

---

安徽科大讯飞信息科技股份有限公司

**ANHUI USTC IFLYTEK CO.,LTD**

---

## MSC 开发指南

---

## 重要声明

### 版权声明

版权所有 © 2012, 安徽科大讯飞信息科技股份有限公司, 保留所有权利。

### 商标声明

安徽科大讯飞信息科技股份有限公司的产品是安徽科大讯飞信息科技股份有限公司专有。在提及及其他公司及其产品时将使用各自公司所拥有的商标, 这种使用的目的仅限于引用。本文档可能涉及安徽科大讯飞信息科技股份有限公司的专利 (或正在申请的专利)、商标、版权或其他知识产权, 除非得到安徽科大讯飞信息科技股份有限公司的明确书面许可协议, 本文档不授予使用这些专利 (或正在申请的专利)、商标、版权或其他知识产权的任何许可协议。

### 不作保证声明

安徽科大讯飞信息科技股份有限公司不对此文档中的任何内容作任何明示或暗示的陈述或保证, 而且不对特定目的的适销性及适用性或者任何间接、特殊或连带的损失承担任何责任。本手册内容若有变动, 恕不另行通知。本手册例子中所用的公司、人名和数据若非特别声明, 均属虚构。未得到安徽科大讯飞信息科技股份有限公司明确的书面许可, 不得为任何目的、以任何形式或手段 (电子的或机械的) 复制或传播手册的任何部分。

### 保密声明

本文档 (包括任何附件) 包含的信息是保密信息。接收人了解其获得的本文档是保密的, 除用于规定的目的外不得用于任何目的, 也不得将本文档泄露给任何第三方。

本软件产品受最终用户许可协议 (EULA) 中所述条款和条件的约束, 该协议位于产品文档和/或软件产品的联机文档中, 使用本产品, 表明您已阅读并接受了 EULA 的条款。

版权所有 © 安徽科大讯飞信息科技股份有限公司

Copyright © 2012 ANHUI USTC iFLYTEK CO., LTD.

# 目录

前言 .....	1
1. 概述 .....	2
1.1. 系统架构 .....	2
1.1.1. 软件架构 .....	2
1.1.2. 硬件架构 .....	3
1.2. 名词和缩略语 .....	4
1.3. 文档说明 .....	5
2. 使用说明 .....	5
2.1. 开发说明 .....	5
2.2. 支持平台 .....	5
3. 环境搭建 .....	5
4. 语音识别 .....	6
4.1. 识别控件 (RecognizerDialog) .....	6
4.1.1. 创建对象 .....	6
4.1.2. 设置识别参数 .....	7
4.1.3. 设置录音采样率 .....	8
4.1.4. 获取上传流量 .....	8
4.1.5. 获取下载流量 .....	9
4.1.6. 设置回调接口 .....	9
4.1.7. 错误界面显示 .....	9
4.1.8. 更多按钮显示 .....	10
4.2. 识别控件回调接口 (RecognizerDialogListener) .....	10
4.2.1. 结果回调 .....	10
4.2.2. 结束回调 .....	10
4.3. 识别结果 (RecognizerResult) .....	11
4.3.1. 成员说明 .....	11
5. 语音合成 .....	11
5.1. 语音合成控件 (SynthesizerDialog) .....	11
5.1.1. 创建对象 .....	11
5.1.2. 设置合成文本 .....	12
5.1.3. 设置回调接口 .....	12
5.1.4. 设置合成音频采样率 .....	12
5.1.5. 设置发音人 .....	13
5.1.6. 设置合成语速 .....	13
5.1.7. 设置合成音量 .....	13
5.1.8. 设置背景音乐 .....	13
5.1.9. 获取播放状态 .....	14
5.1.10. 暂停播放 .....	14
5.1.11. 恢复播放 .....	14
5.1.12. 获取上传流量 .....	14

5.1.13.	获取下载流量.....	15
5.2.	语音合成控件回调接口（SynthesizerDialogListener） .....	15
5.2.1.	结束回调.....	15
5.3.	语音合成播放器（SynthesizerPlayer） .....	15
5.3.1.	创建对象.....	16
5.3.2.	播放文本.....	16
5.3.3.	取消合成.....	16
5.3.4.	设置接口.....	16
5.3.5.	播放控制接口.....	17
5.4.	语音合成播放器回调接口（SynthesizerPlayerListener） .....	17
5.4.1.	开始播放回调.....	17
5.4.2.	缓冲进度回调.....	17
5.4.3.	播放进度回调.....	17
5.4.4.	暂停回调.....	18
5.4.5.	重新播放回调.....	18
5.4.6.	结束回调.....	18
6.	数据上传控件.....	19
6.1.	数据上传控件（UploadDialog） .....	19
6.1.1.	创建对象.....	19
6.1.2.	设置上传数据.....	19
6.1.3.	设置回调接口.....	20
7.	用户登录接口.....	20
7.1.	用户登录（SpeechUser） .....	20
7.1.1.	获取用户对象.....	20
7.1.2.	登录.....	20
7.1.3.	获取登录状态.....	21
7.1.4.	注销.....	21
7.2.	数据上传（DataUploader） .....	21
7.2.1.	创建 DataUploader 对象 .....	21
7.2.2.	上传数据.....	21
7.3.	数据下载（DataDownloader） .....	22
7.3.1.	创建 DataDownloader 对象 .....	22
7.3.2.	下载数据.....	22
7.4.	语义搜索（SpeechSearcher） .....	23
7.4.1.	创建 SpeechSearcher 对象.....	23
7.4.2.	语义搜索.....	23
7.5.	通用回调接口（SpeechListener） .....	23
7.5.1.	数据回调.....	23
7.5.2.	结束回调.....	24
7.5.3.	消息回调.....	24
8.	调试设置接口.....	24
8.1.	获取版本号.....	24
8.2.	测试日志.....	25
8.3.	Logcat 日志 .....	25

8.4.	网络状态检查.....	25
8.5.	设置语言 .....	26
9.	使用示例.....	26
9.1.	Project 设置 .....	26
9.2.	语音转写示例.....	27
9.3.	语音合成示例.....	27
9.4.	登录示例.....	28
9.5.	命令词上传示例.....	29
9.6.	abnf 语法上传示例 .....	29
9.7.	语法 ID 识别示例 .....	30
9.8.	语法文件识别示例.....	31
9.9.	语义识别示例.....	31
9.10.	后台上传命令词示例.....	32
9.11.	语义搜索示例.....	33
附录一：	个性发音人列表.....	34

# 前言

欢迎使用讯飞语音云开放平台！

讯飞语音云开放平台是基于讯飞公司已有的ISP和IMS产品，开发出的一款符合移动互联网用户使用的语音应用开发平台，提供语音合成、语音听写、语音识别等服务，为语音应用开发爱好者提供方便易用的开发接口，使得用户能够基于该开发接口进行多种语音应用开发。其主要功能有：

- 1) 实现基于HTTP协议的语音应用服务器，集成讯飞公司最新的语音引擎，支持语音合成、语音听写、语音识别等服务；
- 2) 提供基于移动平台和PC上的语音客户端子系统，内部集成音频处理和音频编解码模块，提供关于语音合成、语音听写、语音识别完善的API。

科大讯飞作为中国最大的智能语音技术提供商，在智能语音技术领域有着长期的研究积累，并在语音合成、语音识别、口语评测、自然语言处理等多项技术上拥有国际领先的成果。科大讯飞是我国唯一以语音技术为产业化方向的“国家863计划成果产业化基地”、“国家规划布局内重点软件企业”、“国家高技术产业化示范工程”，并被原信息产业部确定为中文语音交互技术标准工作组组长单位，牵头制定中文语音技术标准。

