百度语音 SDK Android 版
开发手册

v1.5

北京百度网讯科技有限公司

修改记录

版本号	修改内容简介	修改日期
1.1	第一版	2013-10-05
1.2	支持语义理解 支持外部音频识别 支持音频数据回调	2013-11-15
1.3	支持语种设置 错误码数值优化	2014-01-20
1.4	识别对话框引导语 优化 speechMode 接口，使用 setProp 设置	2014-03-20
1.4.1	模式合并	2014-05-07
1.5	增加联系人上传	2014-06-10

目录

1 概念解释

对本文中将提到的名词约定如下：

语音识别：也被称为自动语音识别（Automatic Speech Recognition，ASR），其目标是将人类的语音中的词汇内容转换为计算机可读的输入，例如按键、二进制编码或者字符序列

自然语言理解：Natural Language Understanding 俗称人机对话。人工智能的分支学科。研究用电子计算机模拟人的语言交际过程，使计算机能理解和运用人类社会的自然语言如汉语、英语等，实现人机之间的自然语言通信，以代替人的部分脑力劳动，包括查询资料、解答问题、摘录文献、汇编资料以及一切有关自然语言信息的加工处理。

BDVRClient：语音识别 SDK 的简称，详见下条。

语音 SDK：即本开发包，文中简称为 BDVRClient。BDVRClient 是一个封装了语音采集、处理、网络收发等功能的语音识别解决方案。使用 BDVRClient 可以快速在应用程序中集成语音识别功能。

应用程序：在开发中使用了 BDVRClient，具有语音识别功能的产品线产品。

开发者：想使用 BDVRClient 到应用程序中，正在阅读本文档的开发人员。

2 简介

百度语音 SDK Android 版（以下简称 BDVRClient）是运行在 Android 平台的一体化语音识别解决方案，以 JAR 包 + 动态链接库的形式发布。基于该方案，开发者可以轻松的构建功能完备、交互性强的语音识别应用程序。

2.1 功能介绍

BDVRClient 支持下列功能：

语音识别控件：集成提示音、音量反馈动效整套交互的对话框控件，方便开发者快速集成；

基本功能：录音，语音数据处理，端点检测、网络通讯和状态通知，返回识别结果；

播放提示音：在录音前后播放提示音，优化用户体验；

监听语音能量：实时反馈用户当前说话声音能量强度；

语义理解：将语音识别成领域相关的语义结果。

本文档适用于对 Android 应用开发有基本了解的开发人员。

2.2 兼容性

系统：支持 Android 2.2（API Level 8）及以上系统。需要开发者通过 minSdkVersion 来保证支持系统的检测。

机型：手机和平板皆可

构架：支持 ARM 平台、x86 平台

硬件要求：要求设备上有麦克风。

网络：支持 WIFI 及移动网络，移动网络支持使用 NET 网关及 WAP 网关（CMWAP、CTWAP、UNIWAP、3GWAP）。

2.3 开发包说明

表格 1 开发包说明表

文件/文件夹名	说明
/libs/	语音识别 SDK Lib 库，包括各平台的 SO 库及 Jar。SO 库开发者可以按需集成
/res/raw	语音识别对话框音效文件，如果不使用对话框可以不集成。
/docs	JAVADOC
百度语音 SDK Android 版开发手册	本手册
百度语义理解协议	语义理解意图表示说明
/VoiceRecognitionDemo/	Demo 工程

