

# Android快速接入文档

## 一、权限

## 二、初始化

## 三、设置打印机状态回调

## 四、连接打印机

## 五、绘制相关

### 5.1 初始化画板

### 5.2 绘制文本

### 5.3 绘制一维码

### 5.4 绘制二维码

### 5.5 绘制矩形框

### 5.6 绘制线条

### 5.7 生成绘制标签的Json数据

## 六、打印相关

### 6.1 打印回调

### 6.2 开始打印任务

### 6.3 提交打印任务

### 6.4 结束打印

### 6.5 取消打印

## 一、权限

HTML | 复制代码

```
1  <!-- 蓝牙权限 通过蓝牙与打印机交互需要声明的权限 -->
2  <uses-permission android:name="android.permission.BLUETOOTH" />
3  <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
4  <!-- 读写权限 进行固件升级时需要的权限 -->
5  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
6  />
7  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
8  />
```

## 二、初始化

Java | 复制代码

```
1 public class TestActivity extends Activity {  
2     private JCPrintApi jcapi;  
3  
4     @Override  
5     protected void onCreate(Bundle savedInstanceState) {  
6         jcapi = JCPrintApi.getInstance(null);  
7         jcapi.init(getApplication());  
8     }  
9 }
```

## 三、设置打印机状态回调

```
1 ▾ Callback mCallback = new Callback() {  
2  
3     @Override  
4 ▾     public void onConnectSuccess(String deviceAddress, int connectType) {  
5         //连接成功回调，返回设备地址和连接类型  
6     }  
7  
8     @Override  
9 ▾     public void onDisConnect() {  
10        //连接失败或打印机断开时回调该接口  
11    }  
12  
13    @Override  
14 ▾     public void onElectricityChange(int powerLevel) {  
15        //电量变化回调  
16    }  
17  
18    @Override  
19 ▾     public void onCoverStatus(int coverStatus) {  
20        //上盖状态变化回调  
21    }  
22  
23    @Override  
24 ▾     public void onPaperStatus(int paperStatus) {  
25        //标签状态变化回调  
26    }  
27  
28    @Override  
29 ▾     public void onRfidReadStatus(int rfidReadStatus) {  
30        //打印机读取RFID状态回调  
31    }  
32  
33    @Override  
34 ▾     public void onPrinterIsFree(int freeState) {  
35        //打印机忙碌是否空闲状态回调  
36    }  
37  
38    @Override  
39 ▾     public void onHeartDisConnect() {  
40        //心跳断开回调  
41    }  
42 };  
43  
44 jcapi.setCallBack(callback);
```

## 四、连接打印机



Java | 复制代码

```
1 //此链接为同步耗时方法，请使用子线程调用。
2 jcapi.openPrinterByAddress(this.getApplication(),"DC:0D:30:7D:46:BB",Constant.BLUETOOTH_SPP_CONNECT);
```

## 五、绘制相关

### 5.1 初始化画板



HTML | 复制代码

```
1 /**
2  * 初始化画板
3  * @param width 宽
4  * @param height 高
5  * @param rotate 旋转
6  * @param fontDir 字体路径
7  */
8 public void drawEmptyLabel(float width, float height,int rotate, String
   fontDir) {
9     JcImageSdkApi.InitDrawingBoard(width,height,rotate,fontDir);
10 }
```

### 5.2 绘制文本

```
1  /**
2   *
3   * @param x 位置x
4   * @param y 位置y
5   * @param width 宽
6   * @param height 高
7   * @param value 内容
8   * @param fontFamily 字体名称,未传输字体为空字符串时使用默认字体,暂时用默认字体
9   * @param fontSize 字体大小
10  * @param rotate 旋转
11  * @param textAlignHorizontal 水平对齐方式: 0:左对齐 1:居中对齐 2:右对齐
12  * @param textAlignVertical 垂直 对齐方式: 0:顶对齐 1:垂直居中 2:底对齐
13  * @param lineModel 1:宽高固定, 内容大小自适应 (字号/字符间距/行间距 按比例缩放)
14  * @param letterSpacing 字母之间的标准间隔, 单位mm
15  * @param lineHeight 行间距 (倍距), 单位mm
16  * @param mFontStyles 字体样式[斜体, 加粗, 下划线, 删除下划线 (预留)]
17  */
18  public void drawLabelText(float x, float y, float width, float height,
19                          String value, String fontFamily, float
20                          fontSize, int rotate, int textAlignHorizontal, int textAlignVertical,
21                          int lineModel, float letterSpacing, float
22                          lineHeight, boolean [] mFontStyles) {
23      JcImageSdkApi.DrawLabelText(x,y,width,height,value,fontFamily,fontSize,ro
24      tate,textAlignHorizontal,textAlignVertical,lineModel,letterSpace,lineSpac
25      e,mFontStyles);
26  }
```