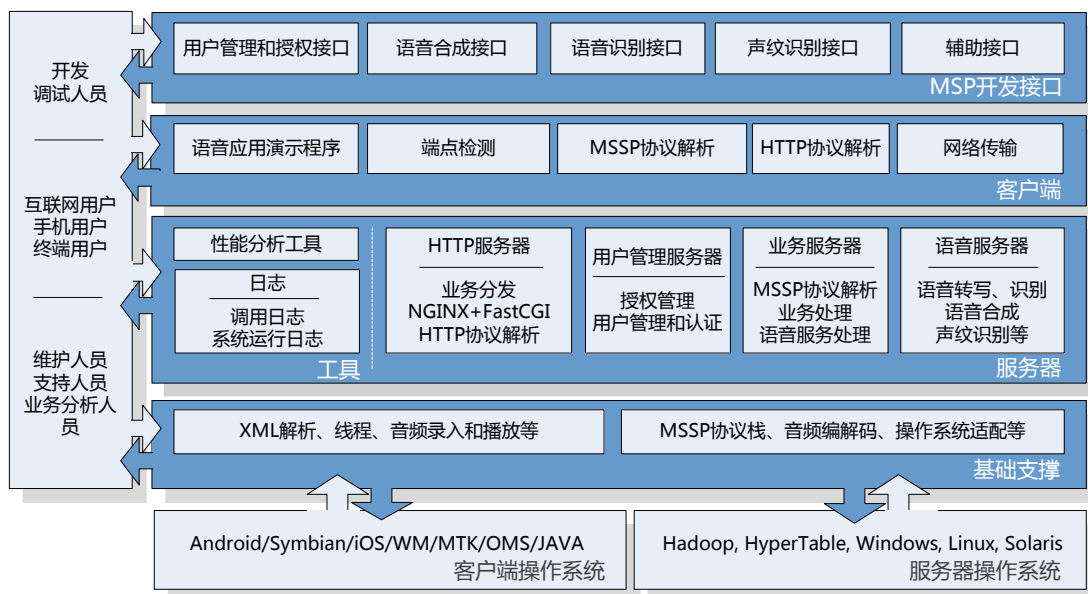
2003年、2011年，科大讯飞两次荣获“国家科技进步奖”。2005年、2011年两次获得中国信息产业自主创新最高荣誉“信息产业重大技术发明奖”。2006年至2012年，已连续七届英文语音合成国际大赛（Blizzard Challenge）荣获第一名。2008年至2011年，连续四年在国际说话人、语种识别评测大赛中名列前茅。2011年，“国家智能语音高新技术产业化基地”、“语音及语言信息处理国家工程实验室”相继落户合肥，有利于进一步汇聚产业资源，提升科大讯飞产业龙头地位。

基于拥有自主知识产权的世界领先智能语音技术，科大讯飞已推出从大型电信级应用到小型嵌入式应用，从电信、金融等行业到企业和消费者用户，从手机到车载，从家电到玩具，能够满足不同应用环境的多种产品。科大讯飞已占有中文语音技术市场70%以上市场份额，开发伙伴超过5000家，以讯飞为核心的中文语音产业链已初具规模。

# 1. 概述

## 1.1. 系统架构

### 1.1.1. 软件架构



语音云平台软件架构图

上图蓝色区域为MSP系统的实现范围，浅色区域是与MSP密切相关的组件或第三方角色。

MSP系统主要包括语音应用接口（Speech Programming Interface, SPI）、客户端（Mobile Speech Client, MSC）、服务器（Mobile Speech Server, MSS）和基础支撑（MSP Infrastructure）四个层次，这四个逻辑层从用户到服务器操作系统底层，共同构成了完整的MSP系统架构。

- **SPI**

应用接口是MSP系统提供的开发接口，集成开发人员应关注这些接口的定义、功能和使用方法。

- **MSC**

MSC负责实现这些接口，同时封装了网络通讯、音频编解码（Audio Codec）、语音检测（VAD）、协议解析（MSSP）等功能，同时为了便于开发和使用，系统在这一层提供了一系列高效、易用的工具。

- **MSS**

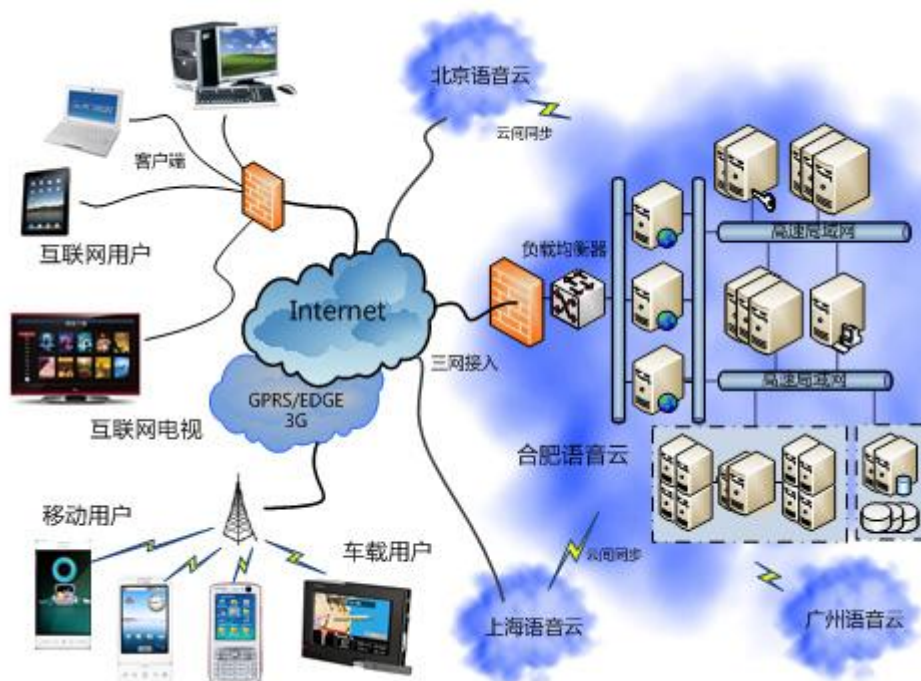
MSS提供语音服务的服务端实现，使用服务端的识别引擎提供语音功能，同时提供管理和维护

功能。

- **MSP Infrastructure**

基础支撑层是 MSP 的服务基础，负责提供适合云计算架构的负载均衡、并行计算、数据存储等功能。

### 1.1.2. 硬件架构



语音云平台硬件架构图

从图中可以看到，完整的MSP平台架构在Internet上，分为服务器端、移动客户端和Internet客户端三个部分。

服务器端为MSP平台的核心部分，提供HTTP应用、用户管理、语音服务等服务，位于局域网内，对外统一接入Internet，为客户端提供唯一的访问点。其中：HTTP服务器负责将客户端发送的服务请求发送至业务服务器，然后由业务服务器按照具体的服务类型进行处理，调用ISP语音应用平台获取具体的语音服务，而后把处理结果返回给HTTP服务器，再回复客户端。

互联网用户直接通过MSP服务器提供的Internet访问点使用语音服务，在集成了MSP平台提供的开发接口后即可在网络畅通的情况下在应用程序中调用语音服务。

移动用户使用智能手机通过移动运营商提供的 2G(GPRS/EDGE/CDMA)或 3G 网络接入 Internet，然后连接到 MSP 服务器获得服务。



## 1.2. 名词和缩略语

### ☐ **TTS ( Text to Speech )**

语音合成(Text To Speech, TTS)是一种能够将任意文字实时转换为连续的自然语音的技术,能够在任何时间、任何地点,向任何人提供语音信息服务的高效便捷手段,非常符合信息时代海量数据、动态更新和个性化查询的需求。

