Printer Android SDK Instruction

Two parts for SDK:

1. **Printer**
2. **Customer Display**

SDK for Printer part:

1 Preface

We provide Gprinter Android SDK to user for developing and operating Gprinter printers more efficiently and convenient. If there is any problem or BUG during developing, please contact our engineer any time via MrAlading@163.com

2 Download GprinterSDK

Please contact the Customer Service Department of your purchased reseller or distributor for information.

3 Create a new project and copy packages to libs

1. gprinter-x.x.x.jar ；
2. jcc-bate-0.7.3.jar ；
3. ksoap2-android-assembly-2.5.2-jar-with-dependencies.jar ；
4. xUtils-2.6.14.jar ；
5. Armeabi ；
6. armeabi-v7a ；
7. x86 .

4 Register service and permission

Register Permission in AndroidManifest. Print service offered in gprinter-x.x.x.jar. It will have trouble connecting issues when don’t register service, please confirm the configuration.

Register Service：

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="com.sample"
          android:versionCode="1"
          android:versionName="1.0" >
    <uses-sdk
          android:minSdkVersion="14"
```

```
        android:targetSdkVersion="22" />

    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.hardware.usb.accessory" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
    <uses-permission android:name="android.permission.GET_TASKS" />
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_SETTINGS" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <uses-feature android:name="android.hardware.usb.host" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <service
            android:name="com.gprinter.service.GpPrintService"
            android:enabled="true"
            android:exported="true"
            android:label="GpPrintService" >
            <intent-filter>
                <action android:name="com.gprinter.aidl.GpPrintService" />
            </intent-filter>
```

```
          </service>


          <service android:name="com.gprinter.service.AllService" >
          </service>
     </application>
</manifest>
```

5 ADD aidl file

Create a new package named com.gprinter.aidl, add GpService.aidl into it, content is as below:

```
package com.gprinter.aidl;
interface GpService{
    int openPort(int PrinterId,int PortType,String DeviceName,int PortNumber);
    void closePort(int PrinterId);
    int getPrinterConnectStatus(int PrinterId);
    int printeTestPage(int PrinterId);
    void queryPrinterStatus(int PrinterId,int Timesout,int requestCode);
    int getPrinterCommandType(int PrinterId);
    int sendEscCommand(int PrinterId, String b64);
    int sendLabelCommand(int PrinterId, String b64);
    void isUserExperience(boolean userExperience);
    String getClientID();
    int setServerIP(String ip, int port);
}
```

6 Start and bind PrinterPrintService

```
private PrinterServiceConnection conn = null;
class PrinterServiceConnection implements ServiceConnection {
     @Override
     public void onServiceDisconnected(ComponentName name) {
          Log.i("ServiceConnection", "onServiceDisconnected() called");
          mGpService = null;
     }
     @Override
```