2.4 总体框图

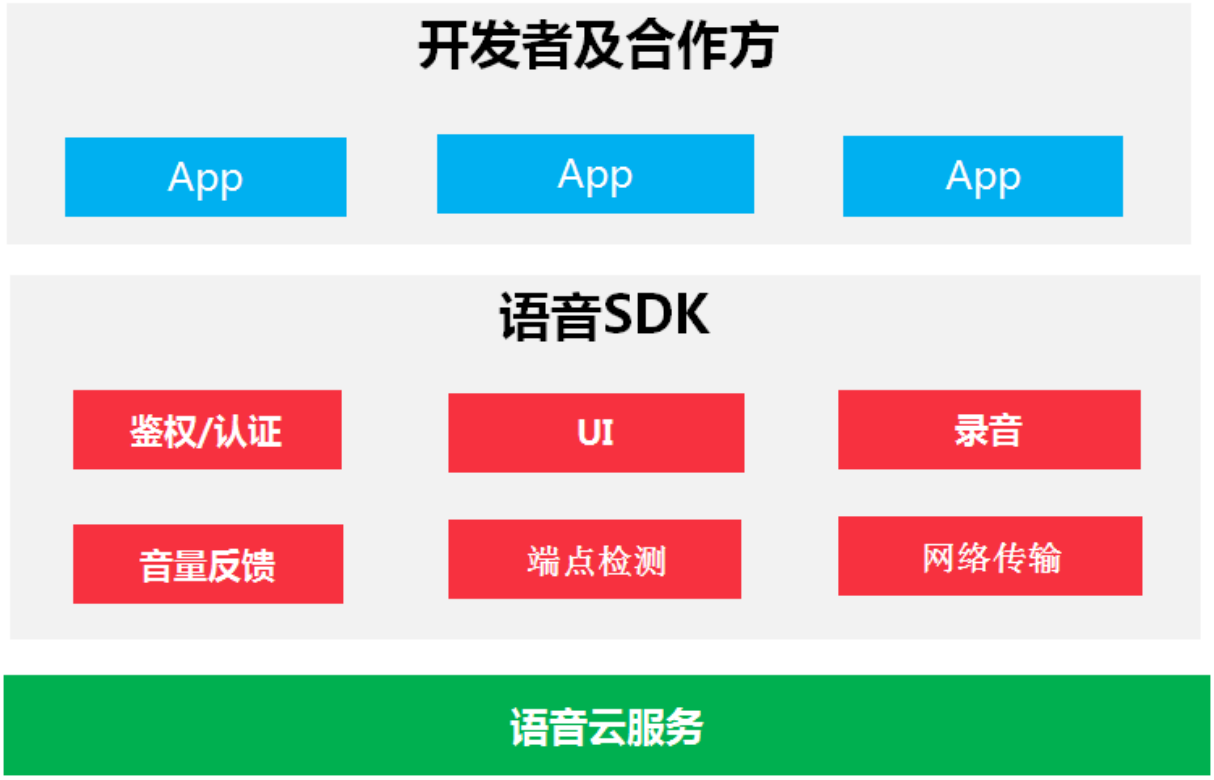


图 1BDVRClient 的总体使用框图

3 集成指南

本章将进行 Step-By-Step 的讲解，如何快速的集成 BDVRClient 到现有应用中。一个完整的 Demo 请参考开发包中的示例程序 VoiceRecognitionDemo。

3.1 注册开放平台及创建应用

开发者需要在百度开发者中心（http://developer.baidu.com/）注册帐号，并创建一个应用。

目前开发者中心管理控制台正处于版本更新时期，因此有“老版管理控制台”与“新版管理控制台”的区分。推荐进入“新版管理控制台”创建应用。如图 2 红框所示，点击开发者中心首页右上角“消息”旁边的“新版管理控制台”进入新版控制台。

图 2 新版本管理控制台入口

创建应用：

（1） 点击“创建应用”