### ☐ **IAT ( iFly Auto Transform )**

语音听写(iFly Auto Transform, IAT)是一种使计算机能够识别人通过麦克风或者电话输入的词语或语句的技术,简单的说就是将语音中的具体内容转换成文字,更适合于日常用语的识别。

### ☐ **ASR ( Automatic Speech Recognition )**

语音识别(Automatic Speech Recognition, ASR)是在语音听写基础上的一种识别技术,着重于对某一领域或特定文法的识别,且识别结果和上传的内容紧密相关,如果用户想自定义识别的范围,在使用前,需上传相关的命令词列表或语法文件。

### ☐ **NLP ( Natural Language Processing )**

自然语言理解(Natural Language Processing, NLP)是人工智能(AI)的一个子领域,也是人工智能中最为困难的问题之一,它包含对语义的判断,使计算机不仅能够识别出语音内容,还能了解用户的意图,帮助用户准确地搜索出想要的结果,最终目的是让计算机做到像人一样的思考。

### ☐ **ISP ( iFLY Speech Platform )**

讯飞语音应用平台(iFLY Speech Platform, ISP),是针对电信级应用场合开发的一个升级扩容方便、能提供高性能、高质量的负载均衡、方便部署、易于维护而且可以进行实时监控和维护的语音应用平台。

### ☐ **IMS ( iFLY MRCP Server )**

讯飞MRCP服务器(iFLY MRCP Server, IMS),支持国际标准协议MRCP 1.0/2.0的语音服务平台,该平台基于ISP架构,提供对国际标准的支持。

### ☐ **MSP ( Mobile Speech Platform )**

讯飞语音云开放平台(Mobile Speech Platform, MSP),或称为IMSP(iFLY Mobile Speech Platform),是讯飞面向移动互联网领域开发的语音服务平台,本项目是该产品的第三个版本。

### ☐ **MSSP ( Mobile Speech Service Protocol )**

移动语音服务协议(Mobile Speech Service Protocol, MSSP),是基于 HTTP1.1 协议扩展的语音应用协议。

## 1.3. 文档说明

本文档定义了科大讯飞的语音合成、语音听写和语音识别的使用说明、体系结构、API 接口，所有接口必需在联网状态下才能正常使用。

其适用的读者为使用语音 SDK 进行开发的产品设计师、软件工程师，通过阅读本文档，读者可以掌握如何集成和使用语音合成、语音听写和语音识别服务。

# 2. 使用说明

## 2.1. 开发说明

- 1、使用语音服务，需要经过我们的授权，请到“<http://open.voicecloud.cn>”注册成为语音云开发者，并为所开发的软件申请 appid；
- 2、已申请 appid 的开发者，将可以免费使用语音识别、语音合成服务，但是对于文档中描述的语义、声纹等业务不具备使用权限，需要和我们签订商务协议并获得授权方可使用；
- 3、应用程序开发语音功能，需要引入 SDK 中包含的 msc.jar 和 libmsc.so 动态库，so 动态库包含 arm、arm-v7a、mips、x86 四种架构，对于普通 Android 手机开发者，如果不需要支持特殊机型，只需引入 arm 架构 so 动态库即可，以减少应用程序安装包大小；
- 4、SDK 中 jar 库包含多个版本，在接口上并无差异，主要在界面上使用不同风格，以满足开发者差异化需求；
- 5、具体业务，请参考开发示例章节。

## 2.2. 支持平台

- 1、支持 Android 1.5 及以上版本系统；
- 2、支持 arm、arm-v7a、mips、x86 架构处理器。

# 3. 环境搭建

Android 开发支持的操作系统为：Windows、Mac OS X 10.4.8、Linux。

由于 Windows 系统为开发者广泛使用的，这里将在 Windows 上的安装环境的搭建步骤简要介绍如下：

- 1) 安装 JDK，编者采用的版本是 jdk1.6.0\_20，读者可以从 Sun 官网 <http://java.sun.com/javase/downloads/index.jsp> 下载所需的版本；
- 2) 安装 Eclipse Java IDE，编者采用的版本是 Ecilpse3.4，读者可以从官网

<http://www.eclipse.org/downloads/packages/release/ganymede/sr2> 下载所需的版本;

- 3) 安装 Android SDK, 编者采用的版本是 android2.2, 读者可以去从 Android 官方网站

<http://developer.android.com/sdk/> 下载所需的版本;

- 4) 安装 ADT 插件, 启动 eclipse 后, 点击 eclipse 的 Help->soft update->find and install->search for new features to install->new remote site->name:https://dl-ssl.google.com/android/eclipse/, 完成安装后重启 eclipse;

- 5) 安装 Android DDMS 和 Android Development Tools;

至此您已经完成了 Android 开发环境的搭建, 更多 Android 了解请参考 SDK 的开发指导文档。

## 4. 语音识别

### 4.1. 识别控件 (RecognizerDialog)

#### 4.1.1. 创建对象

函数原型

```
public RecognizerDialog(Context context, String params);
```

参数说明

参数名	参数解释
context	上下文环境
params	初始化参数列表, 每项中间以半角逗号分隔, 如: "appid=1234567,usr=test,pwd=12345" 可设置参数列表: 1、appid: 应用程序ID (必选) 2、usr: 语音云注册用户名, 默认为null, 使用匿名方式 (可选) 3、pwd: 用户密码 (可选) 4、timeout: 网络超时时间, 单位: ms, 默认为20000, 范围 0-30000 (可选) 5、server_url: 默认连接语音云公网入口 http://dev.voicecloud.cn/index.htm, 只有特定业务才需要设置为固定ip或域名, 普通开发者不需要设置 (可选) 6、besturl_search: 默认为1, 如果server_url设置为固定ip地址, 需要将此参数设置为0, 表示不寻找最佳服务器。如果 server_url为域名, 可以将此参数设置为1 (可选)

说明

- 1、请到<http://open.voicecloud.cn>网站申请appid, 非法的appid将不能使用语音识别服务。
- 2、如果反复调用此接口, 只有第一次传入的params参数有效。

## 4.1.2. 设置识别参数

## 函数原型

```
public void setEngine (String engine, String params, String grammar) ;
```

## 参数说明

参数名	参数解释
engine	<p>识别引擎选择，目前支持以下五种</p> <p>"sms": 普通文本转写</p> <p>"poi": 地名搜索</p> <p>"vsearch": 热词搜索</p> <p>"video": 视频音乐搜索</p> <p>"asr": 语法识别</p>
params	<p>参数列表，每项中间以半角逗号分隔，无附加参数传null，如："vad_bos =4000, vad_eos =2000"</p> <p>可设置参数：</p> <p>1、search_area: 指定 engine 为 poi 搜索时，可通过此参数设定搜索区域，如："search_area =安徽省合肥市"；</p> <p>2、asr_ptt: 默认为 1，当设置为 0 时，将返回无标点符号文本；</p> <p>3、vad_bos: 静音超时时间，即用户多长时间不说话则当做超时处理，单位：ms，engine 指定 sms 识别默认值为 5000，其他情况默认值为 4000，范围 0-10000；</p> <p>4、vad_eos: 后 endpoint 静音检测时间，即用户停止说话多长时间即认为不再输入，自动停止录音，单位：ms，sms 识别默认值为 1800，其他默认值为 700，范围 0-10000；</p> <p>5、plain_result: 返回结果是否在内部进行 json 解析，默认值为 0，即进行解析，返回外部的内容为解析后文本。对于语义等业务，由于服务端返回内容为 xml 或其他格式，需要应用程序自行处理，这时候需要设置 plain_result 为 1，结果回调中返回的 RecognizerResult.text 字段将为未解析原始结果，由外部进行处理；</p> <p>6、asr_sch: 是否需要进行语义处理，默认为 0，即不进行语义识别，对于需要使用语义的应用，需要将 asr_sch 设为 1，并且设置 plain_result 参数为 1，由外部对结果进行解析；</p> <p>7、grammar_type: 语法文件识别类型，如果进行语法文件识别，需要设置此参数（可选值包括 abnf、url-list、grxml，请参考有关标准）。同时设置 engine 为 null，并将语法字符串作为 grammar 参数传入，请参考语法文件识别示例。</p>