```
        public void onServiceConnected(ComponentName name, IBinder service) {
            mGpService = GpService.Stub.asInterface(service);
        }
    }
    @Override
    public void onCreate(Bundle savedInstanceState) {
        conn = new PrinterServiceConnection();
        Intent intent = new Intent(this, GpPrintService.class);
        bindService(intent, conn, Context.BIND_AUTO_CREATE); // bindService
    }
```

7 Using print service AIDL port

- int openPort(int PrinterId,int PortType,String DeviceName,int PortNumber);

  （Please refer to GpService.aidl to get more information about ports.）

  Connect Android device and printer communication interface, the interface will return the connection status via broadcast. The action of broadcast is ***GpCom.ACTION_CONNECT_STATUS,*** EXTRA includes ***GpPrintService.CONNECT_STATUS*** and ***GpPrintService.PRINTER_ID,*** to get connection status value and printer index ID respectively.

  PrinterId is the index ID of printer to be connected.

  PortType is connection type; please refer to the PortParameters from API to know more about connection types.

  DeviceName is address of connection device, generally Bluetooth address, USB device name or IP address.

  PortNumber is used for connecting printer via Ethernet or WiFi (default port number is 9100), Bluetooth or USB connection sets as 0.

- void closePort(int PrinterId);

  Disconnect Android device and printer communication interface (once disconnecting, please make sure the printer status to be ***GpDevice.STATE_NONE*** already if needs to connect again)

  PrinterId is the index ID of printer to be connected.

- int getPrinterConnectStatus(int PrinterId);

Get printer current connection status.

PrinterId is the index ID of printer to be connected.

- int printeTestPage(int PrinterId);

Printer, which the index ID is PrintID, prints test page.

PrinterId is the index ID of printer to be connected.

- void queryPrinterStatus(int PrinterId,int Timesout,int requestCode);

（Please refer to broadcast stating of **MainActivity. java** from GpSample.）

Querying printer’s real-time status, it will be sent a broadcast which is action ***GpCom.ACTION_DEVICE_REAL_STATUS***, please receive this broadcast in the project, with two **EXTRA:** ***GpCom.EXTRA_PRINTER_REQUEST_CODE*** and ***GpCom.EXTRA_PRINTER_REAL_STATUS***, both values are for **int** type.

PrinterId is the index ID of printer to be connected.

requestCode is querying request code

- int getPrinterCommandType(int PrinterId);

Get printer command type: ESC command or Label command.

PrinterId is the index ID of printer to be connected.

- int sendEscCommand(int PrinterId, String b64);

Transmit receipt content to printer which the index ID is **PrinterId**.

PrinterId is the index ID of printer to be connected.

b64 is the data after byte array proceeding to code base64.

- int sendLabelCommand(int PrinterId, String b64);

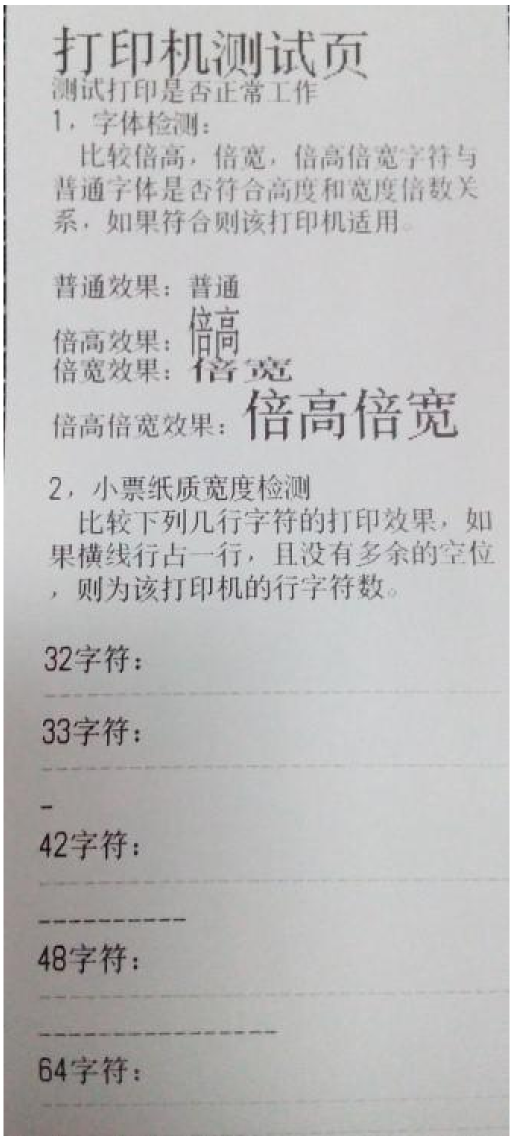
Transmit label content to printer which the index ID is **PrinterId**.

PrinterId is the index ID of printer to be connected.

b64 is the data after byte array proceeding to code base64.

8 Transmit data to printer

8.1 Receipt printer test page sample as picture blow:



打印机测试页
测试打印是否正常工作
1，字体检测：
　比较倍高，倍宽，倍高倍宽字符与普通字体是否符合高度和宽度倍数关系，如果符合则该打印机适用。

普通效果：普通
倍高效果：倍高
倍宽效果：倍宽
倍高倍宽效果：倍高倍宽

2，小票纸质宽度检测
　比较下列几行字符的打印效果，如果横线行占一行，且没有多余的空位，则为该打印机的行字符数。

32字符：

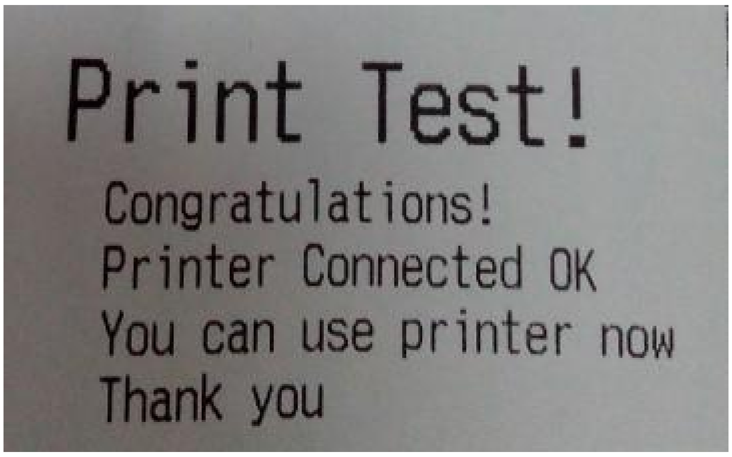
33字符：

-
42字符：

48字符：

64字符：

8.2 Label printer test page sample as picture blow:



Print Test!

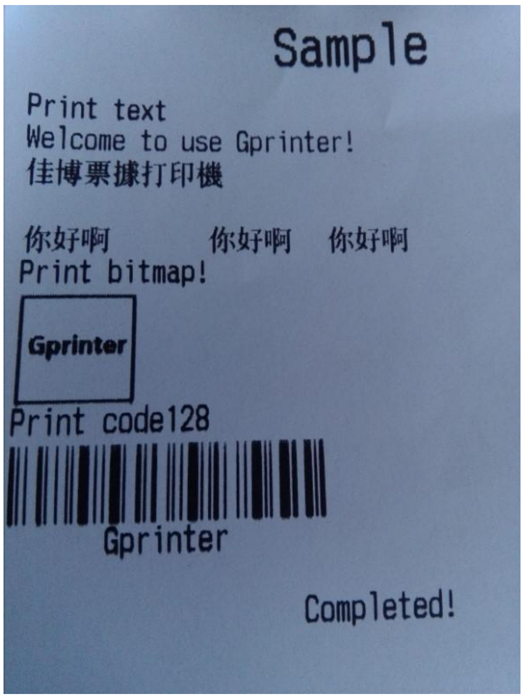
Congratulations!
Printer Connected OK
You can use printer now
Thank you

9 Printing commands

- For receipt:

```
EscCommand esc = new EscCommand();
esc.addPrintAndFeedLines((byte)3);
//set center justification
esc.addSelectJustification(JUSTIFICATION.CENTER);
// Turn on Double-height and Double-width mode
esc.addSelectPrintModes(FONT.FONTA,
                        NABLE.OFF,
                        ENABLE.ON,
                        ENABLE.ON, ENABLE.OFF);
// text “Sample” to be printed
esc.addText("Sample\n");
esc.addPrintAndLineFeed();
    /*Print text*/
// Turn off Double-height and Double-width mode
esc.addSelectPrintModes(FONT.FONTA,
                        ABLE.OFF,
                        ENABLE.OFF,
                        ENABLE.OFF,
                        ENABLE.OFF);
    esc.addSelectJustification(JUSTIFICATION.LEFT);// set left justification
    esc.addText("Print text\n");    //   text “Print text” to be printed
    esc.addText("Welcome to use printer!\n");    // text “welcome to use printer” to be
printed
    esc.addPrintAndLineFeed();
    /* Print image*/
    esc.addText("Print bitmap!\n");    //   text “print bitmap!” to be printed
    Bitmap b = BitmapFactory.decodeResource(getResources(),
            R.drawable.printer);
    esc.addRastBitImage(b,b.getWidth(),0);    // image to be printed
    /* Print 1D barcode */
    esc.addText("Print code128\n");    //   text “Print code128” to be printed
    esc.addSelectPrintingPositionForHRICharacters(HRI_POSITION.BELOW);//Set
printing position for HRI characters below the barcode
    esc.addSetBarcodeHeight((byte)60); // Set barcode height to 60 dots
    esc.addSetBarcodeWidth((byte)1); // Set barcode width to 1 dot
    esc.addCODE128(esc.genCode("123456789"));   //Print Code128
    esc.addCODE128(esc.genCodeB("Gprinter"));
    esc.addPrintAndLineFeed();
    /*QRCode command print
      This command is only used to the printers which supported QR code.
      Transmit QR code image to the printers which not supported QR code.
    */
    esc.addText("Print QRcode\n");    // text “Print QRcode” to be printed
    esc.addSelectErrorCorrectionLevelForQRCode((byte)0x31); //Set error correction
level for QR code
    esc.addSelectSizeOfModuleForQRCode((byte)3);// Set size of QR code module
    esc.addStoreQRCodeData("www.printer.com.cn");// Set QR code data
    esc.addPrintQRCode();// Printer QR Code
    esc.addPrintAndLineFeed();
    /* Print text */
    esc.addSelectJustification(JUSTIFICATION.CENTER);// Set center justification
    esc.addText("Completed!\r\n");    //   text “Completed!” to be printed
    // Open cash drawer
    esc.addGeneratePlus(LabelCommand.FOOT.F5, (byte) 255, (byte) 255);
```