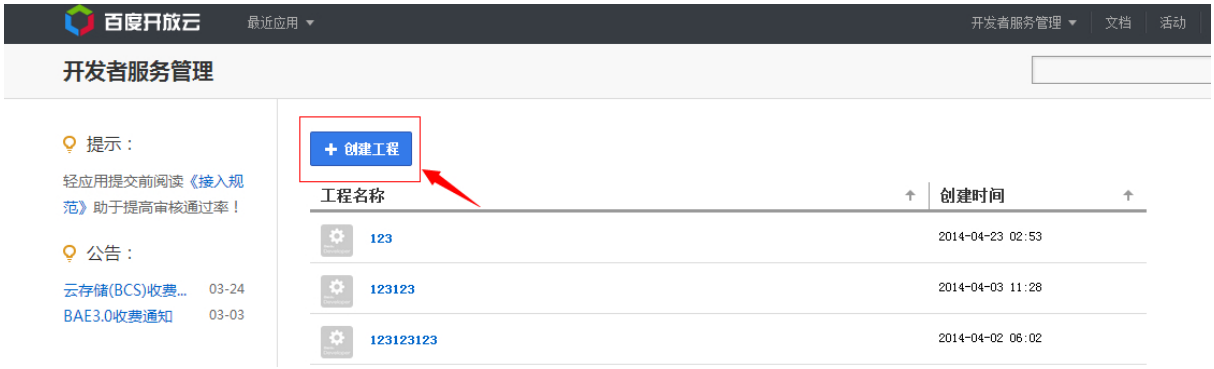


图 2 创建应用

（2） 填写应用名称以及应用类型

图 3 添加应用信息

（3） 应用创建成功。在应用基本信息页面，获取应用的 AK 与 SK。

图 4 获取应用 AK/SK

3.2 申请开启语音识别服务

在使用语音识别服务之前，需要申请开启服务。您的申请提交后，后台工作人员会根据您的申请内容，进行批准或拒绝操作。申请的结果，会通过开发者中心的消息中心通知您。

语音识别服务申请

进入应用，选择“媒体云”—>“语音识别”，点击“申请开启服务”。根据提示填写使用场景、申请理由，提交申请。

申请通过后，即拥有服务的使用权限。

3.3 添加 BDVRClient 到工程

将开发包中的 Libs 目录整体 Copy 到工程目录，Libs 目录包括了各平台的 SO 库，开发者视应用需要可以进行删减。galaxy.jar 是百度 Android 公共基础库，如果项目中还集成了其它百度 SDK，如 Push SDK，在打包过程中出现类似如下的错误信息：

```
[2013-10-22 11:02:57 - Dex Loader] Unable to execute dex: Multiple dex files define Lcom/baidu/android/common/logging/Configuration;
[2013-10-22 11:02:57 - VoiceRecognitionDemo] Conversion to Dalvik format failed: Unable to execute dex: Multiple dex files define Lcom/baidu/android/common/logging/Configuration;
```

请将此 Jar 包移除。

将开发包中的 res 目录整体 Copy 到工程目录，此文件包括语音识别对话框的音效文件，如果不使用可以忽略。

如果 Eclipse ADT 版本插件低于 17，需要手工添加依赖库，添加方法为：Project => Properties => Java Build Path => Libraries => Add JAR ...

3.4 权限声明

BDVRClient 需要权限如下：

名称	用途
android.permission.RECORD_AUDIO	允许应用使用麦克风录音
android.permission.INTERNET	允许应用联网，发送语音数据至服务器，获得识别结果。
android.permission.ACCESS_NETWORK_STATE	获取当前的网络状态，优化录音参数及网络参数。
android.permission.READ_PHONE_STATE	获取用户手机的 IMEI，用来唯一的标识用户。

需要在 AndroidManifest.xml 文件，增加以上四个权限：

```
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

3.5 Proguard 配置

如果应用配置了代码混淆，需要在 Proguard 配置文件增加以下参数：

```
-keep class com.baidu.android.**{*;}
-keep class com.baidu.voicerecognition.android.**{*;}
```

4 语音识别

百度语音 SDK 提供英语、粤语、普通话识别，支持输入、搜索、地图、音乐等不同的应用场景，搜索适合短 Query 的输入场景，在识别过程中会将语气词过滤，尾点检测会更灵敏。输入适合短信输入等长句场景，尾点检测会迟钝一些，可以进行连续多句识别。是否需要标点可配置。开发者可以通过调用语音识别 UI 快速集成，也可以使用底层 API 接口自行设计语音交互。

SDK 还支持语义理解能力，可以将用户的语音直接转换成需求意图。语义具有领域性特征，不属于任何领域的语义是不存在的。同样的语言，在不同的领域中所代表的含义可能截然不同。语义理解就是把语言在特定领域所代表语义通过计算机可处理的表示方式理解出来。具体支持的领域及数据格式请参考《百度语义理解协议》。

如果开启了语义解析能力，结果将做为 JSON 字符串放至在候选数组的首个元素。JSON 对象的结构如下：

字段名	数据类型	描述
item	JSONArray<String>	为 String 类型的 JSONArray，存储语音识别结果
json_res	String	语义解析结果，为 Json 格式的 String。

json_res 转换成 Json 对象结构如下，详细参考《百度语义理解协议》

字段名	数据类型	描述
raw_text	string	原始文本
parsed_text	string	分词结果
results	JSONArray	语义理解意图数组，可能为空

4.1 语音识别控件

BaiduASRDigitalDialog 语音识别控件，提供了整套语音交互、提示音、音量反馈、动效反馈。开发者初始化一个 BaiduASRDigitalDialog 对象，并设置相关参数及结果回调，调用 Show()方法就可以弹出对话框开始识别，识别结束后会在回调中得到识别结果。

4.1.1 创建对话框