grammar	<p>不进行语法识别时，grammar 参数设为“null”；</p> <p>进行语法识别时，engine 参数设为“asr”；</p> <p>语法识别支持两种方式，第一种方式是通过<a href="#">上传命令词</a>或<a href="#">上传 abnf 语法文件</a>获得的 ID 进行识别，第二种方式是直接携带语法内容进行识别。</p> <p>1、第一种方式，grammar 参数设为语法 ID，语法 ID 通过上传接口上传命令词或语法内容获得；</p> <p>2、第二种方式，grammar 参数设为语法内容字符串，同时在 params 参数中设置“grammar_type =***”进行标识；</p>
---------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**说明**

1、调用此接口后启动录音进行识别服务，无效的参数会抛出错误信息。

2、语法 ID 识别示例：

```
RecognizerDialog#setEngine("asr",null,"5a398ba6be0c6cc.....");
```

3、abnf 语法字符串识别示例：

```
RecognizerDialog#setEngine("asr","grammar_type=abnf","ABNF.....");
```

4、sms 转写示例：

```
RecognizerDialog#setEngine("sms",null,null);
```

### 4.1.3. 设置录音采样率

**函数原型**

```
public void setSampleRate(RATE rate);
```

**参数说明**

参数名	参数解释
rate	录音采样率，支持 rate8k、rate11k、rate16k、rate22k 四种，默认为 rate16k

**说明**

1、调用此接口后在下次识别时生效。

2、Android 手机一般只支持 8K 和 16K 两种采样率，为了获得更好的识别效果，推荐使用 16K。

### 4.1.4. 获取上传流量

**函数原型**

```
public int getUpflowBytes(boolean isTotal);
```

**参数说明**

参数名	参数解释
isTotal	true 表示获取应用程序启动到当前的上传流量，false 表示获取最后一次语音识别的上传流量

**返回值：**

与服务器交互所产生的上传流量，单位：字节数（byte）。

#### 说明

本次交互所产生的流量可以在回调接口的 [onEnd](#) 函数中调用，总流量在任意时刻调用均有效。

### 4.1.5. 获取下载流量

#### 函数原型

```
public int getDownflowBytes(boolean isTotal);
```

#### 参数说明

参数名	参数解释
isTotal	true 表示获取应用程序启动到当前的下载流量，false 表示获取最后一次语音识别的下载流量

#### 返回值：

与服务器交互所产生的下载流量，单位：字节数（byte）。

#### 说明

无

### 4.1.6. 设置回调接口

#### 函数原型

```
public void setListener(RecognizerDialogListener listener);
```

#### 参数说明

参数名	参数解释
listener	回调接口，通知外部获取识别结果

#### 返回值

无

#### 说明

- 1、开发者需要实现此接口以获得语音识别的结果，请参考 [RecognizerDialogListener](#) 接口。
- 2、如果用户手动点击界面取消了当前的识别，将不会再有消息通过此接口调用。

### 4.1.7. 错误界面显示

#### 函数原型

```
public void showErrorView(boolean showView,boolean showErrorCode);
```

#### 参数说明

参数名	参数解释
showView	如果设置为 false，发生错误时，RecognizerDialog 将自动调用 dismiss 方法消失，在回调 <a href="#">onEnd</a> 接口中可以获得错误信息。 如果设置为 true，将停留在错误界面，默认为 true

showErrorCode	1、设置为 <code>false</code> ，错误页面将不显示错误码，只显示错误信息，设置为 <code>true</code> ，错误码和错误信息均显示，默认为 <code>true</code> ； 2、此参数在设置错误页面显示的情况下有效。
---------------	-----------------------------------------------------------------------------------------------------------------------------------

#### 4.1.8. 更多按钮显示

##### 函数原型

```
public static void showMoreButton(boolean more);
```

##### 参数说明

参数名	参数解释
more	1、设置为 <code>true</code> ，在发生网络超时，将显示更多按钮，用户可以使用更多页面中的重新说话、重新获取结果，录音回放功能，如果设置为 <code>false</code> ，将不显示更多按钮，只有重新说话可用，默认为 <code>true</code> ，建议使用默认设置； 2、此参数在设置错误页面显示的情况下有效。

### 4.2. 识别控件回调接口（RecognizerDialogListener）

#### 4.2.1. 结果回调

##### 函数原型

```
public void onResults(ArrayList<RecognizerResult> results,boolean isLast);
```

##### 参数说明

参数名	参数解释
results	识别结果，请参考 <a href="#">RecognizerResult</a> 定义
isLast	<code>true</code> 表示最后一次结果， <code>false</code> 表示结果未取完

##### 说明

1、控件采用边录音边上传数据的方式，可能会多次返回结果，建议用户在此接口中只保存结果内容，在 `onEnd` 回调中进行下一步的结果处理。

#### 4.2.2. 结束回调

##### 函数原型

```
public void onEnd(SpeechError error);
```

##### 参数说明

参数名	参数解释
error	请求成功 <code>error</code> 等于 <code>null</code> ，否则返回错误码

##### 说明

1、本次识别过程结束，如果识别成功，对话框自动消失，调用者可以在此函数中进行下一步的处理。如果出现错误，界面不消失，显示相应错误文字，开发者不需要对错误情况进行处理，用户会根据界面提示进行下一步操作。

### 4.3. 识别结果（RecognizerResult）

#### 4.3.1. 成员说明

成员名	参数解释
String text	文本结果
int confidence	结果置信度
ArrayList<HashMap<String,String>> semanteme	语义结果，由本次识别所选择服务定义

##### 说明

- 1、命令词和语法识别时，text 表示识别结果，confidence 表示置信度，取值范围 0-100。
- 2、命令词识别时，如果需要和上传的原词表进行比对，请使用 semanteme 中 key 为 contact 的字段。例：[{"sc":"50","gm":"0","w":"双鹤药业","mn":[{"contact":"双鹤药业"}]}, {"sc":"50","gm":"0","w":"西藏药业","mn":[{"contact":"西藏药业"}]}，其中"contact"对应的内容为原词表内容，"w"对应的内容为识别结果(text 字段)。
- 3、定制化业务的搜索结果存放在 semanteme 中，由具体的业务类型进行约定。
- 4、转写结果时，text 表示转写文本，confidence 置信度默认为 100。

## 5. 语音合成

### 5.1. 语音合成控件（SynthesizerDialog）

#### 5.1.1. 创建对象

##### 函数原型

```
public SynthesizerDialog(Context context, String params);
```

##### 参数说明

参数名	参数解释
context	上下文环境
params	请参考 <a href="#">RecognizerDialog</a> 构造函数

##### 说明

无



### 5.1.2. 设置合成文本

函数原型

```
public void setText(String text, String params);
```

参数说明

参数名	参数解释
text	需要合成的文本
params	初始化参数列表，每项中间以英文逗号分隔，无附加参数传null， 如："tts_buffer_time=5000" 可设置参数： 1、tts_buffer_time：缓冲多少毫秒音频之后开始播放，默认为第一句话缓冲完成开始播放，单位：ms。