```
//Open cash drawer immediately
//esc.addGeneratePluseAtRealtime(LabelCommand.FOOT.F2, (byte) 8);
esc.addPrintAndFeedLines((byte)8);
Vector<Byte> datas = esc.getCommand(); // Transmit data
Byte[] Bytes = datas.toArray(new Byte[datas.size()]);
byte[] bytes = GpUtils.ByteTo_byte(Bytes);
String str = Base64.encodeToString(bytes, Base64.DEFAULT);
intrel;
try {
  rel = mService.sendEscCommand(mPrinterIndex, str);
  GpCom.ERROR_CODE r=GpCom.ERROR_CODE.values()[rel];
  if(r != GpCom.ERROR_CODE.SUCCESS){
      Toast.makeText(getApplicationContext(),GpCom.getErrorText(r),
              Toast.LENGTH_SHORT).show();
         }
} catch (RemoteException e) {
  e.printStackTrace();
}
```



● For label:

```
LabelCommand label = new LabelCommand();
```

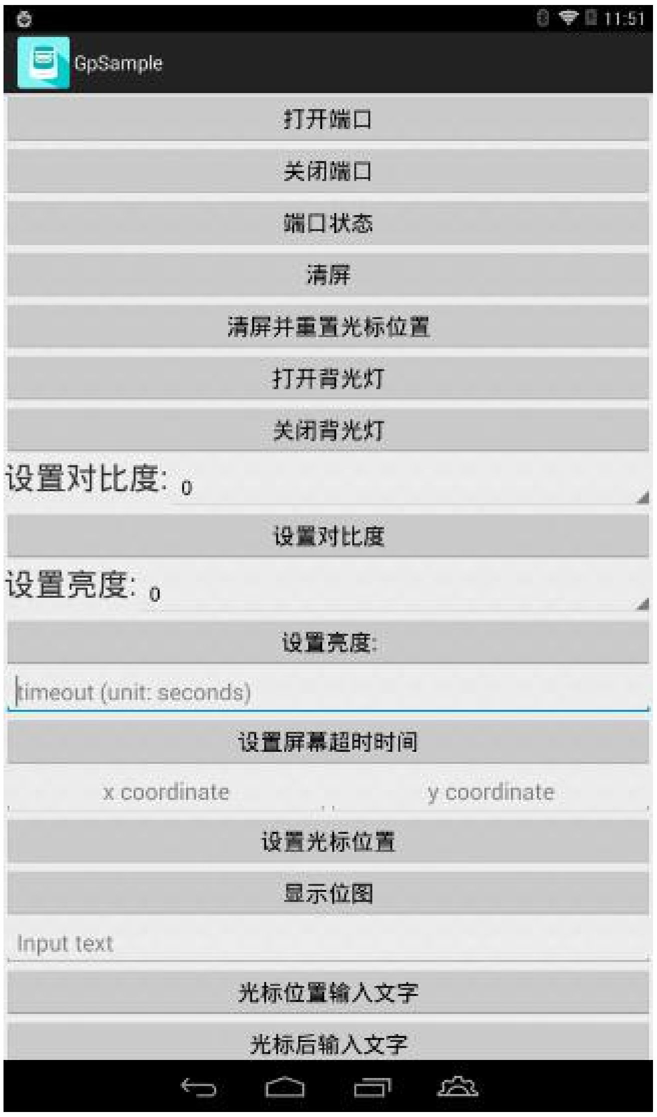
```
label.addSize(60, 60); //  Set label width and length as actual
label.addGap(0);                  //  set label gap as actual, if no gap then set 0
label.addDirection(DIRECTION.BACKWARD,MIRROR.NORMAL);//set print direction
label.addReference(0, 0);//  Set the reference point of the label
label.addTear(ENABLE.ON); //  Turn on tear off paper mode
label.addCls();// Clear the buffer
//Input text
label.addText(20,20,
                FONTTYPE.SIMPLIFIED_CHINESE,
                ROTATION.ROTATION_0,
                FONTMUL.MUL_1,
                FONTMUL.MUL_1,
                "Welcome to use Printer!");
// Input image
Bitmap b = BitmapFactory
              .decodeResource(getResources(),R.drawable.printer);
label.addBitmap(20,50,
                    BITMAP_MODE.OVERWRITE,
                    b.getWidth(),
                    b);
label.addQRCode(250, 80, EEC.LEVEL_L,5,
                     ROTATION.ROTATION_0,
                     "gprinter.com");
// Input 1D bar code
label.add1DBarcode(20,250,
                         BARCODETYPE.CODE128,
                         100,
                         READABEL.EANBEL,
                         ROTATION.ROTATION_0,
                         "printer");

