USB 扫码器相当于 USBkeyboard 键盘，**只支持数据接收**。**接收方式**有如下 2 种（2 选 1，切换接收方式需要进行设置，出厂**默认为 KeyEvent**）：
方式 1、KeyEvent：用 dispacthKeyEvent 即可。
方式 2、广播：此模式下，数据不会像键盘模式那样输出到 APP 页面中的输入框内；必须用如下方式**切换接收模式**和**接收广播数据**。

1、切换接收模式：

法 1：在“设置”->“扫码与键盘”中修改为“不输出”+“广播输出”。
法 2(建议)：
action:com.sunmi.scanner.ACTION_BAR_DEVICES_SETTING
<p>字段说明：

Key（*必传字段）	解释	字段类型
*name	设备名	String(可通过枚举 UsbDevice 获得)
*pid	扫码器 pid	Integer(同上)
*vid	扫码器 vid	Integer(同上)
*type	数据接收方式	Integer(见下面 type 类型说明)
toast	是否显示调试 Toast	Boolen(默认 false)

<p> name/pid/vid 列表：

name	pid	vid
Synmbol Bar Code Scanner	0x1200	0x05E0
Point of Sale Fixed Barcode Scanner	0x2514	0x05F9
SM-S100W USB HID Keyboard	0x0022	0x324F
SM-S100W USB HID Keyboard	0x00C1	0x324F

<p> type 类型说明：
0-->键盘

1-->扫码枪，直接到 UI（KeyEvent）
2-->扫码枪，不直接到 UI（广播模式）
3-->扫码枪，加速模式（数据内容一次性填充到输入框，需要 1.0.18）

<p>示例（设置某设备为广播输出）：

```
Intent intent = new Intent();
intent.setAction("com.sunmi.scanner.ACTION_BAR_DEVICES_SETTING");
intent.putExtra("name","Point of Sale Fixed Barcode Scanner.");
intent.putExtra("pid",9492);
intent.putExtra("vid",1529);
intent.putExtra("type",2);      //1 KeyEvent 输出   2 广播输出
intent.putExtra("toast",true);
context.sendBroadcast(intent);
```

2、通过广播接收扫码内容

监听广播："com.sunmi.scanner.ACTION_DATA_CODE_RECEIVED"
示例：

```
private static fifinal String ACTION_DATA_CODE_RECEIVED =
"com.sunmi.scanner.ACTION_DATA_CODE_RECEIVED";
private static fifinal String DATA = "data";
private BroadcastReceiver receiver = new BroadcastReceiver()
{
@Override
public void onReceive(Context context, Intent intent)
{
String code = intent.getStringExtra(DATA);
if (code != null && !code.isEmpty())
{
mCode.setText(code);
}
}
};
private void registerReceiver()
{
IntentFilter fifilter = new IntentFilter();
fifilter.addAction(ACTION_DATA_CODE_RECEIVED);
registerReceiver(receiver, fifilter);
}
```

二、串口扫码器：

串口扫码台适用于扫屏幕码，例如手机付款码、电子会员码等。**同时支持 KeyEvent 和广播输出，不需要切换或设置**。
方式一、KeyEvent： 与 USB 扫码器相同，用 dispacthKeyEvent 即可。
方式二、广播： 与 USB 扫码器相同，用 BroadcastReceiver 即可。

此外，串口扫码器可以**通过广播发送指令**，实现扫码器的控制（例如控制扫码器开启和关闭）：
串口扫码器命令文档
串口扫码器 SourcecodeDemo

广播发送指令的方式

action：com.sunmi.scanner.Setting_cmd
cmd byte[]：cmd_data:命令+两位校验位(校验和计算)
Demo 如下：

```
/*
**发送串口命令
*/
public void onSendSerialCmd(View view) {
try {
String s = “NLS0302010;”;//串口命令，例如：NLS0302010;
byte[] bytes = s.getBytes();
byte[] cmd = new byte[bytes.length + 2];
System.arraycopy(bytes, 0, cmd, 0, bytes.length);
lrcCheckSum(cmd);
// send cmd
Intent intent = new Intent("com.sunmi.scanner.Setting_cmd");
intent.putExtra("cmd_data", cmd);
sendBroadcast(intent);
} catch (Exception e) {
e.printStackTrace();
}
}
private void lrcCheckSum(byte[] content) {
int len = content.length;
int crc = 0;
for (int l = 0; l < len - 2; l++) {
crc += content[l] & 0xFF;
}
crc = ~crc + 1;
content[len - 2] = (byte) ((crc >> 8) & 0xFF);
```

```
content[len - 1] = (byte) (crc & 0xFF);
}
```

常用命令：

1、常见的自动扫码的场景，可设置为“感应模式”（默认为此模式）：

- “@SCNMOD2”设置为感应模式。此模式下，扫码器自动扫码。
- “@ORTSET$”设置等待时间，$为时间 ms，建议值 1000。
- “@RRDDUR$”设置同码间隔，$为时间 ms，建议值 800~1000。
- “@SENIST$”设置异码间隔，$为时间 ms，建议值 200~400。这命令实际是控制感应间隔，大于同码间隔时会对同码也生效。

2、针对支付场景，扫码器只需要在付款时扫码，可设置为“指令触发模式”：

- “@SCNMOD0”设置为指令模式。此模式下，扫码器默认处于关闭状态，需要发送如下“开启一次识读”的指令才会进行一次扫码。
- “#SCNTRG1”开启一次识读。扫到码或超过等待时间后，立即回到关闭状态。
- “#SCNTRG0”关闭识读。
- “@ORTSET$”设置等待时间，$为时间 ms，建议值 60000。

3、其他常用命令：

- “@TSUENA1”激活后缀。
- “@TSUSET0D0A”设置后缀为回车换行。
- “@GRBENA1”开启蜂鸣器。