OCRLib 识别服务使用文档

信大捷安

编写： 崔华伟
日期： 2016-10-10
版本： Version 1.0
使用环境：Android4.0 以上版本

需要在应用中集成 OCRLib.apk 识别功能应用，可以按照本说明来实现需要的功能。使用过程中，如果有问题请联系： cuihauwei@xdja.com

版本修改内容：

修改人	内容	版本	日期
崔华伟	初稿	Version 1.0	2016-10-10

第三方应用为了能够识别车牌，身份证，驾驶证，护照等主要证件照里面的证件号码，可以使用本应用；使用本应用的方法可以集成到第三方的应用里面，然后第三方通过调用识别库定义的接口就可以使用证件识别功能。

- 集成的方式可以使用安装 apk 应用，然后通过应用间接口调用的方式来识别证件号码；这种方式下：

  1）第三方应用可以直接调用相机界面，在相机界面扫描要识别的证件，识别后的结果通过 Intent 来返回给调用者。
  2）第三方应用也可以传入一张含有证件的照片来识别。要求图片是 JPG 格式的图片，传入的图片要求人眼清晰可见。

- 也可以使用 jar 包的方式来识别证件信息，这种方式下，不提供界面，第三方应用可以通过 API 接口，传入图片参数进行识别。

传入图片识别的时候，直接传入图片类型，和图片的文件路径。要求图片是 JPG 格式的图片；

含有车牌图片的大小要求最好在小于 800*700， 大于 80* 50 之间，如果含有车牌的图片大于这个尺度，程序会自动剪裁，会影响识别速度；如果小于最小范围，会因为图片信息不够，造成识别错误；

证件图片大小要求最好推荐使用的分辨率为 1280*960，其次是 1600*1200，最后是 2048*1536，大于 480*320 ；如果图片过大，会自动剪裁为合适大小的图片。

主要内容：

1. APK 集成方式 OCRLib.apk 集成
- 识别应用文件安装
- 识别接口调用
- 调用相机扫描方式调用
  - 车牌识别调用
  - 身份证识别调用
  - 驾驶证识别调用
  - 护照识别调用
- 通过图片地址传输识别调用
  - 图片作为参数调用

2. 封装 Jar 包集成方式
- 认证文件添加
- 资源文件的添加

- 车牌识别调用
- 身份证识别调用
- 驾驶证识别调用

3. 原生 Jar 包集成方式

- 信大捷安的认证文件
- 认证文件位置和认证方式
- 调用认证服务
- 调用识别服务

一. APK 集成方式 OCRLib.apk 集成：

识别文件的安装

调用识别功能，需要首先在手机上安装 OCRLib.apk 文件，安装以后就可以调用这些功能了。您可以选择手动安装或者是嵌入到程序里面，由程序自动安装。

下面介绍通过程序自动安装的情况。

- 在需要识别功能的程序里面新建一个 asset 文件夹，把需要安装的 OCRLib.apk 拷贝到这个文件下面。
- 在程序启动的时候判断程序是否已经安装，如果没有安装，那么开始启动安装程序：

```
// check OCRlib.apk is installed.
private boolean checkPackageInstalled() {
    boolean flag = PackageUtil.isInstalled(this, "com.xdja.zdsb");
    return flag;
}

// install APK
public boolean installApk(){
    if (checkPackageInstalled () ) {
        return true;
    }

    // copy file from asset fold.
    String filename = copyFileFromAsset();

    try {
```

```
        Runtime.getRuntime().exec("chmod 644 " + filename);
    } catch (IOException e) {
         e.printStackTrace();
         return false;
    }

    Intent intent = new Intent(Intent.ACTION_VIEW);
    intent.setDataAndType(Uri.fromFile(new File(filename)),
                 "application/vnd.android.package-archive");

    context.startActivity(intent);
    return true;
}

private String copyFileFromAsset() {

    String filename ="/data/data/{%packagename%}/files/ORClib.apk"

    InputStream in = null;
    FileOutputStream out = null;

    try {
        in = (InputStream)this
                  .getResources().getAssets().open(Security);
        out = this.openFileOutput(Security, this.MODE_PRIVATE);
        byte[] buffer = new byte[8192];
        int count = 0;

        while ((count = in.read(buffer)) > 0) {
            out.write(buffer, 0, count);
        }
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    } finally {
        in.close();
        out.close();
    }

    return filename;
}
```