label.addPrint(1,1); // Print label
label.addSound(2, 100); // Beep after print
Vector<Byte> datas = label.getCommand(); // Transmit data
Byte[] Bytes = datas.toArray(new Byte[datas.size()]);
```

```
    byte[] bytes = ArrayUtils.toPrimitive(Bytes);
    String str = Base64.encodeToString(bytes, Base64.DEFAULT);
    intrel;
    try {
        rel = mGpService.sendLabelCommand(mPrinterIndex, str);
        GpCom.ERROR_CODE r=GpCom.ERROR_CODE.values()[rel];
        if(r != GpCom.ERROR_CODE.SUCCESS){
            Toast.makeText(getApplicationContext(),
                           GpCom.getErrorText(r),
                           Toast.LENGTH_SHORT)
                           .show();
      }
    } catch (RemoteException e) {
      e.printStackTrace();
    }
```

10. SDK instruction for Customer Display

Please refer to **CustomerDispaly** from **API** to know ports in details.

Open port, close port and get port status as picture showing below:



- Open port

```
/*  打开端口 */
private CustomerDisplay port;
private void openPort() {
port = CustomerDisplay.getInstance(this);
try {
// 打开端口
port.openPort();
} catch (IOException e) {
e.printStackTrace();
}
// 设置监听回调数据
port.setReceivedListener(this);
port.getBacklightTimeout();
}
```

- Close port

```
// 关闭端口
private void closePort() {
if (port != null) {
port.closeSerialPort();
toast(" 关闭端口 ");
}
}
```

- Get port status

```
// 获取端口状态
private void getPortStatus() {
if (port != null) {
toast(" 端口状态： " + (port.isPortOpen() ? " 打开 " : " 关闭 "));
} else {
toast(" 端口状态：关闭 ");
}
}
```

- Set backlight brightness

```
// 设置背光灯亮度
private void setBrightness() {
Spinner spinner = (Spinner) findViewById(R.id.spinner_brightness);
byte brightness=Byte.valueOf(spinner.getSelectedItem().toString());
port.setBrightness(brightness);
}
```

- Set customer display contrast

```
private void setContrast() {
Spinner spinner = (Spinner) findViewById(R.id.spinner_contrast);
byte contrast=Byte.valueOf(spinner.getSelectedItem().toString());
port.setContrast(contrast);
}
```

- Turn off backlight

```
private void turnOffBacklight() {
port.setBacklight(false);
}
```

- Reset customer display