```
Bundle params = new Bundle();
//设置开放 API Key
params.putString(BaiduASRDigitalDialog.PARAM_API_KEY,your_API_KEY);
//设置开放平台 Secret Key
params.putString(BaiduASRDigitalDialog.PARAM_SECRET_KEY,your_SECRET_KEY);
//设置识别领域：搜索、输入、地图、音乐……，可选。默认为输入。
params.putInt( BaiduASRDigitalDialog.PARAM_PROP,VoiceRecognitionConfig.PROP
_INPUT);
//设置语种类型：中文普通话，中文粤语，英文，可选。默认为中文普通话
params.putString( BaiduASRDigitalDialog.PARAM_LANGUAGE,VoiceRecognitionConf
ig.LANGUAGE_CHINESE);
//如果需要语义解析，设置下方参数。领域为输入不支持
params.putBoolean( BaiduASRDigitalDialog.PARAM_NLU_ENABLE,true);
// 设置对话框主题，可选。BaiduASRDigitalDialog 提供了蓝、暗、红、绿、橙四中颜色，每种颜
色又分亮、暗两种色调。共 8 种主题，开发者可以按需选择，取值参考 BaiduASRDigitalDialog 中
前缀为 THEME_的常量。默认为亮蓝色
params.putInt(BaiduASRDigitalDialog.PARAM_DIALOG_THEME,BaiduASRDigitalDialo
g.THEME_RED_DEEPBG);
mDialog = new BaiduASRDigitalDialog(context, params);
```

4.1.2 参数列表

识别对话框支持的参数定义在 BaiduASRDigitalDialog 中以 PARAM_前缀的常量。列表如下：

参数名称	参数类型	默认值	描述
PARAM_API_KEY	string		开放平台认证 API_key
PARAM_SECRET_KEY	string		开放平台认证 Secret_key
PARAM_LANGUAGE	string	LANGUAGE_CHINESE	语种，取值定义在 VoiceRecognitionConfig 类中前缀为 LANGUAGE_的常量
PARAM_PARTIAL_RESULTS	boolean	true	连续上屏
PARAM_NLU_ENABLE	boolean	false	是否语义解析。Prop 为输入时暂不支持语义，请显示指定为其它领域。
PARAM_NLU_PARAMS	string		预留语义解析参数
PARAM_PROP	int	PROP_INPUT	领域参数，定义在 VoiceRecognitionConfig 类中前缀为 PROP_

			的常量
PARAM_PORMPT_TEXT	string	“请说话”	对话框提示语
PARAM_PROMPT_SOUND_ENABLE	boolean	true	提示音，需要集成 SDK 包 Raw 文件夹的资源
PARAM_DIALOG_THEME	int	THEME_BLUE_LIGHTBG	样式。定义在前缀为 THEME_的常量中
PARAM_TIPS	String[]		引导语列表
PARAM_SHOW_TIPS_ON_START	boolean	false	对话框弹出时首先显示引导语列表
PARAM_SHOW_TIP	boolean	false	识别启动 3 秒未检测到语音，随机出现一条引导语
PARAM_SHOW_HELP_ON_SILENT	boolean	false	静音超时后将“取消”按钮替换为“帮助”

4.1.3 设置回调

