商米扫码

为什么要使用商米封装的扫码SDK

商米提供了适配自己设备的扫码SDK，相对于目前使用的开源方案，商米的扫码SDK有以下5个优势

1. 识别率高，经过大量模拟真实场景的测试，商米扫码SDK相对普遍使用的基于ZXing开源项目的扫码方案在扫码识别率上平均提高了74%。
2. 比ZXing方案在一维码扫码速度上快了100%以上。
3. 使用方式更简单，5行代码就能在自己的项目中添加扫码功能。
4. 支持扫描多达17种码，后续还将添加更多的码种。
5. 与商米的设备完美适配，软硬件结合可以保证功能的高效稳定。

怎么使用商米的扫码SDK

开发者有两种方式使用商米的扫码SDK

1. 开发者的应用调用SUNMIUI系统集成的扫码模块完成扫码，获取返回值，该方法简单易用。
2. 自己写相机界面，调用商米的封装的扫码SDK完成图片的解析，该方式相对复杂，但提供了更高的自由度。

第一种使用方式：

为了降低开发难度，SUNMI OS内置了一个扫码的模块，开发者在项目需要调用扫码的地方通过startActivityForResult()调用商米的扫码模块，然后在onActivityResult()方法中接受扫码结果返回值。

/\*

外部应用在自己的业务代码需要启动扫码的地方使用下面的方式创建Intent，然后使用startActivityForResult()调用起商米的扫码模块;

\*/

Intent intent = new Intent("com.summi.scan");

intent.setPackage("com.sunmi.sunmiqrcodescanner")

/\*

使用该方式也可以调用扫码模块

Intent intent = new Intent("com.summi.scan");

intent.setClassName("com.sunmi.sunmiqrcodescanner",

"com.sunmi.sunmiqrcodescanner.activity.ScanActivity");

扫码模块有一些功能选项，开发者可以通过传递参数控制这些参数，

所有参数都有一个默认值，开发者只要在需要的时候添加这些配置就可以。

intent.putExtra("CURRENT\_PPI", 0X0003);//当前分辨率

//M1和V1的最佳是800\*480,

//PPI\_1920\_1080 = 0X0001;

//PPI\_1280\_720 = 0X0002;

//PPI\_BEST = 0X0003;

intent.putExtra("PLAY\_SOUND", true);// 扫描完成声音提示 默认true

intent.putExtra("PLAY\_VIBRATE", false);

//扫描完成震动,默认false，目前M1硬件支持震动可用该配置，V1不支持

intent.putExtra("IDENTIFY\_MORE\_CODE", false);// 识别画面中多个二维码，默认false

intent.putExtra("IS\_SHOW\_SETTING", true);// 是否显示右上角设置按钮，默认true

intent.putExtra("IS\_SHOW\_ALBUM", true);// 是否显示从相册选择图片按钮，默认true

\*/

startActivityForResult(intent, START\_SCAN);

在onActivityResult方法中接收返回的扫码结果参数，参考如下代码：

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

super.onActivityResult(requestCode, resultCode, data);

if (requestCode == 1 && data != null) {

Bundle bundle = data.getExtras();

ArrayList result = (ArrayList<>) bundle .getSerializable(“data”);

Iterator<> it = result.iterator();

while (it.hasNext()) {

HashMap hashMap = it.next();