说明

- 1、调用此函数后，需要将文本传送到服务端进行合成，会有一定缓冲时间，缓冲时间视手机网络状况而定。

### 5.1.3. 设置回调接口

函数原型

```
public void setListener (SynthesizerDialogListener listener) ;
```

参数说明

参数名	参数解释
listener	合成通知的回调接口

说明

- 1、如无特殊需要可以不用实现此接口。

### 5.1.4. 设置合成音频采样率

函数原型

```
public void setSampleRate(RATE rate);
```

参数说明

参数名	参数解释
rate	支持 rate8k、rate16k 两种，默认为 rate16k

说明

- 1、为了获得更好的音质效果，建议使用 16k 采样率。

### 5.1.5. 设置发音人

#### 函数原型

```
public void setVoiceName(String vcn);
```

#### 参数说明

参数名	参数解释
vcn	发音人名称, 可以设置为"xiaoyan"、"xiaoyu"等, 默认为"xiaoyan" 更多发音人请参考附录一 <a href="#">个性发音人列表</a> 。

#### 说明

- 1、不同的发音人代表了不同的音色, 如男声、女声、童声、地方话等。
- 2、部分发音人只支持中文或英文中一种, 如果需要进行中英文混读, 请选择语言为中英文的发音人。
- 3、如果需要设置发音人为青年女声玛丽, 可以按照下面的方式进行设置  
`SynthesizerDialog.setVoiceName("vimary");`

### 5.1.6. 设置合成语速

#### 函数原型

```
public void setSpeed(int speed);
```

#### 参数说明

参数名	参数解释
speed	范围值: 0-100, 默认为50

#### 说明

无

### 5.1.7. 设置合成音量

#### 函数原型

```
public void setVolume(int volume);
```

#### 参数说明

参数名	参数解释
volume	范围值: 0-100, 默认为50

#### 说明

无

### 5.1.8. 设置背景音乐

#### 函数原型

```
public void setBackgroundSound(String id);
```

## 参数说明

参数名	参数解释
id	背景音乐id，默认为null，表示无背景音乐

## 说明

1、目前支持 id 为“1”的背景音乐，后续会在服务端不断添加，请关注语音云开发网站。

### 5.1.9. 获取播放状态

## 函数原型

```
public PLAY_STATE getState();
```

## 返回值

BUFFERING：缓冲音频状态

PLAYING：正在播放

PAUSED：暂停状态

STOPED：播放完成状态

## 说明

无

### 5.1.10. 暂停播放

## 函数原型

```
public void pause();
```

## 参数说明

无

## 说明

1、合成处于 BUFFERING 和 PLAYING 状态下有效。

### 5.1.11. 恢复播放

## 函数原型

```
public void resume();
```

## 参数说明

无

## 说明

1、合成处于 PAUSED 状态下有效。

### 5.1.12. 获取上传流量

## 函数原型

```
public int getUpflowBytes(boolean isTotal);
```

## 参数说明

参数名	参数解释
isTotal	true 表示获取应用程序启动到当前的合成上传流量，false 表示获取最后一次合成的上传流量

返回值:

与服务器交互所产生的上传流量。

说明

无

### 5.1.13. 获取下载流量

函数原型

```
public int getDownflowBytes(boolean isTotal);
```

参数说明

参数名	参数解释
isTotal	true 表示获取应用程序启动到当前的合成下载流量，false 表示获取最后一次合成的下载流量

返回值:

与服务器交互所产生的下载流量。

说明

无

## 5.2. 语音合成控件回调接口（SynthesizerDialogListener）

### 5.2.1. 结束回调

函数原型

```
public void onEnd(SpeechError error);
```

参数说明

参数名	参数解释
error	播放完成或发生错误时调用此接口，error 为 null 表示播放完成

说明

1、播放完成后界面会显示重新播放按钮，不自动消失，如果发生错误会提示用户重新合成，开发者无需在 onEnd 回调中进行处理。

## 5.3. 语音合成播放器（SynthesizerPlayer）

通过此接口可以在后台进行语音合成播放，而不用通过 [SynthesizerDialog](#) 显示界面。

### 5.3.1. 创建对象

函数原型

```
public static SynthesizerPlayer createSynthesizerPlayer(Context context ,String params);
```

参数说明

参数名	参数解释
context	上下文环境
params	请参考 <a href="#">RecognizerDialog</a> 构造函数

说明

无

### 5.3.2. 播放文本

函数原型

```
public void playText(String text, String params, SynthesizerPlayerListener listener);
```

参数说明

参数名	参数解释
text	需要合成的文本
params	请参考 <a href="#">SynthesizerDialog</a> 的 <a href="#">setText</a> 接口
listener	合成回调接口

说明

1、如果多次连续调用此接口，会抛出网络繁忙的错误，只有当前一次会话完成，才可进行下一次播放。

### 5.3.3. 取消合成

函数原型

```
public void cancel();
```

参数说明

无

说明

1、取消本次音频合成和播放。

### 5.3.4. 设置接口

说明

1、设置接口请参考 [SynthesizerDialog](#)。

### 5.3.5. 播放控制接口

说明

- 1、播放控制接口请参考 [SynthesizerDialog](#)。

## 5. 4. 语音合成播放器回调接口（SynthesizerPlayerListener）

### 5.4.1. 开始播放回调

函数原型

```
void onPlayBegin();
```

参数说明

无

说明

- 1、音频缓冲完成，开始播放。
- 2、通过合成开始时的 `tts_buffer_time` 参数可以设置缓冲时间。

### 5.4.2. 缓冲进度回调

函数原型

```
void onBufferPercent(int percent,int beginPos,int endPos);
```

参数说明

参数名	参数解释
percent	音频缓冲进度，范围值：0-100
beginPos	缓冲音频在文本中开始位置
endPos	缓冲音频在文本中结束位置

说明

无

### 5.4.3. 播放进度回调

函数原型

```
void onPlayPercent(int percent,int beginPos,int endPos);
```

参数说明

参数名	参数解释
percent	播放进度，范围值：0-100
beginPos	当前播放音频在文本中开始位置

endPos	当前播放音频在文本中结束位置
--------	----------------

说明

- 1、播放过程中会不断调用此接口通知播放进度。

#### 5.4.4. 暂停回调

函数原型

```
void onPlayPaused();
```

参数说明

无

说明

- 1、播放过程中如果无缓冲音频，会调用此接口通知播放暂停。

#### 5.4.5. 重新播放回调

函数原型

```
void onPlayResumed();
```

参数说明

无

说明

- 1、无缓冲音频的暂停过程中，如果接收到音频，重新恢复播放时会调用此接口。

#### 5.4.6. 结束回调

函数原型

```
void onEnd(SpeechError error);
```

参数说明

参数名	参数解释
error	播放完成返回 null，否则返回错误码

说明

无

## 6. 数据上传控件

### 6.1. 数据上传控件（UploadDialog）

#### 6.1.1. 创建对象

函数原型

```
public UploadDialog(Context context);
```

参数说明

参数名	参数解释
context	上下文环境

说明

- 1、数据上传控件用于上传自定义的命令词表，上传成功可以获得语法文件 ID 进行命令词识别。
- 2、上传数据之前，需要调用 `SpeechUser` 接口先进行登录操作，请参考 [SpeechUser](#) 接口。

#### 6.1.2. 设置上传数据

函数原型

```
public void setContent(String name, byte[] data, String params);
```

参数说明

参数名	参数解释
name	上传的数据的名称，如果联系人数据，可以设置为“keys”，开发者可以自己进行设置
data	上传数据的二进制流
params	参数列表