```
mRecognitionListener = new DialogRecognitionListener() {

    @Override
    public void onResults(Bundle results) {
        //在 Results 中获取 Key 为 DialogRecognitionListener .RESULTS_RECOGNITION
的 StringArrayList，可能为空。获取到识别结果后执行相应的业务逻辑即可，此回调会在主线程调
用。
        ArrayList<String> rs = results != null ? results
        .getStringArrayList(RESULTS_RECOGNITION) : null;
        if (rs != null) {
            //此处处理识别结果，识别结果可能有多个，按置信度从高到低排列，第一个元素是置信
度最高的结果。
        }
    }
};
mDialog.setDialogRecognitionListener(mRecognitionListener);
```

注意，如果用户在对话中取消了识别操作，此回调不会被执行。

4.1.4 启动识别

调用 mDialog.show();方法开始识别，建议开发者在当前 Activity 复用一个识别对话框。切记在 OnDestroy 方法调用对话框 Dismiss，防止 Dialog 泄露。

4.2 API 方式

上一章描述了如何使用 BaiduASRDigitalDialog 进行语音识别，此种方式提供了统一的简便易用的交互，如果此种交互无法满足开发者的需求，开发者可以直接使用语音识别 API 定制自己的语音交互。

下面将从初始化、参数设置、状态处理、状态控件、音量反馈、结果处理等方面逐步介绍

4.2.1 实现识别监听器

为了不阻塞 UI，BDVRClient 将异步的工作，因此开发者在使用 BDVRClient 时需要实现一个识别状态监听器，以便应用获取 BDVRClient 识别过程事件通知以及识别结果。此监听会在主线程被调用，用户可以直接操作 UI，同时需要注意不要进行耗时操作，以防引起 ANR。

实现 `VoiceClientStatusChangeListener` 开发者需要重写三个方法，这三个回调方法分别用来接收 BDVRClient 在后台网络请求发起和结束时的通知、识别进度的通知、以及出错时具体错误信息的通知。后面应用程序发起识别时需要将此监听器传入 BDVRClient 中。

4.2.1.1 onClientStatusChange

此方法提供识别过程的事件通知，事件类型参考在 VoiceRecognitionClient 中 CLIENT_STATUS_ 前缀的常量定义。常用事件如下表，详细参阅 API 文档

事件名	描述
CLIENT_STATUS_START_RECORDING	引擎准备就绪事件，开始录制音频。此时可以在 UI 上提示用户可以说话
CLIENT_STATUS_SPEECH_START	检测到用户语音起点事件
CLIENT_STATUS_AUDIO_DATA	有音频数据输出，音频数据为单声道，采样率依赖内部实现或者用户的设置
CLIENT_STATUS_SPEECH_END	检测到用户语音尾点事件，此时引擎会关闭录音设备不再接收用户语音。
CLIENT_STATUS_UPDATE_RESULTS	中间结果返回事件，用户说话过程中间识别结果返回，识别过程中此方法可能会被多次调用。需要注意，搜索返回结果类型为 List<String>，领域为输入返回结果类型为 List<List<Candidate>>。
CLIENT_STATUS_USER_CANCELED	用户取消事件
CLIENT_STATUS_FINISH	识别结束事件。返回最终识别结果，搜索模式类型为 List<String>，领域为输入类型为 List<List<Candidate>>。

4.2.1.2 onError

此方法会接收到识别异常回调。异常类别及错误码定义在 VoiceRecognitionClient 中。分为 ERROR_CLIENT、ERROR_RECORDER、ERROR_NETWORK、ERROR_SERVER，错误定义参见

ERROR_前缀的常量。

ErrorType	描述	ErrorCode	值	描述
ERROR_CLIENT	客户端异常	ERROR_CLIENT_UNKNOWN	0x20001	未知
		ERROR_CLIENT_NO_SPEECH	0x20002	没有说话
		ERROR_CLIENT_TOO_SHORT	0x20003	语音太短
		ERROR_CLIENT_TOO_LONG	0x20004	语音太长
		ERROR_CLIENT_JNI_EXCEPTION	0x20005	SO 异常
ERROR_RECORDER	录音设备异常	ERROR_RECORDER_UNAVAILABLE	0x30001	设备不可用
		ERROR_RECORDER_INTERCEPTED	0x30002	录音中断
ERROR_NETWORD	网络异常	ERROR_NETWORK_UNUSABLE	0x40001	网络不可用
		ERROR_NETWORK_CONNECT_ERROR	0x40002	连接失败
		ERROR_NETWORK_TIMEOUT	0x40003	网络超时
		ERROR_NETWORK_PARSE_ERROR	0x40004	数据解析失败
ERROR_SERVER		ERROR_SERVER_PARAMETER_ERROR	0x53001	参数错误
		ERROR_SERVER_BACKEND_ERROR	0x53002	内部错误
		ERROR_SERVER_RECOGNITION_ERROR	0x53003	识别错误
		ERROR_SERVER_INVALID_APP_NAME	0x53004	认证失败
		ERROR_SERVER_SPEECH_QUALITY_ERROR	0x53005	语音质量

				错误

4.2.1.3 示例代码

```
class MyVoiceRecogListener implements VoiceClientStatusChangeListener {

    @Override
    public void onClientStatusChange(int status, Object obj) {
        switch (status) {
        case VoiceRecognitionClient.CLIENT_STATUS_START_RECORDING:
            break;
        case VoiceRecognitionClient.CLIENT_STATUS_SPEECH_START:
            //检测到语音起点
            break;
         case VoiceRecognitionClient.CLIENT_STATUS_AUDIO_DATA:
            //有音频数据输出
            if (obj != null && obj instanceof byte[]) {
                //处理数据
            }
            break;
        case VoiceRecognitionClient.CLIENT_STATUS_SPEECH_END:
            //已经检测到语音终点，等待网络返回
            break;
        case VoiceRecognitionClient.CLIENT_STATUS_FINISH:
            //语音识别完成
            if(currentVoiceType == VOICE_TYPE_SEARCH){
                mStatusTextView.setText(R.string.finished);
                //obj 是一个 ArrayList<String>，里面有多个候选词
            }else if(currentVoiceType == VOICE_TYPE_INPUT){
                 //obj 是一个 List<List<Candidate>>，里面有多个候选词
             }
            break;
        case VoiceRecognitionClient.CLIENT_STATUS_UPDATE_RESULTS:
            //多句模式会有部分结果（一个分句）返回
            if(currentVoiceType == VOICE_TYPE_SEARCH){
                //obj 是一个 ArrayList<String>，里面有多个候选词
            }else if(currentVoiceType == VOICE_TYPE_INPUT){
                //obj 是一个 List<List<Candidate>>，里面有多个候选词
                List<List<Candidate>> result = (List<List<Candidate>>) obj;
            }
            break;
        case VoiceRecognitionClient.CLIENT_STATUS_USER_CANCELED:
            //通知用户已取消
```

