**Android ASR SDK 集成进入helloworld 项目**

# 简介：

本文将集成ActivityMiniRecog至官方android合成demo项目。

android合成demo，可以将文本合成为音频并且播放。

我们添加的功能为：识别成功后，将识别出来的文本，合成之后播放出来。

参考文档：<https://cloud.baidu.com/doc/SPEECH/ASR-Android-SDK.html#.E9.9B.86.E6.88.90.E6.8C.87.E5.8D.97>

前提： 将**官方demo**的源代码通过android studio在android真机上运行起来并且**测试成功**。Demo暂不支持模拟器。

目标用户：Android初学者或者想立即集成的用户。

|  |  |
| --- | --- |
| Android Studio | 3.0.1 |
| 官方Demo | 3.0.6 |
|  |  |
| 测试机型 | OPPO R7 - ColorOS 1.4  Android 4.3 |

# 官网新建一个应用

从合成demo中，我们得知项目的**包名**是com.baidu.tts.sample，定义在build.gradle中。

在官网中新建一个应用，开通语音识别和语音合成能力。填写包名为com.baidu.tts.sample。也可以使用本文示例的应用，测试成功上线前替换。

**APP\_ID** **10455099**

**API\_KEY** **rKCHBLmYiFPuCQTS0HttLbUD**

**SECRET\_KEY 037dc446820ec143d1628c20146b9d34**

包名Package name：**com.baidu.tts.sample**

修改合成demo的appId appKey secretKey，并且**测试是否有效。**

**public class** SynthActivity **extends** BaseActivity **implements** View.OnClickListener {  
  
 *// ================== 初始化参数设置开始 ==========================  
 /\*\*  
 \* 发布时请替换成自己申请的appId appKey 和 secretKey。注意如果需要离线合成功能,请在您申请的应用中填写包名。  
 \* 本demo的包名是com.baidu.tts.sample，定义在build.gradle中。  
 \*/* **protected** String **appId** = **"10455099"**;  
  
 **protected** String **appKey** = **"rKCHBLmYiFPuCQTS0HttLbUD"**;  
  
 **protected** String **secretKey** = **"037dc446820ec143d1628c20146b9d34"**;



# 复制 AndroidManifest.xml

从官方demo的AndroidManifest.xml中找到如下信息，然后复制到您自己的同名文件中。

此处需要您复制a)权限 b) 官网申请的应用信息 c) SDK的Service



紫色的请替换成您自己的申请的应用信息。

红色是您需要的添加的部分

# 识别初始化，复制识别jar库

在 ActivityRecog中可以发现如下初始化代码：



我们暂时不需要离线的，在SynthActivity中添加如下代码：

@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **mShowText**.setText(**DESC**);  
  
 initialButtons(); *// 配置onclick* initialTts(); *// 初始化TTS引擎* **initialAsr();** *// 初始化ASR引擎*}  
**private** MyRecognizer **myRecognizer**;

**private void** initialAsr(){

IRecogListener listener = null;  
 toPrint(**"识别引擎初始化开始"**);  
 **myRecognizer** = **new** MyRecognizer(**this**, listener);  
 toPrint(**"识别引擎初始化结束"**);

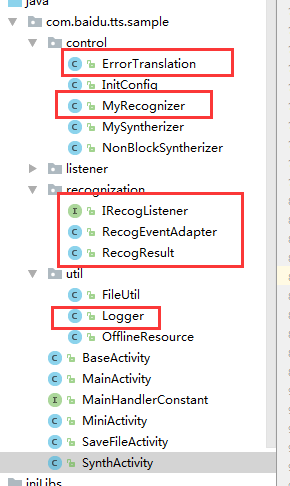
}

@Override  
**protected void** onDestroy() {  
 **synthesizer**.release();  
 Log.*i*(**TAG**, **"释放资源成功"**);  
  
 **myRecognizer**.release(); //销毁资源  
 Log.*i*(**TAG**, **"onDestory"**);  
 **super**.onDestroy();  
}

这个时候MyRecognizer类没有引用，识别的jar包也没有。

将识别的app\libs\bdasr\_V3\_20171108\_9800a2a.jar复制到合成的同名目录。

复制缺少的JAVA类，解决编译问题





**注意此时底层SO库还没有集成，开始识别的时候会发生报错。**

# 集成，准备识别。

我们需要集成的逻辑是识别出文字后然后合成音频播放。

因此第一步是识别出文字，第二步在识别的成功的回调中播放音频。

先准备一个回调Listener：extends StatusRecogListener

**public class** TtsRecogListener **extends** StatusRecogListener {  
 **private** MySyntherizer **synthesizer**;  
  
 **private final static** String ***TAG*** = **"TtsRecogListener"**;  
  
 **public** TtsRecogListener(MySyntherizer synthesizer) {  
 **this**.**synthesizer** = synthesizer;  
 }  
  
 @Override  
 **public void** onAsrFinalResult(String[] results, RecogResult recogResult) {  
 **super**.onAsrFinalResult(results, recogResult);  
 String msg = **"识别成功："** + results[0];  
 Log.*i*(***TAG***, msg);  
 }  
  