### 5.3 绘制一维码

```
1  /**
2   *
3   * @param x 水平坐标
4   * @param y 垂直坐标
5   * @param width 宽度,单位mm
6   * @param height 高度,单位mm
7   * @param codeType 一维码类型20:CODE12821:UPC-A,22:UPC-
8   * @param value 文本内容
9   * @param fontSize 文本字号
10  * @param rotate 旋转角度, 仅支持0,90,180,270
11  * @param textHeight 文本高度
12  * @param textPosition 文本位置, int,一维码文字识别码显示位置,0:下方显示,1:上方显示,2:不显示
13  */
14  public void drawLabelBarCode(float x, float y, float width, float
15  height,
16  int codeType, String value, float fontSize,
17  int rotate,
18  int textHeight, int textPosition) {
    JcImageSdkApi.DrawLableBarCode(x,y,width,height,codeType,value,fontSize,
    rotate,textHeight,textPosition);
}
```

## 5.4 绘制二维码

```
1  /**
2   *
3   * @param x 水平坐标
4   * @param y 垂直坐标
5   * @param width 宽度,单位mm
6   * @param height 高度,单位mm
7   * @param value 文本内容
8   * @param codeType 一维码类型,31:QR_CODE,32:PDF417,33:DATA_MATRIX,34:AZTEC
9   * @param rotate 旋转角度, 仅支持0,90,180,270
10  */
11  public void drawLabelQrCode(float x, float y, float width, float height,
12  String value,int codeType, int rotate) {
13      JcImageSdkApi.DrawLableQrCode(x,y,width,height,value,codeType,rotate);
14  }
```

## 5.5 绘制矩形框

```

1  /**
2   *
3   * @param x 水平坐标
4   * @param y 垂直坐标
5   * @param width 宽度,单位mm
6   * @param height 高度,单位mm
7   * @param graphType 线条类型,1:实线,2:虚线类型,虚实比例1:1
8   * @param rotate 旋转角度, 仅支持0,90,180,270
9   * @param cornerRadius 圆角
10  * @param lineWidth 线宽
11  * @param lineType 图形类型
12  * @param dashwidth 线条为虚线宽度, 【实线段长度, 空线段长度】
13  */
14  public void drawLabelGraph(float x, float y, float width, float height,
15  int graphType, int rotate,
16  float cornerRadius, float lineWidth, int
17  lineType, float [] dashwidth) {
18      JcImageSdkApi.DrawLableGraph(x,y,width,height,graphType,rotate,cornerRad
19  ius,lineWidth,lineType,dashwidth);
20  }

```

## 5.6 绘制线条

```

1  /**
2   *
3   * @param x 水平坐标
4   * @param y 垂直坐标
5   * @param width 宽度,单位mm
6   * @param height 高度,单位mm
7   * @param rotate 旋转角度, 仅支持0,90,180,270
8   * @param lineType 线条类型,1:实线,2:虚线类型,虚实比例1:1
9   * @param dashWidth 线条为虚线宽度, 【实线段长度, 空线段长度】
10  */
11  public void drawLabelLine(float x, float y, float width, float height,
12  int rotate, int lineType, float [] dashWidth) {
13      JcImageSdkApi.DrawLableLine( x, y, width, height, rotate,
14  lineType, dashWidth);
15  }

```



## 5.7 生成绘制标签的Json数据



Java

复制代码

```
1  /**
2   *
3   * @return 标贴完整json
4   */
5  public byte[] generateLabelJson() {
6      return JcImageSdkApi.GenerateLableJson();
7  }
```