```
            break;
        default:
            break;
        }
    }

    @Override
    public void onError(int errorType, int errorCode) {
        //errorType 为错误类型，errorCode 为错误码。此回调先于 onClientStatusChange
的 CLIENT_STATUS_ERROR
    }

    @Override
    public void onNetworkStatusChange(int status, Object obj) {
        // 这里不做任何操作不影响简单识别
    }

}
```

4.2.2 获得识别 BDVRClient 单例对象

BDVRClient 以单例模式运行，通过调用

```
mRecognitionClient =
VoiceRecognitionClient.getInstance(context.getApplicationContext());
mRecognitionClient.setTokenApis(your_API_KEY,
your_SECRET_KEY);
```

方法获得单例对象后，需要使用开放平台 ApiKey、SecretKey 初始化。

4.2.3 请求参数设置

每次识别请求通过 VoiceRecognitionConfig 设置参数。常用参数设置方法如下，详细参阅 API 文档。

方法	参数	描述
enableBeginSoundEffect	**int** soundResourceId 启动提示音资源 Id	设置开始提示音，soundResourceId 为放置在 Raw 文件夹的资源 Id。
enableEndSoundEffect	**int** soundResourceId 说话结束提示音资源 Id	检测到用户说话结束播报的提示音，非识别结束
setSampleRate	**int** rate 采样率	设置音频采样率， 通常建议开发者不指定采样频率，由 BDVRClient 自动根据当前网络环境选择采样频率。WiFi 环境下将使用 16kHz 采样，

		移动网络下将使用 8kHz 采样，来节省流量。参考常量定义 *SAMPLE_RATE_8K 8K 采样率* *SAMPLE_RATE_16K 16K 采样率*
setProp	**int** prop	开发者可以通过指定垂直分类来获取更精准的语音识别结果。 **注：**垂直分类目前支持地图，音乐，视频，APP，网址，开发者需要注意设定采样频率时只能在这五种垂直分类中选择。若指定其他分类，可能会影响识别结果的精度。参考 PROP_前缀的常量定义。
setUseDefaultAudioSource	**boolean** useDefaultSource	设置是否使用缺省的录音。 如果不使用，用户需要调用 `VoiceRecognitionClient` 对象的 feedAudioBuffer 方法为识别器提供语音数据
enableNLU		启用语义解析，只在搜索模式起作用
getSampleRate		获取当前识别采样率
setLanguage	**String** Language	设置语种。目前支持的语种有中文普通话（LANGUAGE_CHINESE）、中文粤语（LANGUAGE_CANTONSE）、英文（LANGUAGE_ENGLISH）。
setAccessToken	**String** token	直接设置 token
disablePunctuation		禁用标点符号

4.2.4 开始语音识别

通过调用 `VoiceRecognitionClient` 对象的 startVoiceRecognition()方法，即可开始语音识别。startVoiceRecognition()方法需要传入两个参数：识别监听器和识别参数信息。

例如，示例程序，在“开始”按钮的 OnClickListener 的 onClick 回调中，添加以下代码：