认证接口调用

调用识别功能，是使用 Intent filter 来直接调用识别库中的 Activity， 调用后启动 Activity 并打开相机开始扫描，扫描后会弹出一个对话框，提示识别结果，如果结果没有错误，点击确认。如果识别错误，可以点击重新扫描来继续完成扫面。
识别正确后调用方会得到一个 Result 通过重载 onActivityResult 函数来获得识别结果数据。

调用相机扫描方式调用：

（注意：新升级开发的调用接口要求向上兼容以前的版本）

- 车牌识别调用

车牌识别调用的 action filter：
String CPSB = "com.xdja.zdsb.cpsb.action";
如果调用的时候不希望保存识别的图片，输入参数：（默认是保存图片的）
intent.putExtra("savePic", true); 否则参数为 false
调用方法如下：

```
private static final int CAR_PLATE_REQUEST = 1;
public void recognizeCarPlate() {

    Intent intent = new Intent();
    Toast.makeText(mContext,
         "请将车牌放置在边框内进行扫描效果更佳！",
         Toast.LENGTH_SHORT)
         .show();
    intent.setAction(CPSB);
    intent.putExtra("savePic", true);
    startActivityForResult(intent, CAR_PLATE_REQUEST);
}
```

识别完成后，会返回识别结果。 通过 onActivityResult 读取识别数据。
可以直接通过 data.getStringExtra("number"); 获得车牌号码
可以通过 data.getStringExtra("data"); 获得车牌颜色，号码等信息。

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

```
    if (resultCode == RESULT_OK) {
        switch (requestCode) {
            case CAR_PLATE_REQUEST:
                String number = null, result  = null;
                try{
                    number = data.getStringExtra("number");
                }catch (Exception e) {
                    e.printStackTrace();
                }
                result = data.getStringExtra("data");
                break;
            default:
                ;
            }
        }
        super.onActivityResult(requestCode, resultCode, data);
    }
```

- 身份证识别调用

身份证识别调用的 action filter：
String SFZSB = "com.xdja.zdsb.sfzsb.action";
如果调用的时候不希望保存识别的图片，输入参数：
intent.putExtra("savePic", true); 否则参数为 false
调用方法如下：

```
private static final int ID_CARD_REQUEST = 2;
public void recognizeCarPlate() {

    Intent intent = new Intent();
    Toast.makeText(mContext,
         "请将身份证放置在边框内进行扫描",
         Toast.LENGTH_SHORT)
         .show();
    intent.setAction(SFZSB);
    startActivityForResult(intent, ID_CARD_REQUEST);
}
```

识别完成后，会返回识别结果。 通过 onActivityResult 读取识别数据。
可以通过 data.getStringExtra("data"); 获得姓名，身份证号码等信息。
代码如下：

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

    if (resultCode == RESULT_OK) {
        switch (requestCode) {
            case ID_CARD_REQUEST:
                String result  = null;
                result = data.getStringExtra("data");
                break;
            default:
               ;
        }
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

- 驾驶证识别调用

驾驶证识别调用的 action filter：
String JSZSB = "com.xdja.zdsb.jszsb.action";
如果调用的时候不希望保存识别的图片，输入参数：
intent.putExtra("savePic", true); 否则参数为 false
调用方法如下：

```
private static final int DRIVER_LICENSE_REQUEST = 3;
public void recognizeCarPlate() {

    Intent intent = new Intent();
    Toast.makeText(mContext,
         "请将身份证放置在边框内进行扫描",
         Toast.LENGTH_SHORT)
         .show();
    intent.setAction(JSZSB );
    startActivityForResult(intent, DRIVER_LICENSE_REQUEST);
}
```

识别完成后，会返回识别结果。 通过 onActivityResult 读取识别数据。
可以通过 data.getStringExtra("data"); 获得姓名，号码等信息。
代码如下：

```
@Override
protected void onActivityResult(int requestCode,
```

```
            int resultCode, Intent data) {
    if (resultCode == RESULT_OK) {
        switch (requestCode) {
            case DRIVER_LICENSE_REQUEST:
                String result  = null;
                result = data.getStringExtra("data");
                break;
            default:
               ;
        }
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

- 护照识别调用

护照识别调用的 action filter：
String PASSWORTSB = "com.xdja.zdsb.passport.action";
如果调用的时候不希望保存识别的图片，输入参数：
intent.putExtra("savePic", true); 否则参数为 false
调用方法如下：