说明

- 1、命令词上传示例：

支持以“,”分割的命令词组，传入数据需要以 utf-8 格式进行编码，数据名称可以设为“keys”，参数为“subject =asr,data\_type =keylist”。

```
UploadDialog#setContent("keys",keys.getBytes("utf-8"), "subject =asr,data_type =keylist");
```

- 2、abnf 语法上传示例：

支持标准 abnf 语法，编码格式由 abnf 文件头标识确定，如：#ABNF 1.0 gb2312;则需要将 abnf 语法字符串以 gb2312 方式进行编码，数据名称可以设为“abnf”，参数为“subject =asr,data\_type =abnf”。

```
UploadDialog#setContent("abnf",keys.getBytes("gb2312"), "subject =asr,data_type =abnf");
```



### 6.1.3. 设置回调接口

函数原型

```
public void setListener(SpeechListener listener);
```

参数说明

参数名	参数解释
listener	请参考 <a href="#">SpeechListener</a>

说明

无

## 7. 用户登录接口

扩展接口涉及到用户登录、数据上传、下载等操作。

### 7.1. 用户登录（SpeechUser）

#### 7.1.1. 获取用户对象

函数原型：

```
public static SpeechUser getUser();
```

说明：

- 1、获取用户对象，用于用户登录、注销。
- 2、进行数据上传、声纹等相关业务前，需要先调用此接口执行登录操作。

#### 7.1.2. 登录

函数原型：

```
public boolean login(Context context, String usr, String pwd, String param, SpeechListener listener);
```

参数说明：

参数名	参数解释
context	上下文环境
usr	用户名，通过在语音云注册用户获得，传null使用匿名方式进行登录
pwd	密码
param	请参考 <a href="#">RecognizerDialog</a> 或SpeechRecognizer初始化接口
listener	登录回调接口

说明:

- 1、进行数据上传、声纹等相关业务前，需要先调用此接口执行登录操作。
- 2、username 和 password 为 <http://open.voicecloud.cn> 网站上申请的用户名和密码。
- 3、如果 username 和 password 传 null，则表示使用匿名方式进行登录，匿名用户 ID 由语音云生成，并在手机端进行保存，下次登录仍然有效，是设备访问语音云的唯一性标识。
- 4、对于终端应用，建议使用匿名用户方式进行登陆，以保证终端唯一性。如果使用同样的用户名和密码，服务端会当成一个用户来处理，不建议使用这种方式。

### 7.1.3. 获取登录状态

函数原型:

```
public Login_State getLoginState ();
```

返回值:

登录状态，返回的 Login\_State 为枚举类型，Logged 表示已登录，Unlogin 表示未登录。

### 7.1.4. 注销

函数原型:

```
public boolean logout();
```

返回值:

注销是否成功，如果注销成功或用户未登录，返回 true，注销失败返回 false。

## 7.2. 数据上传（DataUploader）

DataUploader 对象实现的功能和 [UploadDialog](#) 一样，所不同的是不会弹出上传提示框。

### 7.2.1. 创建 DataUploader 对象

函数原型:

```
public DataUploader();
```

说明:

构造 DataUploader 对象，用于上传数据到语音云。

### 7.2.2. 上传数据

函数原型

```
public void uploadData(Context context,SpeechListener listener,String name,String params,  
byte[] uploadData);
```

参数说明

参数名	参数解释
context	上下文环境
listener	接口监听器
name	上传数据名称，可以由开发者自定义
params	数据携带参数值，用以说明数据类型，由服务端具体业务进行约定
data	需要上传的数据

#### 说明

##### 1、上传联系人示例：

`DataUploader#uploadData(context,listener,"contacts","subject=uup,data_type=contact",data)`  
 data:联系人名称之间以换行符进行分隔，拼接成字符串后以 utf-8 方式进行编码。

##### 2、命令词上传示例：

支持以“,”分割的命令词组，传入数据需要以 utf-8 格式进行编码，数据名称可以设为“keys”，参数为“subject =asr,data\_type =keylist”

```
DataUploader#uploadData(context, listener, " keys ", "subject=asr,data_type=keylist",
keys.getBytes("utf-8"));
```

##### 3、abnf 语法上传示例：

支持标准 abnf 语法，编码格式由 abnf 文件头标识确定，如：#ABNF 1.0 gb2312;则需要将 abnf 语法字符串以 gb2312 方式进行编码，数据名称可以设为“abnf”，参数为“subject =asr,data\_type =abnf”。

```
DataUploader # uploadData (context, listener,
"subject=asr,data_type=abnf","abnf",keys.getBytes("gb2312"));
```

## 7.3. 数据下载（DataDownloader）

### 7.3.1. 创建 DataDownloader 对象

#### 函数原型：

```
public DataDownloader ();
```

#### 说明：

- 1、构造 DataDownloader 对象，用于下载数据。

### 7.3.2. 下载数据

#### 函数原型

```
Public void downloadData(Context context, SpeechListener listener, String param);
```

#### 参数说明

参数名	参数解释
context	上下文环境

listener	接口监听器
param	请求数据描述 (必填)

说明

- 1、下载数据需要的参数由具体业务进行约定，暂无说明。

## 7.4. 语义搜索 (SpeechSearcher)

### 7.4.1. 创建 SpeechSearcher 对象

函数原型

```
public SpeechSearcher ();
```

说明

- 1、构造 SpeechSearcher 对象，用于语义搜索。语义搜索为直接传入文本形式接口。

### 7.4.2. 语义搜索

函数原型

```
public void searchText(Context context, SpeechListener listener, String param, String text);
```

参数说明

参数名	参数解释
context	上下文环境
listener	接口监听器
param	搜索文本所携带描述 (可以为空)
text	需要进行语义搜索的文本

说明

- 1、文本搜索返回语义格式结果，请参考《MSC 语义解析》，注：根据客户需求我们会提供不同的版本。
- 2、语义搜索需要使用纯文本内容，过长或者包含特殊符号的字符串，服务端将会返回错误。
- 3、搜索结果为 utf-8 编码的二进制数据。

## 7.5. 通用回调接口 (SpeechListener)

### 7.5.1. 数据回调

函数原型

```
public void onData(byte[] buffer);
```

参数说明

参数名	参数解释
buffer	服务端返回的二进制数据

**说明**

- 1、服务端返回的数据，由具体业务进行不同处理。
- 2、上传命令词只会返回一次结果，但是下载数据时可能会多次调用此接口。

## 7.5.2. 结束回调

**函数原型**

```
public void onEnd(SpeechError error);
```

**参数说明**

参数名	参数解释
error	error 不为 null 表示发生错误，null 表示会话成功

**说明**

- 1、会话成功，上传界面消失，开发者可以进行下一步的处理；如果发生错误，控件界面会提示用户重新上传或设置网络，不需要开发者进行处理。

## 7.5.3. 消息回调

**函数原型**

```
public void onEvent(int eventType, Bundle params);
```

**参数说明**

参数名	参数解释
eventType	消息类型
params	消息数据对象

**返回值**

无

**说明**

- 1、扩展用接口，由具体业务进行约定。

# 8. 调试设置接口

获取版本号接口包含在 `com.iflytek.Version` 类中，通过 `getVersion` 接口实现。其他设置接口包含在 `com.iflytek.Setting` 类中，主要用以记录调试日志，请参考如下的接口。

## 8.1. 获取版本号

**函数原型**

```
public static String getVersion();
```

返回值

版本号，示例：2.0.1013.1034

## 8.2. 测试日志

函数原型

```
public static void saveLogFile(LOG_LEVEL level,String path);
```

参数说明

参数名	参数解释
level	日志打印级别，如果发生一般的网络问题，可以将级别设为 detail，级别说明如下： all：所有日志 detail：高，异常分析需要的级别 normal：中，打印基本日志信息 low：低，只打印主要日志信息 none：不打印
path	日志文件保存路径，设置时请确保路径可读写，如： /sdcard/msc.log，如果设为 null，默认保存路径为 /sdcard/msc/msc.log。