```
public void onClick(View v) {
    int ret=mRecognitionClient.startVoiceRecognition(new MyVoiceRecogListener
(), new new VoiceRecognitionConfig());
    if (code == VoiceRecognitionClient.START_WORK_RESULT_WORKING) {
        // 启动成功
    } else {
        //启动失败，错误码描述参见 VoiceRecognitionClient START_WORK_RESULT_前缀的
常量定义
    }
}
```

即可开始语音识别。如果启动失败，不会在回调内获得事件通知，需要直接做错误处理。

BDVRClient 在开始识别后，会启动录音、预处理、上传到服务器并获取识别结果。识别过程中监听器会接收到事件通知，更新相应 UI 即可。

4.2.5 用户自定义输入语音数据

用户可以调用 feedAudioBuffer 方法向识别器传送音频数据，确认 startVoiceRecognition 在此之前被调用，同时，在 startVoiceRecognition 方法的 config 参数中调用 setUseDefaultAudioSource(false)以及 setSampleRate(int rate)设置音频数据的采样率.

需要注意的是，传送的音频数据尾部应该有多余一秒的静音数据，如果没有足够的静音数据，应额外再传送一些 0x00 数据，以保证识别器能够完成尾点检测。

4.2.6 结束语音输入

进行语音识别时周围环境可能会嘈杂。可能会干扰 BDVRClient 判断用户语音结束的时机，造成 BDVRClient 一直工作在录音识别中。因此 BDVRClient 提供了一个接口供开发者来主动结束录音，进而更加保证语音识别的准确性。

在“说完了”按钮的 onClickListener 中，通过重写 onClick()方法：

```
@Override
public void onClick(View v) {
    mRecognitionClient.speakFinish();
}
```

即结束语音输入。

在结束语音输入后，BDVRClient 将停止录音。若 BDVRClient 当前已经检测到了语音则等待服务器返回识别结果；若实际没有检测到用户说话，会返回出错信息提示用户。

4.2.7 取消语音识别

对于用户来说，一次语音识别有可能是误操作进入，或者由于网络环境不好，一直处于等待中，用户需要主动停止本次语音识别。BDVRClient 提供了用户取消接口，来响应用户取消操作，并安全释放相关资源。

在“取消”按钮的 onClickListener 中，通过重写 onClick()方法：

```
@Override
public void onClick(View v) {
    mRecognitionClient.stopVoiceRecognition();
}
```

即可实现取消语音识别功能。

4.2.8 播放提示音

语音识别开始和完成时，BDVRClient 可以播放一个很短的如“嘀”的提示音来提醒用户可以说

话或说话完成。开发者可以通过接口设定 BDVRClient 将要播放的提示音，步骤如下：

添加音频文件到工程

请提前将提示音文件拷贝到/res/raw 文件夹下。支持 Wav、Mp3 格式文件。

启用识别开始提示音

```
VoiceRecognitionConfig config = new VoiceRecognitionConfig();
// 设置识别开始提示音
config.enableBeginSoundEffect(R.raw.bdspeech_recognition_start);
// 设置说话结束提示音
config.enableEndSoundEffect(R.raw.bdspeech_speech_end);
```

注：提示音仅用于提醒用户注意，长度以不干扰用户识别等待时间为宜，建议提示音的时长不超过 0.3s，开发者在选择音频文件时应当注意。

4.2.9 监听语音能量

开发者可能希望实时地获取当前用户说话的声音音量大小，以便在应用程序界面上显示（如跳动的音量条），或者提示用户当前说话声音太大或太小。当开发者需要获取当前音量功能时可通过如下方法实现：

获取语音能量

在识别过程中，应用程序通过调用：

```
mRecognitionClient.getCurrentDBLevelMeter();
```

来获得当前语音能量。获得的返回值是一个已经归一化后的 int 值，范围是 0-100，对应表示声音强度由小到大。

关闭语音能量计算

如果开发不需要音量反馈功能，可以通过 VoiceRecognitionConfig，关闭计算语音能量：

```
config.enableVoicePower(false);
```

注：开启语音能量计算功能后 BDVRClient 将自动不停地在底层计算当前音量，调用 `getCurrentDBLevelMeter()`方法并不会导致 BDVRClient 计算当前的语音能量，而是返回最近一次计算的语音能量值。开启音量计算功能后，即使不调用 `getCurrentDBLevelMeter()`方法，BDVRClient 自己也会不断计算更新。BDVRClient 计算音量的频率大约为每秒 30 次，开发者可根据应用程序需要决定是否开启音量计算功能以及向 BDVRClient 获取最近音量信息的频率。

4.3 自定义识别结果

开发者可以根据自己的需求上传相关数据，使得语音识别结果更加精确，当前支持联系人上传。

4.3.1 联系人上传

通过上传用户的联系人数据，可以使得人名的识别更加准确，联系人上传代码示例如下：

```
DataUploader dataUploader = new DataUploader(ApiDemoActivity.this);
dataUploader.setApiKey("yourKey", "yourSecret");
String jsonString = "[{\"name\":\"李镶\", \"frequency\":1}]";
try {
    dataUploader.uploadContactsData(jsonString.getBytes("utf-8"));
```

