

扫描开发指南

文档修订记录

版本号	*变化 状态	简要说明	日期	变更人	批准 日期	批准 人
V1.0	c	初始版本	2021/3/15	Ct		
V1.1	M	迭代版本	2022/10/10	King		

*变化状态：C = 创立，A = 增加，M = 修改，D = 删除

文档审批记录

序号	审批人	角色	审批日期	签字	备注

目录

扫描开发指南 1

1. 简介 3

2. 注意事项 3

 2.1 配置 android 开发环境 3

 2.2 扫描调用流程 4

..... 4

..... 4

3.接口 4

 3.1.0 检查扫描是否打开 4

 3.1.1 打开扫描 5

 3.1.2 关闭扫描 5

 3.1.3 启动扫描 5

 3.1.4 停止扫描 5

 3.1.5 设置连续扫描 5

 3.1.6 设置扫描模式 6

 3.1.7 获取当前连续扫描状态 6

 3.1.8 获取当前扫描模式 6

 3.1.9 重置扫描 7

4 附录 1 7

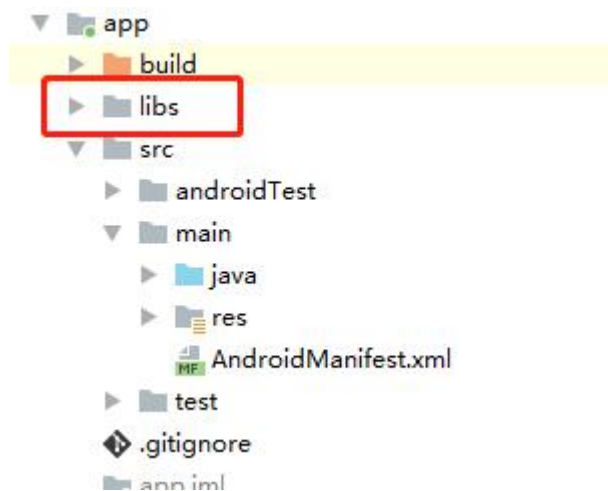
 4.1.调用示例 7

1. 简介

为了方便进行二次开发，我们提供了可以在 Java 平台进行运行的函数库。该库采用 Java 语言编写。

2. 注意事项

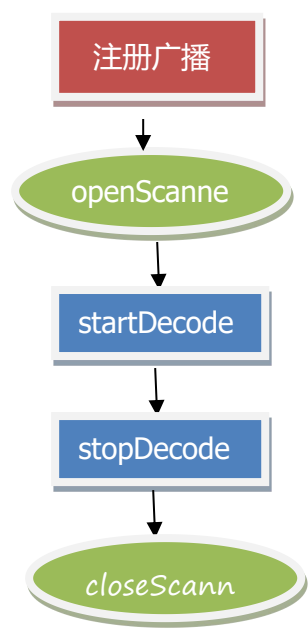
2.1 配置 android 开发环境



1、将 demo 中的 libs 文件夹下的 jar 包导入项目

2、在 app/ build.gradle 中添加 jar 包的引用

2.2 扫描调用流程



3.接口

3.1.0 检查扫描是否打开

函数接口	public boolean isScanOpened()
功能说明	扫描是否打开
参数说明	无
返回值	boolean

3.1.1 打开扫描

函数接口	public boolean openScan()
功能说明	打开扫描
参数说明	无
返回值	boolean

3.1.2 关闭扫描

函数接口	public boolean closeScan()
功能说明	关闭扫描
参数说明	无
返回值	boolean

3.1.3 启动扫描

函数接口	public boolean startScan()
功能说明	开始扫描
参数说明	无
返回值	boolean

3.1.4 停止扫描

函数接口	public boolean stopScan()
功能说明	停止扫描
参数说明	无
返回值	boolean

3.1.5 设置连续扫描

函数接口	public void setScanLaserMode(int mode)
------	--

功能说明	开启或关闭连续扫描
参数说明	mode:4 开 启连续扫描 mode:8 关 闭连续扫描
返回值	无

3.1.6 设置扫描模式

函数接口	public boolean setOutScanMode(int mode)
功能说明	设置扫描模式
参数说明	mode:0 广播模式 mode:1 编辑框模式 mode:2 键盘模式
返回值	无

3.1.7 获取当前连续扫描状态

函数接口	public void setScanLaserMode(int mode)
功能说明	获取当前连续扫描状态
参数说明	mode:4 开 启连续扫描 mode:8 关 闭连续扫描
返回值	Int

3.1.8 获取当前扫描模式

函数接口	public int getOutScanMode()
功能说明	获取当前扫描模式
参数说明	mode:0 广播模式 mode:1 编辑框模式 mode:2 键盘模式

返回值	Int
-----	-----

3.1.9 重置扫描

函数接口	public boolean resetScan()
功能说明	重置扫描
参数说明	无
返回值	boolean

4 附录 1

4.1.调用示例

```
// 实例化

ScanDevice sd = new ScanDevice();
sm.setOutScanMode(0); // 模式-值:0 广播模式, 1 个编辑框模式, 2 个键盘模式
```

```
// 广播模式下需注册 scan.rcv.message 广播
```

```
IntentFilter filter = new IntentFilter();
filter.addAction(SCAN_ACTION);
registerReceiver(mScanReceiver, filter);
```

广播示例:

```
private BroadcastReceiver mScanReceiver=new BroadcastReceiver () {
    @Override
    public void onReceive(Context context, Intent intent) {
        Log.e ("TAG", "onReceive: "+intent.getAction ());
    }
};
```

```

String action = intent.getAction ();
if (action.equals (SCAN_ACTION)){
    byte[] barcode=intent.getByteArrayExtra ("barcode");
    int barocodelen=intent.getIntExtra ("length", 0);
    byte temp=intent.getByteExtra ("barcodeType", (byte) 0);
    byte[] aimid=intent.getByteArrayExtra ("aimid");
    barcodeStr=new String (barcode, 0, barocodelen);
    showScanResult.append (barcodeStr);
    showScanResult.append ("\n");
    UtilSound.play ();
    sm.stopScan ();
}
}
};

```

广播接收参数

```

byte[] broadCode = intent.getByteArrayExtra("barcode"); // 条码数据
int broadCodeLen = intent.getIntExtra("length", 0); // 数据长度

```