```
private static final int PASSPORT_REQUEST = 4;
public void recognizeCarPlate() {

    Intent intent = new Intent();
    Toast.makeText(mContext,
         "请将身份证放置在边框内进行扫描",
         Toast.LENGTH_SHORT)
         .show();
    intent.setAction(PASSWORTSB);
    startActivityForResult(intent, PASSPORT_REQUEST);
}
```

识别完成后，会返回识别结果。 通过 onActivityResult 读取识别数据。
可以通过 data.getStringExtra("data"); 获得姓名，号码等信息。
代码如下：

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

```
    if (resultCode == RESULT_OK) {
        switch (requestCode) {
            case PASSPORT_REQUEST:
                String result  = null;
                result = data.getStringExtra("data");
                break;
            default:
               ;
        }
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

通过图片地址传输识别调用

- 图片调用方式：

图片识别调用的 action filter：
String PICTURE_SB = "com.xdja.zdsb.picture.action";

调用方法如下：

```
Private static final int PICTURE_REQUEST = 5;

// picture type
Private static final int CAR_PLATE = 1;
Private static final int ID_CARD = 2;
Private static final int DRIVER_LICENSE = 3;
Private static final int PASSPORT = 4;

public void recognizePicture(int ID_type) {
    Intent intent = new Intent();
     intent.setAction(PICTURE_SB)
     intent.putExtra("picture_type", ID_type)
     intent.putExtra("path", filepath)
     startActivityForResult(intent, PICTURE_REQUEST);
}
```

识别完成后，返回识别结果。 根据传入参数的不同，获得的结果也不相同。可以通过 data.getIntExstra(“type”)获得识别类型。 获得的结果数字代表的意义：
0： 识别不成功。
1： 识别车牌
2： 识别身份证
3： 识别驾照
4： 识别护照
获得识别的类型后，然后再调用 data.getStringExstra(“data”) 获得识别的具体数据。数据的每一项用逗号(，)隔开然后根据需要自己解析。
data.getStringExstra(“keyNumber”) 可以直接获得证件号码。

比如身份证的数据：
Data = “姓名: 王五,性别:男,民族:汉,出生:1998-08-08,住址: 北京市东城区长安大街 50 号,公民身份号码: 110101199808086868,