```
//  复位客显
private void reset() {
port.reset();
}
```

- Clear display

```
//  清屏
private void clear() {
port.clear();
}
```

- Display bitmap

```
//  显示位图
private void displayBitmap() {
Bitmap bitmap=BitmapFactory
.decodeStream(getResources().openRawResource(R.raw.fl));
port.displayBitmap(bitmap, 48);
}
```

- Turn on backlight

```
//  打开背光灯
private void turnOnBacklight() {
port.setBacklight(true);
}
```

- Set display timeout

```
//  设置客显背光灯超时时间
private void setDisplayTimeout() {
```

```
String displayTimeoutStr = mEtDisplayTimeout.getText().toString();
if (TextUtils.isEmpty(displayTimeoutStr)) {
Toast.makeText(CustomerDiaplayActivity.this,
R.string.str_input_display_timeout,
Toast.LENGTH_SHORT).show();
return;
}
mTimeoutMsg.setText(R.string.str_set_successful);
int timeout = Integer.parseInt(displayTimeoutStr);
port.setBacklightTimeout(timeout);
}
```

- Set cursor position

```
//    设置客显光标位置
private void setCursorPosition() {
String xStr = mEtX.getText().toString();
String yStr = mEtY.getText().toString();
if (TextUtils.isEmpty(xStr) || TextUtils.isEmpty(yStr)) {
Toast.makeText(this,
R.string.str_please_input_x_y,
Toast.LENGTH_SHORT).show();
return;
}
int x = Integer.parseInt(xStr);
int y = Integer.parseInt(yStr);
port.setCursorPosition(x, y);
}
```

- Set text current cursor, cursor position no changes.

```
private void setTextCurrentCursor() {
String inputText = mInputText.getText().toString();
port.setTextCurrentCursor(inputText);
}
```

- Set text current cursor, cursor position changes according to text inputing.

```
private void setTextBebindCursor() {
String inputText = mInputText.getText().toString();
port.setTextBebindCursor(inputText);
}
```

11. Get Customer Display information, and implement callback information

```
/**
* get　部分
*/
private void getDisplayStatus() {
port.getBacklight();
}
private void getDisplayTimeout() {
port.getBacklightTimeout();
}
private void getCursorPosition() {
port.getCursorPosition();
}
private void getDisplayRowAndColumn() {
port.getDisplayRowAndColumn();
}
/**
*　获取客显屏的状态开启或关闭
*
* @param isOpen
* 客显屏背光灯开启或关闭
*/
@Override
public void onPortOpen(boolean isOpen) {
if (isOpen) {
toast(" 打开端口成功 ");
} else {
```

```
toast(" 打开端口失败 ");
}
}
/**
*  获取客显屏背光灯开启或关闭
*
* @param isOn
* 客显屏背光灯开启或关闭
*/
@Override
public void onBacklightStatus(final boolean isOn) {
Log.d("==onBacklightStatus==", String.valueOf(isOn));
toast("==onBacklightStatus==  背光灯状态 ->" + String.valueOf(isOn));
}
/**
*  获取客显屏光标的位置
*
* @param x
* 横坐标
* @param y
* 纵坐标
*/
@Override
public void onCursorPosition(final int x, final int y) {
toast("==onCursorPosition==x = " + x + ",y =" + y);
Log.d("==onCursorPosition==", "x 坐标 = " + x + ",y 坐标 =" + y);
}
/**
*  获取客显屏的行和列
* @param row
* 行
* @param column
* 列
*/
@Override
```

```
public void onDisplayRowAndColumn(final int row, final int column) {
toast(" 行数 = " + row + ", 列数 =" + column);
Log.d("==onCursorPosition==", "row = " + row + ",column =" + column);
}
/**
*   获取客显屏背光灯超时时间
*
* @param timeout
* 单位：秒
*/
@Override
public void onBacklightTimeout(final int timeout) {
toast(" 超时时间 = " + timeout);
Log.d("==onBacklightTimeout==", "timeout = " + timeout);
}
/**
*   更新客显固件完成回调方法
*/
@Override
public void onUpdateSuccess() {
//noting to do
}
@Override
public void onUpdateFail(String error) {
//noting to do
}
```