说明

1、软件发布时尽量不要保存日志，以免生成大量日志文件，如果发生网络连接等问题，可以在开发阶段将生成的日志文件发送到 [msp\\_support@iflytek.com](mailto:msp_support@iflytek.com) 获得支持。

## 8.3. Logcat 日志

函数原型

```
public static void showLogcat(boolean showlog);
```

参数说明

参数名	参数解释
showlog	设置为 false，将不显示 logcat 的打印信息，设置为 true，将显示 logcat 的打印信息，默认为 true，建议开启，以便出现问题时技术支持人员可以及时定位。

## 8.4. 网络状态检查

函数原型

```
public static void checkNetwork(boolean checknetwork);
```

参数说明

参数名	参数解释
checknetwork	设置为 true，进行识别和合成会话前，将进行网络检查，直接尝试连接服务器，false 将不进行检查，默认为 true。

#### 说明

- 1、提供此设置接口的主要原因是在部分电视平台 Android 系统中，获取网络状态的结果与实际情况不一致造成。
- 2、如无特殊需要不建议进行设置，这样在无网络的情况下，可以直接通过对话框设置按钮打开系统网络设置页面。

## 8.5. 设置语言

#### 函数原型

```
public static void setLanguage(Locale lan);
```

#### 参数说明

参数名	参数解释
lan	设置界面显示语言，支持简体中文-Locale.CHINESE，英文-Locale.ENGLISH，繁体中文 Locale.TRADITIONAL_CHINESE，设置无效值使用默认语言-简体中文

## 9. 使用示例

### 9.1. Project 设置

1、在 Eclipse 中选中工程，通过工具栏 Project->Properties->Java Build Path->Libraries->Add JARS 或 ADD External JARS 引入 Msc.jar。

2、将 SDK.\lib 目录下 libs 文件夹拷贝到工程根目录，确保.\libs\\*\*\libmsc.so 文件存在。  
so 动态库包含 arm、arm-v7a、mips、x86 四种架构，对于普通 Android 手机开发者，如果不需要支持特殊机型，只需引入 arm 架构 so 动态库即可，以减少应用程序安装包大小。

3、在工程 AndroidManifest.xml 文件中添加如下权限

```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

4、工程混淆设置：如需打包或者生成 APK 的时候进行混淆，在 proguard.cfg 中添加如下代码  
`-keep class com.iflytek.**{*;}`

## 9. 2. 语音转写示例

```
// 创建识别对话框，appid通过http://open.voicecloud.cn站点申请
RecognizerDialog isrDialog = new RecognizerDialog(this,"appid=12345678");
isrDialog.setEngine("sms",null,null);
isrDialog.setListener(recognizeListener);
isrDialog.show();
String text = "";
// 转写回调监听器.
RecognizerDialogListener recognizeListener = new RecognizerDialogListener(){
    // 识别结果回调接口
    public void onResults(ArrayList<RecognizerResult> results, boolean
        isLast){
        // 一般情况下会通过onResults接口多次返回结果，完整的识别内容是多次结果的累加.
        for(int i = 0; i < results.size(); i++)
            text += results.get(i).text;
    }
}
// 会话结束回调接口.
public void onEnd(SpeechError error) {
    // error为null表示会话成功，可在此处理text结果，error不为null，表示发生错误，对话框停留
    // 在错误页面
}
}
```

## 9. 3. 语音合成示例

```
// 创建SynthesizerDialog对象，appid通过http://open.voicecloud.cn站点申请
// 显示对话框方式
SynthesizerDialog synDialog = new SynthesizerDialog(this, "appid=12345678");
synDialog.setListener(synListener);
synDialog.setVoiceName("xiaoyu");
synDialog.setText("安徽科大讯飞信息科技股份有限公司", null);
synDialog.show();
SynthesizerDialogListener synListener = new SynthesizerDialogListener(){
    // 会话结束回调接口
    public void onEnd(SpeechError error) {
        // error为null，表示播放完成，error不为null，表示发生错误
    }
};
// 不显示对话框方式
SynthesizerPlayer player = SynthesizerPlayer.createSynthesizerPlayer(this, "appid=12345678");
player.setVoiceName("vixm");
```



```
player.playText("安徽科大讯飞信息科技股份有限公司","tts_buffer_time=2000", synbgListener);
SynthesizerPlayerListener synbgListener = new SynthesizerPlayerListener (){
    void onPlayBegin(){
        // 播放开始回调，表示已获得第一块缓冲区音频开始播放
    }
    void onBufferPercent(int percent,int beginPos,int endPos){
        // 缓冲回调，通知当前缓冲进度
    }
    void onPlayPaused(){
        // 暂停通知，表示缓冲数据播放完成，后续的音频数据正在获取
    }
    void onPlayResumed(){
        // 暂停通知后重新获取到后续音频，表示重新开始播放
    }
    void onPlayPercent(int percent,int beginPos,int endPos){
        // 播放回调，通知当前播放进度
    }
    void onEnd(SpeechError error){
    }
};
```

## 9. 4. 登录示例

```
// 用户只有在登录状态才能进行数据上传、下载、语义搜索、密码文本下载等操作
// 注册用户登录需要上传用户名和密码，非匿名方式
// 用户名和密码传null，表示使用匿名方式进行登录,每台设备具备唯一性，建议使用此种方式
SpeechUser.getUser().login(this, null, null, "appid=12345678", loginListener);
// 登录回调监听器
SpeechListener loginListener = new SpeechListener()
{
    public void onEnd(SpeechError error) {
        // 若error不为null，表示登录成功
    }
    public void onData(byte[] buffer) {
        // 登录操作服务端不会返回数据，忽略此接口
    }
    public void onEvent(int eventType, Bundle params) {
        // 扩展消息，忽略此接口
    }
};
```

## 9.5. 命令词上传示例

```
// 上传语法需要先进行登录操作，请参考登录示例
SpeechUser.getUser().login(this, null, null, "appid=12345678", null);
// 命令词示例，命令词中间以半角逗号进行分割
String keys = "张三,李四,Andy,Tom,张学友";
// 创建上传对话框
UploadDialog uploadDialog = new UploadDialog(this);
uploadDialog.setListener(uploadListener);
// subject=asr,data_type=keylist表示上传命令词参数
uploadDialog.setContent("keys", keys.getBytes("UTF-8"), "subject=asr,data_type=keylist");
uploadDialog.show();
// 语法ID
String grammarID = null;
// 上传数据回调监听器
SpeechListener uploadListener = new SpeechListener()
{
    // 会话结束回调
    public void onEnd(SpeechError error) {
        // 若error不为null，对话框将显示错误提示
    }
    // 回调数据
    public void onData(byte[] buffer) {
        // 获得语法ID
        grammarID = new String(buffer);
    }
    public void onEvent(int eventType, Bundle params) {
    }
};
```

## 9.6. abnf 语法上传示例

```
// 上传语法需要先进行登录操作，请参考登录示例
SpeechUser.getUser().login(this, null, null, "appid=12345678", null);
// ABNF语法示例，可以说“北京到上海”
String grammar = "#ABNF 1.0 gb2312;
    language zh-CN;
    mode voice;
    root $main;
    $main = $place1 到$place2 ;
    $place1 = 北京 | 武汉 | 南京 | 天津 | 天京 | 东京;
    $place2 = 上海 | 合肥;";
// 创建上传对话框
```