解析代码：

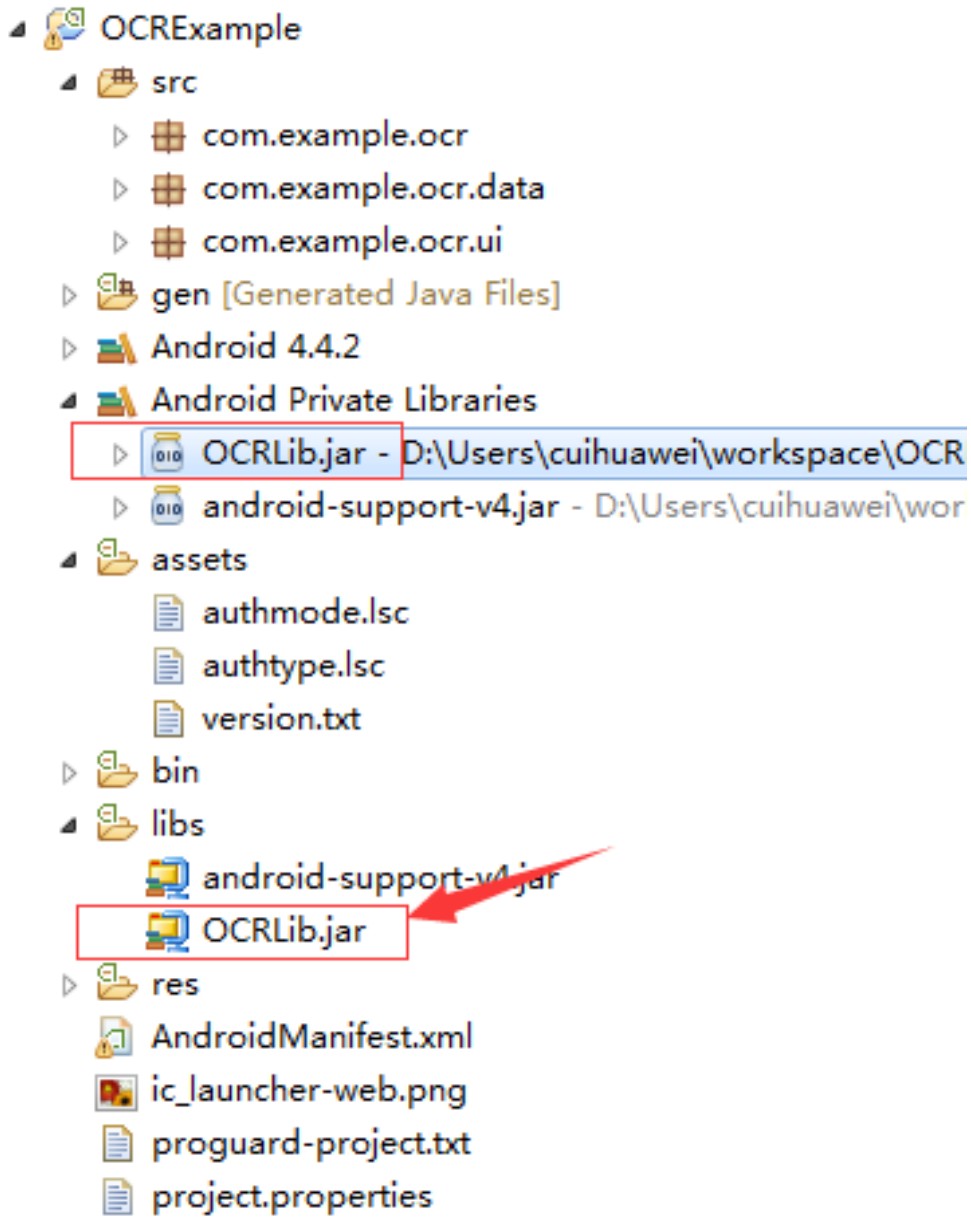
```
@Override
protected void onActivityResult(int requestCode,
            int resultCode, Intent data) {

    if (resultCode == RESULT_OK) {
        switch (requestCode) {
            case PICTURE_REQUEST:
                int  result  = null;
                result = data.getIntExtra("type",0);
                String info = data.getIntExtra("data",0)
                // ……
                break;
            default:
               ;
        }
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

二． 封装 Jar 包集成方式

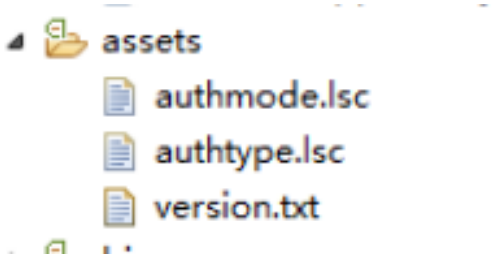
Jar 包的封装目录 OCRLib.jar

1. 新建一个 Android 项目，比如 sampleOCR ，建立后，增加识别库文件， 项目 project 的 properties->android 里 library 点击 Add, 注意： 不是 Build path。

   添加项目后，项目的代码结构如下：



2. 将需要的授权认证文件放入到 assets 目录下，以供认证时候读取文件。



3. AndroidManifest.xml 文件修改；

因为需要直接调用带有 activity 的界面去识别图片，所以需要在 AndroidManifest.xml 文件中增加 activity 的注册声明，否则的话，在调用的时候会找不到类导致识别失败。识别使用到的 activity 类有几个：车牌识别界面的 activity； 身份证识别界面 activity； 驾照识别界面的 activity； 护照识别界面。

声明代码如下：

```
<activity
    android:name="com.xdja.zdsb.view.PassportActivity"
    android:label="@string/app_name"
    android:screenOrientation="landscape" >
    <intent-filter>
        <action android:name
            ="com.xdja.zdsb.passport.action" />
        <category android:name
            ="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>

<activity
    android:name
       ="com.xdja.zdsb.view.DriverLicenseActivity"
    android:label="@string/app_name"
    android:screenOrientation="landscape" >
    <intent-filter>
        <action android:name
            ="com.xdja.zdsb.jszsb.action" />
        <category android:name
            ="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity
    android:name="com.xdja.zdsb.view.CameraActivity"
    android:label="@string/app_name"
```

