语音交互服务

SDK 参考

文档版本 01

发布日期 2025-09-11

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 SDK 简介

语音交互概述

语音交互服务（Speech Interaction Service，简称SIS）是一种人机交互方式，用户通过实时访问和调用API获取语音交互结果。支持用户通过语音识别功能，将口述音频、普通话或者带有一定方言的语音文件识别成可编辑的文本，同时也支持通过语音合成功能将文本转换成逼真的语音等提升用户体验。适用场景如语音客服质检、会议记录、语音短消息、有声读物、电话回访等。

新版 SDK

- 优先推荐使用新版SDK**语音交互服务SDK**，该SDK基于统一规范开发，支持Java/Python/C++/.NET/Go/NodeJs/PHP。
- 使用方法可参考**SDK（新版）**，该SDK暂不支持websocket方法。

websocket SDK

- 如果需要使用实时语音识别，可考虑使用该SDK。
- 当前支持Java SDK、Python SDK、CPP SDK、iOS SDK、Android SDK。使用方法可参考**Java SDK**、**Python SDK**、**CPP SDK（Windows）**、**CPP SDK（Linux）**。

SDK 接口与 API 对应关系

Java接口与API对应关系请参见**表 Java接口与API对应关系表**。

表 1-1 Java 接口与 API 对应关系表

Class	Method	API	功能名称
RasrClient	void continueStreamConnect(RasrRequest request)	wss://{endpoint}/v1/{project_id}/rasr/continue-stream	实时流连续模式

Class	Method	API	功能名称
	void shortStreamConnect(RasrRequest request)	wss://{endpoint}/v1/{project_id}/rasr/short-stream	实时流一句话模式
	void sentenceStreamConnect(RasrRequest request)	wss://{endpoint}/v1/{project_id}/rasr/sentence-stream	实时流单句模式
AsrCustomizationClient	AsrCustomShortResponse getAsrShortResponse(AsrCustomShortRequest request)	POST /v1/{project_id}/asr/short-audio	一句话识别
	String submitJob(AsrCustomLongRequest request)	POST /v1/{project_id}/asr/transcriber/jobs	录音文件识别-提交请求
	AsrCustomLongResponse getAsrLongResponse(String jobId)	GET /v1/{project_id}/asr/transcriber/jobs/{job_id}	录音文件识别-状态查询
TtsCustomizationClient	TtsCustomResponse getTtsResponse(TtsCustomRequest request)	POST /v1/{project_id}/tts	语音合成
HotWordClient	String create(HotWordRequest request)	POST /v1/{project_id}/asr/vocabularies	创建热词表
	String update(HotWordRequest request, String vocabularyId)	PUT /v1/{project_id}/asr/vocabularies/{vocabulary_id}	更新热词表
	HotWordResponse query(String vocabularyId)	GET /v1/{project_id}/asr/vocabularies/{vocabulary_id}	查询热词表信息
	HotWordsResponse query()	GET /v1/{project_id}/asr/vocabularies	查询热词表列表
	void delete(String vocabularyId)	DELETE /v1/{project_id}/asr/vocabularies/{vocabulary_id}	删除热词表

Python接口与API对应关系请参见**表 Python接口与API对应关系表**。

表 1-2 Python 接口与 API 对应关系表

Class	Method	API	功能名称
RasrClient	continue_stream_connect(request)	wss://{endpoint}/v1/{project_id}/rasr/continue-stream	实时流连续模式
	short_stream_connect(request)	wss://{endpoint}/v1/{project_id}/rasr/short-stream	实时流一句话模式
	sentence_stream_connect(request)	wss://{endpoint}/v1/{project_id}/rasr/sentence-stream	实时流单句模式
AsrCustomizationClient	get_short_response(request)	POST /v1/{project_id}/asr/short-audio	一句话识别
	submit_job(request)	POST /v1/{project_id}/asr/transcriber/jobs	录音文件识别-提交请求
	get_long_response(job_id)	GET /v1/{project_id}/asr/transcriber/jobs/{job_id}	录音文件识别-状态查询
TtsCustomizationClient	get_tts_response(request)	POST /v1/{project_id}/tts	语音合成
HotWordClient	create(request)	POST /v1/{project_id}/asr/vocabularies	创建热词表
	update(request, vocabulary_id)	PUT /v1/{project_id}/asr/vocabularies/{vocabulary_id}	更新热词表
	query_by_vocabulary_id(vocabularyId)	GET /v1/{project_id}/asr/vocabularies/{vocabulary_id}	查询热词表信息
	query()	GET /v1/{project_id}/asr/vocabularies	查询热词表列表
	delete(vocabulary_id)	DELETE /v1/{project_id}/asr/vocabularies/{vocabulary_id}	删除热词表