```
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
```

接口详细信息请参考 JavaDoc。上传数据格式必须跟示例中一样。

5 注意事项

单例释放：BDVRClient 工作在单例模式下，在不再使用识别功能时（如应用程序退出识别界面），需要调用 release 方法来释放 BDVRClient。

提示音：BDVRClient 开放设置接口由开发者来设定识别提示音。开发者在设置时需注意，提示音不应过长（建议不超过 0.3s）。在提示音播放完成前，BDVRClient 不会实际进行识别工作，因此过长的提示音会延长发起语音识别的时间，降低用户体验。如果需要替换语音识别对话框的提示音效果，需要保证名称与开发包中文件一致。

开始识别时间：应用开发者也需要注意语音识别真正开始的时间（即应用程序收到 CLIENT_STATUS_START_RECORDING 事件通知的时刻），在收到此通知前不应在界面上暗示用户可以说话。

FAQ

1. 如何联系我们？

   开发过程中有任何 bug 或反馈意见，请发送邮件至 voice_feedback@baidu.com

2. Unable to execute dex: Multiple dex files define Lcom/baidu/android/common/logging/Log

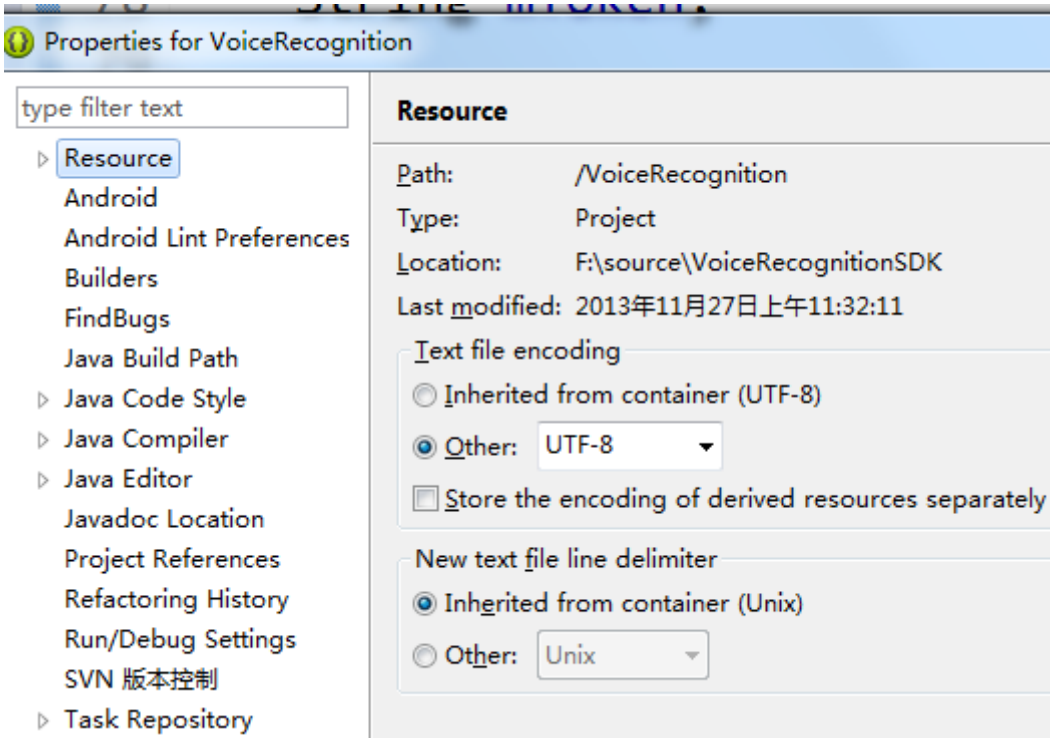
   应用同时集成了百度其它 SDK，造成公共库冲突，请删除语音 SDK 中的 galaxy.jar

3. Caused by: java.lang.UnsatisfiedLinkError: Couldn't load BaiduMapVOS_v2_1_3: findLibrary returned null

   百度语音 SDK 提供了 armeabi 标准库及 armeabi-v7a 库。如项目只包含其中一个目录，请只将语音 SDK 中的同名目录集成，否则会造成其它库无法正常加载的错误。如项目只有 armeabi，请只集成 armeabi，反之如果只有 armeabi-v7a，则只集成 armeabi-v7a。

4. Eclipse 导入 Demo 应用后，项目飘红，无法正常编译

   右键点击项目，选择属性。在弹出的页面修改项目文件格式和字符编码。



Demo 使用的 Target API 是 17，属性页面的 Android 项指定本地有的 API。最低支持 API 8