Log.i(“sunmi”, hashMap.get(“TYPE”));//扫码类型

Log.i(“sunmi”, hashMap.get(“VALUE”));//扫码结果

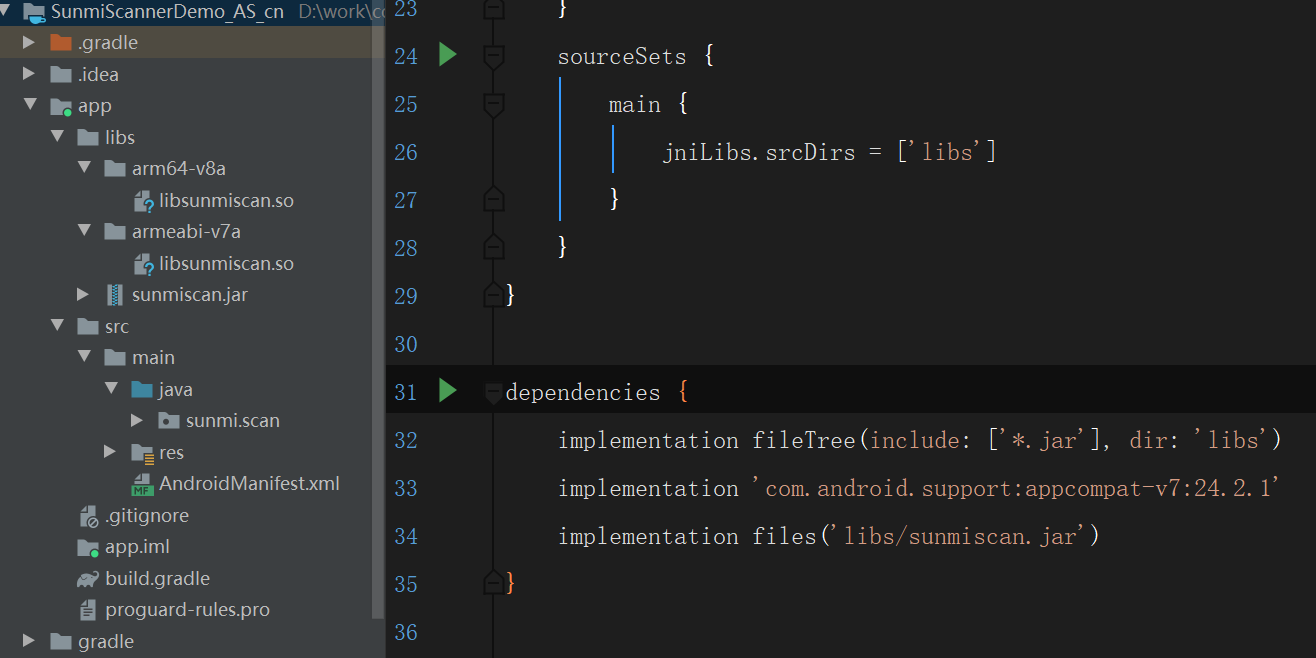
}

}

}

第二种方式：

1. 在项目的libs目录中按以下层级添加libsunmiscan.so和sunmiscan.jar两个文件，以及在build.gradle添加扫码库的引用，如下图所示：



2. 在处理业务的代码中引入相关java接口类 ：

import com.sunmi.scan.Config;

import com.sunmi.scan.Image;

import com.sunmi.scan.ImageScanner;

import com.sunmi.scan.Symbol;

import com.sunmi.scan.SymbolSet;

3. 相关参数初始化：

private ImageScanner scanner;

scanner = new ImageScanner();//创建扫描器

/\*创建解码图像，width, height 分别为摄像头预览分辨率的宽度和高度。一般来说,分辨率越高图像越清晰，但解码速度越慢，建议选取不大于1280X720分辨率，推荐的分辨率为640X480,800X480,1280X720，可以通过getSupportedPreviewSizes获取当前设备支持的分辨率。预览图像的默认格式为 YCbCr\_420\_SP,参数"Y800"表示取YUV的“Y”分量数据

\*/

Image source = new Image(previewSize\_width, previewSize\_height, "Y800");

scanner.setConfig(0, Config.ENABLE\_MULTILESYMS, 0);//是否开启同一幅图一次解多个条码,0表示只解一个，1为多个

scanner.setConfig( Symbol.QRCODE,Config.ENABLE, 1);//允许识读QR码，默认1:允许

scanner.setConfig( Symbol.PDF417,Config.ENABLE, 1);//允许识读PDF417码，默认0：禁止

scanner.setConfig(Symbol.DataMatrix, Config.ENABLE, 1);//允许识读DataMatrix码，默认0：禁止

scanner.setConfig(Symbol.AZTEC, Config.ENABLE, 1);//允许识读AZTEC码，默认0：禁止

4. 图像解码，如果是使用android系统相机解码，可以在图像预览回调函数即PreviewCallback.onPreviewFrame(byte[] data, Camera camera)方法中直接调用预览数据解码。

source.setData(data); //填充图像数据,data为摄像头原始数据

int result = scanner.scanImage(source); //解码

如果是其他形式的图像数据，如相册图片，需要将相应格式的图像转换为BMP格式，如果是彩色图像，还需要转化为灰度图像，目前解码库只处理灰度图像数据（即每个像素点的亮度值范围为0~255）。

5. 获取解码结果和条码类型。

if(result >0){

SymbolSet syms = scanner.getResults();

for (Symbol sym : syms) {

Log.i("sunmi", "码型:"+sym.getSymbolName());//条码类型,如“EAN-8”

Log.i("sunmi","结果:"+sym.getResult())//解码结果字符串

}

}

更多说明

1. 商米扫码SDK目前支持的扫码类型包括如下：

* 一维码: EAN-8, EAN-13, UPC-A, UPC-E, Codabar, Code39, Code93, Code128, ISBN10, ISBN13, DataBar, DataBar Expanded, Interleaved 2 of 5
* 二维码: QR Code , PDF417，DataMatrix，AZTEC

2. 更新解码库时请注意同时更新libsunmiscan.so和sunmiscan.jar两个文件

3. 虽然最新的解码库也提供了64位的解码库（arm64-v8a目录下），但如果当前设备为64位的平台，也可以兼容32位平台（armeabi-v7a目录下），在build.gradle中可以指定平台类型，代码如下：

externalNativeBuild {

cmake {

// abiFilters "arm64-v8a" //指定为arm64-v8a

abiFilters "armeabi-v7a" //指定为armeabi-v7a

}

}