## 六、打印相关

### 6.1 打印回调

```
1 //打印回调
2 private PrintCallback printCallback = new PrintCallback() {
3     int pageIndex = 0;
4     int quantityIndex = 0;
5
6     // 普通模式下的打印进度回调, copies表示份数
7     @Override
8     public void onPrintProgress(int copies) {
9         quantityIndex = copies;
10        runOnUiThread(new Runnable() {
11            @Override
12            public void run() {
13                progressBar.setProgress(pageIndex, quantityIndex);
14                if (pageIndex == (pageCount)) {
15                    finishPrint();
16                }
17            }
18        });
19    }
20
21    // 普通模式下, 打印页数的进度回调
22    @Override
23    public void onPrintPageCompleted(final int pageIndex) {
24
25    }
26
27    // 碳带用尽时的回调
28    @Override
29    public void onRibbonUsed(double ribbonUsed) {
30        runOnUiThread(new Runnable() {
31            @Override
32            public void run() {
33                Toast.makeText(V3TestActivity.this, "碳带用尽",
34                    Toast.LENGTH_SHORT).show();
35            }
36        });
37    }
38
39    @Override
40    public void onPageNumberReceivingTimeout() {
41
42    }
43
44    // 打印异常回调, errorCode为错误码
45    @Override
46    public void onError(int errorCode) {
```

```

45         progressBar.dismissDialog();
46         handleError(type);
47     }
48
49     // 暂停模式下的异常回调, printState表示打印状态, 0表示在打印, 1表示打印暂停, 2表示打印停止
50     // errorCode错误码
51     @Override
52     public void onError(int errorCode, int printState) {
53
54     }
55
56     // 打印缓冲区有空闲时的回调, pageIndex表示打印页数序号, bufferSize表示缓存可用大小
57     @Override
58     public void onBufferFree(int pageIndex, int bufferSize) {
59         //提交打印任务
60         api.commitData(jsonList.subList(pageIndex - 1, Math.min(pageIndex - 1 + bufferSize, jsonList.size()))),
61             infoList.subList(pageIndex - 1, Math.min(pageIndex - 1 + bufferSize, jsonList.size())));
62     }
63
64     //打印进度回调, pageIndex表示页数, quantityIndex表示份数,
65     //计数都是从1开始
66     @Override
67     public void onProgress(int pageIndex, int quantityIndex) {
68
69     }
70 };

```

## 6.2 开始打印任务



Java

复制代码

```

1  /**
2   * @param density 打印浓度 -2至5
3   * @param paperType 纸张类型 <1>,间隙纸; <2>,黑标纸; <3>,连续纸; <5>,透明纸;
   * <6>,标牌
4   * @param printMode 打印模式
5   * @param printCallback 打印相关回调
6   */
7  public void startJob(final int density, final int paperType, final int
    printMode, final PrintCallback printCallback)

```

### 6.3 提交打印任务

Java | 复制代码

```
1  /*
2   * json形式提交数据
3   * 提交打印数据, 构建打印数据, 参考demo
4   */
5  ArrayList printInfoList= infoList.subList(pageIndex - 1,
6  Math.min(pageIndex - 1 + bufferSize, jsonList.size()))
7  ArrayList printdataList= jsonList.subList(pageIndex - 1,
8  Math.min(pageIndex - 1 + bufferSize, jsonList.size()))
9  jcapi.commitData(dataList,printInfoList);
10
11 /*
12 * bitmap形式提交数据
13 *
14 * @param orientation 旋转角度
15 * @param printBitmap 图片
16 * @param pageWidth 画板宽度
17 * @param pageHeight 画板高度
18 * @param printQuantity 打印份数
19 */
20 jcapi.commitImageData(final int orientation,final Bitmap
21 printBitmap,final int pageWidth,final int pageHeight,final int
22 printQuantity,final int marginTop,final int marginLeft,final int
23 marginBottom,final int marginRight)
```

### 6.4 结束打印


Java | 复制代码

```
1  //结束打印
2  jcapi.endJob();
```

### 6.5 取消打印



Java

 复制代码

```
1 //取消打印
2 jcapi.cancelJob();
```