```
UploadDialog uploadDialog = new UploadDialog(this);
uploadDialog.setListener(uploadListener);
// subject=asr,data_type=abnf表示上传命令词参数
uploadDialog.setContent("abnf", grammar.getBytes("gb2312"), "subject=asr,data_type=abnf");
uploadDialog.show();
// 语法ID
String grammarID = null;
// 上传数据回调监听器
SpeechListener uploadListener = new SpeechListener()
{
    // 会话结束回调
    public void onEnd(SpeechError error) {
        // 若error不为null，对话框将显示错误提示
    }
    // 回调数据
    public void onData(byte[] buffer) {
        // 获得语法ID
        grammarID = new String(buffer);
    }
    public void onEvent(int eventType, Bundle params) {
    }
};
```

## 9.7. 语法 ID 识别示例

```
// 语法ID通过上传命令词或abnf语法获得，请参考上传命令词、上传abnf语法示例
String grammarID = "adkjdn.....";
// 创建识别对话框
RecognizerDialog isrDialog = new RecognizerDialog(this, "appid=12345678");
isrDialog.setEngine("asr", null, grammarID);
isrDialog.setListener(recognizeListener);
isrDialog.show();
// 识别回调监听器
RecognizerDialogListener recognizeListener = new RecognizerDialogListener(){
    // 识别结果回调
    public void onResults(ArrayList<RecognizerResult> results,boolean
        isLast){
        // results表示识别结果列表，可能会返回多个识别结果，默认通过置信度进行排序
    }
    //会话结束回调
    public void onEnd(SpeechError error) {
        // 若error不为null，对话框将显示错误提示
    }
};
```

## 9.8. 语法文件识别示例

```
// 创建对话框,appid通过http://open.voicecloud.cn站点申请
RecognizerDialog isrDialog = new RecognizerDialog(this, "appid=12345678");
isrDialog.setListener(recognizeListener);
// ABNF语法示例,可以说“北京到上海”
String grammar = "#ABNF 1.0 gb2312;
                    language zh-CN;
                    mode voice;
                    root $main;
                    $main = $place1 到$place2 ;
                    $place1 = 北京 | 武汉 | 南京 | 天津 | 天京 | 东京;
                    $place2 = 上海 | 合肥;";
isrDialog.setEngine("asr", "grammar_type=abnf", grammar);
isrDialog.show();
//识别回调监听器
RecognizerDialogListener recognizeListener = new RecognizerDialogListener(){
    //识别结果回调
    public void onResults(ArrayList<RecognizerResult> results,boolean
        isLast){
        // results表示识别结果列表,可能会返回多个识别结果,默认通过置信度进行排序
    }
    //会话结束回调
    public void onEnd(SpeechError error) {
        // 若error不为null,对话框将显示错误提示
    }
}
```

## 9.9. 语义识别示例

```
// 创建对话框,APPID通过http://open.voicecloud.cn站点申请
RecognizerDialog isrDialog = new RecognizerDialog(this, "appid=12345678");
isrDialog.setListener(recognizeListener);
// 设置语义识别的参数,plain_result=1表示服务端返回的结果SDK内部不需要进行解析,由
// 开发者根据服务端约定格式进行处理,asr_sch=1表示需要进行语义处理
isrDialog.setEngine("sms", "asr_sch=1,plain_result=1", null);
isrDialog.show();
private RecognizerDialogListener recognizeListener = new
RecognizerDialogListener() {
    public void onResults(ArrayList<RecognizerResult> results,boolean isLast) {
        // 开发者通过获取 RecognizerResult.text 字段进行自定义处理
    }
    public void onEnd(SpeechError error) {
```

```
// 若error不为null，对话框将显示错误提示
}
};
// 对于部分语义业务，需要上传联系人名称，以支持打电话给***的说法，可以按照下面方式
// 传入，上传数据格式utf-8，联系人中间以\n换行符进行分割
// 上传命令词需要先进行登录操作，请参考登录示例
SpeechUser.getUser().login(this, null, null, "appid=12345678", null);
UploadDialog uploadDialog = new UploadDialog(this);
String contacts = "王小贰\n张小山\n李四\n科大讯飞\ngoog\n汪山三\n章栋";
byte[] datas = contacts.getBytes("utf-8");
uploadDialog.setListener(uploadListener);
// 上传联系人参数为"subject=uup,data_type=contact"
uploadDialog.setContent("contacts",datas, "subject=uup,data_type=contact");
uploadDialog.show();
// 上传联系人接口回调
SpeechListener uploadListener = new SpeechListener()
{
    @Override
    public void onEnd(SpeechError error) {
        // error为null表示上传成功
    }
    @Override
    public void onData(byte[] buffer) {
        // 上传联系人服务端不返回数据，忽略此消息
    }
    @Override
    public void onEvent(int arg0, Bundle arg1) {
    }
};
```

## 9.10. 后台上传命令词示例

```
// 用户登录
SpeechUser.getUser().login(this, null, null, "appid=12345678", null);
// 创建上传数据对象
DataUploader uploader = new DataUploader();
uploader.setListener(uploadListener);
String keys = "张三,李四,Andy,Tom,张学友";
// subject=asr,data_type=keylist表示上传命令词参数
uploader.uploadData(this,"contact","subject=asr,data_type=keylist", keys.getBytes("UTF-8"));
// 语法ID
String grammarID = null;
SpeechListener uploadListener = new SpeechListener()
{
```

```
//会话结束接口
public void onEnd(SpeechError error) {
    // error 不为 null 表示上传失败
}
public void onData(byte[] buffer) {
    // 获得语法ID
    grammarID = new String(buffer);
}
public void onEvent(int eventType, Bundle params) {
    // 扩展接口，忽略此消息
}
};
```

## 9.11. 语义搜索示例

```
// 用户登录
SpeechUser.getUser().login(this, null, null, "appid=12345678", null);
// 创建搜索对象
SpeechSearcher searcher = new SpeechSearcher();
searcher.searchText(this, searchListener, null, "科大讯飞");
SpeechListener searchListener = new SpeechListener()
{
    // 会话结束接口
    public void onEnd(SpeechError error) {
        // error 不为 null 表示搜索失败
    }
    // 返回搜索结果
    public void onData(byte[] buffer) {
        // 搜索结果，utf-8 编码
        String text = new String(arg0,"utf-8")
    }
    public void onEvent(int eventType, Bundle params) {
        // 扩展接口，忽略此消息
    }
};
```

## 附录一：个性发音人列表

- 1、语言为中英文的发音人可以支持中英文的混合朗读；
- 2、英文发音人只能朗读英文，中文无法朗读；
- 3、汉语发音人只能朗读中文，遇到英文会以单个字母的方式进行朗读。

发音人名称	属性	语言	参数名称	备注
小燕	青年女声	中英文（普通话）	xiaoyan	默认
小宇	青年男声	中英文（普通话）	xiaoyu	
凯瑟琳	青年女声	英文	Catherine	
亨利	青年男声	英文	henry	
玛丽	青年女声	英文	vimary	
小研	青年女声	中英文（普通话）	vixy	
小琪	青年女声	中英文（普通话）	vixq	
小峰	青年男声	中英文（普通话）	vixf	
小梅	青年女声	中英文（粤语）	vixm	
小莉	青年女声	中英文（台湾普通话）	vixl	
小蓉	青年女声	汉语（四川话）	vixr	
小芸	青年女声	汉语（东北话）	vixyun	
小坤	青年男声	汉语（河南话）	vixk	
小强	青年男声	汉语（湖南话）	vixqa	
小莹	青年女声	汉语（陕西话）	vixying	
小新	童年男声	汉语（普通话）	vixx	
楠楠	童年女声	汉语（普通话）	vinn	
老孙	老年男声	汉语（普通话）	vils	