 @Override  
 **public void** onAsrFinishError(**int** errorCode, **int** subErrorCode, String errorMessage, String descMessage, RecogResult recogResult) {  
 **super**.onAsrFinishError(errorCode, subErrorCode, errorMessage, descMessage, recogResult);  
 String msg = **"错误码是："** + errorCode;  
 Log.*i*(***TAG***, msg);  
 }  
}

还记得之前的IRecogListener listener = null;么？

**private** MyRecognizer **myRecognizer**;  
  
**private void** initialAsr(){  
 IRecogListener listener =**new** TtsRecogListener(**synthesizer**);  
 toPrint(**"识别引擎初始化开始"**);  
 **myRecognizer** = **new** MyRecognizer(**this**, listener);  
 toPrint(**"识别引擎初始化结束"**);  
}

把合成demo中的“使用说明”按钮复用下，改为”开始识别”按钮

**case** R.id.***help***: *// 启动另一个activity  
 //mShowText.setText(DESC);* startAsr();  
 **break**;

我们看见识别demo中是这样开始识别的：myRecognizer.start(params);

**private void** startAsr(){  
 Map<String, Object> params = **new** TreeMap<String, Object>();  
 **myRecognizer**.start(params);  
 toPrint(**"请开始说话"**);  
 Toast.*makeText*(**this**, **"请开始说话"**, Toast.***LENGTH\_LONG***);  
}

点击“使用说明”按钮，会发现如下保存。（还记得底层so文件没有复制么？）

{"sub\_error":5001,"error":5,"origin\_result":{"sub\_error":5001,"error":5,"sn":"","desc":"Can not load so library"},"desc":"Can not load so library"}

# 复制jni库的so文件

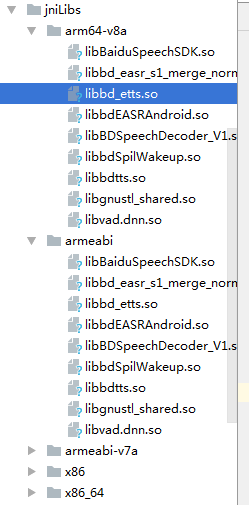
**由于缺少so文件**，点击开始后，有如下报错：

:{"sub\_error":5001,"error":5,"origin\_result":{"sub\_error":5001,"error":5,"sn":"","desc":"Can not load so library"},"desc":"Can not load so library"}

我们继续修改项目：

**复制官方demo中 app\src\main\jniLibs 至项目的同名目录**，如果对android studio不熟悉，这一步可以在windows 资源管理器操作

由于原先合成demo项目已经有了5个架构目录，因此此时只要合并即可。



*如果假设，合成demo项目有6个架构目录，然而合成只有5个，那么两者目录取交集，最终只留下同名的5个。如果假设，合成demo项目只有一个armeabi目录，那么删除识别项目额外的4个架构目录，最终只留下同名的armeabi目录。*

# 测试识别及自定义参数：

启动app，正常情况可以看见

**public void** onAsrFinalResult(String[] results, RecogResult recogResult) {  
 **super**.onAsrFinalResult(results, recogResult);  
 String msg = **"识别成功："** + results[0];  
 Log.*i*(***TAG***, msg);  
 }

这个方法被执行。

打开demo，这里以“在线识别”界面为例。 进入界面后，点击“设置”按钮。

经过一轮测试，准备选择“保存音频文件”及VAD时长2000ms作为在线识别的参数。



点击开始录音后，可以看见如下输入参数：



这个json在logcat中也可以找到。

“vad.endpoint-timeout” 是定义在SpeechConstant中的常量。

项目中填写同样的参数，生成一样的json。

**private void** startAsr(){  
 Map<String, Object> params = **new** TreeMap<String, Object>();  
 params.put(SpeechConstant.***VAD\_ENDPOINT\_TIMEOUT***,2000);  
 params.put(SpeechConstant.***ACCEPT\_AUDIO\_VOLUME***, **false**);  
 params.put(SpeechConstant.***OUT\_FILE***, **"/storage/emulated/0/baiduASR/outfile.pcm"**);  
 params.put(SpeechConstant.***ACCEPT\_AUDIO\_DATA***, **true**);  
 params.put(SpeechConstant.***DISABLE\_PUNCTUATION***,**false**);  
  
 **myRecognizer**.start(params);  
 toPrint(**"请开始说话"**);  
 Toast.*makeText*(**this**, **"请开始说话"**, Toast.***LENGTH\_LONG***);  
}

# 集成合成播放

我们修改下TtsRecogListener中的回调

@Override  
**public void** onAsrFinalResult(String[] results, RecogResult recogResult) {  
 **super**.onAsrFinalResult(results, recogResult);  
 String msg = **"识别成功："** + results[0];  
  **synthesizer.speak(msg);**  
 Log.*i*(***TAG***, msg);  
}  
  
@Override  
**public void** onAsrFinishError(**int** errorCode, **int** subErrorCode, String errorMessage, String descMessage, RecogResult recogResult) {  
 **super**.onAsrFinishError(errorCode, subErrorCode, errorMessage, descMessage, recogResult);  
 String msg = **"错误码是："** + errorCode;  
 **synthesizer.speak(msg);**  
 Log.*i*(***TAG***, msg);  
}

正常的话，会听见合成的女声。

# 代码下载