```
                android:screenOrientation="landscape" >
            <intent-filter>
                <action android:name
                    ="com.xdja.zdsb.sfzsb.action" />
                <category android:name
                    ="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>

        <activity
            android:name="com.xdja.zdsb.view.CarPlateActivity"
            android:screenOrientation="portrait"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name
                    ="com.xdja.zdsb.cpsb.action" />
                <category android:name
                    ="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>

        <activity
            android:name="wintone.idcard.android.IDCardBean"
            android:configChanges="keyboardHidden|orientation"
            android:screenOrientation="landscape" >
            <intent-filter>
                <action android:name="wintone.idcard" />
                <category android:name
                    ="android.intent.category.DEFAULT" />
            </intent-filter>
    </activity>
```

同时，需要添加两个服务，一个是认证授权的服务，一个是图形识别的服务。注册声明的代码如下：

```
    <!-- win_tone service  -->
    <service
        android:name="com.wintone.plateid.AuthService"
        android:enabled="true">
        <intent-filter>
            <action android:name
                ="wintone.plateid.authservice"/>
```

```
            </intent-filter>
        </service>

        <!-- vehicle plate -->
        <service
            android:name="com.wintone.plateid.RecogService"
            android:enabled="true" >
            <intent-filter>
                <action android:name
                    ="wintone.plateid.recogService" />
            </intent-filter>
        </service>

        <!-- id card and license -->
        <service
            android:name="wintone.idcard.android.RecogService"
            android:enabled="true" >
            <intent-filter>
                <action android:name
                    ="wintone.idcard.recogService" />
            </intent-filter>
        </service>
```

这些工作做好以后，就可以使用接口调用来识别；识别授权认证是通过在识别之前，已经调用认证授权文件，所以不用再显示的调用授权认证来通过认证了； 根据上面的 activity 的声明也能看出来，通过 activity 来调用识别界面开始识别。

车牌识别调用：

```
Intent intent = new Intent(this, CarPlateActivity.class);
startActivityForResult(intent, CAR_PLATE_REQUEST);
```

身份证识别调用：

```
Intent intent = new Intent(this, CameraActivity.class);
startActivityForResult(intent, ID_CARD_REQUEST);
```

驾驶证识别调用：

```
Intent intent = new Intent(this, DriverLicenseActivity.class);
startActivityForResult(intent, DRIVER_LICENSE_REQUEST);
```

护照识别调用：

```
Intent intent = new Intent(this, PassportActivity.class);
startActivityForResult(intent, PASSPORT_REQUEST);
```

用户也可以通过传入照片，来得到需要识别的结果。调用这个借口的时候，需要传入另外的参数，传入的参数包括图片文件的地址和目标识别类型，使用 PictureRecognize 类来进行识别，由于识别的过程比较慢，这里使用的是异步方式来识别的，需要传入一个接口函数用来回调识别的结果。

接口函数定义如下：

```
public interface RecognizerInterface {
    public void onRecognizeSucceed();
    public void onRecognizeFailed();
}
```

PictureRecognize 类中的 doRecognize 函数定义：

```
public String doRecognize(String pictruePath, int IdType,
              RecognizerInterface recognizerInterface)
```

识别调用如下：

```
String picPath = Environment.getExternalStorageDirectory()
            .getAbsolutePath() + "{%pacture/path/name.jpg%}";

// picture type
Private static final int CAR_PLATE = 1;
Private static final int ID_CARD = 2;
Private static final int DRIVER_LICENSE = 3;
Private static final int PASSPORT = 4;

PictureRecognize pictureRecognize = new PictureRecognize();
String result = pictureRecognize.doRecognize(picPath, ID_CARD,
               recognizerInterface)
```

在调用的地方生成一个实例，用来回调识别后的结果。
代码如下：

```
    RecognizerInterface recognizerInterface
           = new RecognizerInterface(){
```

```
        @Override
        public void onRecognizeSucceed() {
            // TODO 识别成功，继续做接下来的工作
        }

        @Override
        public void onRecognizeFailed() {
            // TODO 识别失败，根据情况提示用户或者继续识别
        }
    };
```

三.　原生 Jar 包集成方式

（参考文通科技的 android 平台开发文档。）