2 SDK（新版）

推荐使用新版SDK**语音交互服务SDK**，该SDK基于统一规范开发，支持Java/Python/C++/.NET/Go/NodeJs/PHP，使用方法可参考**API Explorer**。API Explorer可以自动生成SDK代码示例，并提供SDK代码示例调试功能，但该SDK暂不支持通过websocket的方法调用的API。如需使用websocket接口，请前往**SDK（websocket）**。

在线生成 SDK 代码

API Explorer能根据需要动态生成SDK代码功能，降低您使用SDK的难度，推荐使用。

SDK 列表

在开始使用之前，请确保您安装的是最新版本的SDK。使用过时的版本可能会导致兼容性问题或无法使用最新功能。您可以在**SDK中心**查询版本信息。

表2-1提供了**huaweicloud-sdk-php-v3** SIS服务支持的SDK列表，您可以在GitHub仓库查看SDK更新历史、获取安装包以及查看指导文档。

表 2-1 SDK 列表

编程语言	Github地址	参考文档
Java	**huaweicloud-sdk-java-v3**	**Java SDK使用指导**
Python	**huaweicloud-sdk-python-v3**	**Python SDK使用指导**
C++	**huaweicloud-sdk-cpp-v3**	**C++ SDK使用指导**
.NETet	**huaweicloud-sdk-net-v3**	**.NET SDK使用指导**
Go	**huaweicloud-sdk-go-v3**	**Go SDK使用指导**
NodeJs	**huaweicloud-sdk-nodejs-v3**	**NodeJs SDK使用指导**
PHP	**huaweicloud-sdk-php-v3**	**PHP SDK使用指导**

3 SDK（websocket）

下载 SDK 包

鉴于**新版SDK**未提供 WebSocket 相关功能支持，若您的业务场景中需要使用 WebSocket 方法，建议选择本页提供的 SDK（WebSocket 版）以满足需求。

语音交互SDK软件包获取请参见**表 下载SDK包**。

示例音频参见**示例音频**。

当您使用过旧版本的Java SDK时，需要注意旧版本的SDK依赖的jar包与新版本SDK的jar包有无冲突。新版本SDK升级日志为log4j2，同时java-sdk-core升级到3.0.12版本。

说明

- 优先推荐使用新版SDK**语音交互服务SDK**，该SDK基于统一规范开发，支持Java/Python/C++/.NET/Go/NodeJs/PHP，使用方法可参考**api-explorer**、**SDK开发指南**。该SDK暂不支持websocket方法。
- 如果需要使用实时语音识别，可考虑使用**表 下载SDK包**下载SDK使用，当前支持Java SDK、Python SDK、CPP SDK。后序章节均指代该SDK，使用方法可参考**Java SDK**、**Python SDK**、**CPP SDK（Windows）**、**CPP SDK（Linux）**。当前自研SDK仅java和python提供企业项目配置入口，在代码示例中配置，其他语言暂未支持。

表 3-1 下载 SDK 包

SDK语言	下载地址
Java	**https://sis-sdk-repository.obs.cn-north-1.myhuaweicloud.com/java/huaweicloud-java-sdk-sis-1.8.3.zip**
Python	**https://sis-sdk-repository.obs.cn-north-1.myhuaweicloud.com/python/huaweicloud-python-sdk-sis-1.8.3.zip**
iOS(Swift)	**https://sis-sdk-repository.obs.cn-north-1.myhuaweicloud.com:443/ios/huaweicloud-ios-sdk-sis-1.1.1.zip**

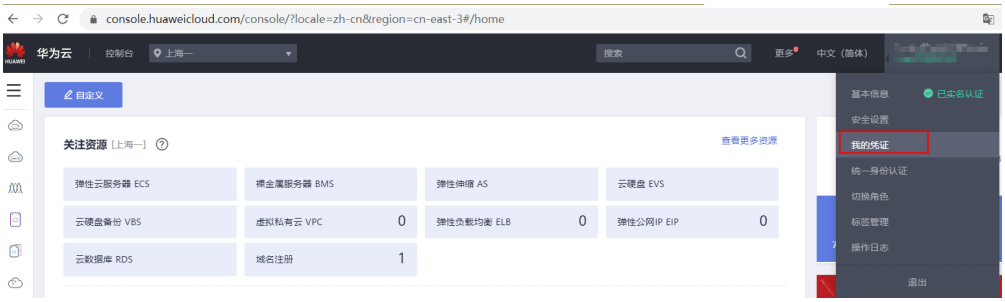
SDK语言	下载地址
Android	https://sis-sdk-repository.obs.cn-north-1.myhuaweicloud.com:443/android/huaweicloud-android-sdk-sis-1.1.1.1.zip
Cpp(Windows)	https://sis-sdk-repository.obs.cn-north-1.myhuaweicloud.com:443/cpp/huaweicloud-sdk-cpp-sis-win-1.4.0.zip
Cpp(Linux)	https://sis-sdk-repository.obs.cn-north-1.myhuaweicloud.com:443/cpp/huaweicloud-cpp-sdk-sis-linux.1.3.2.tar.gz

4 AK/SK 认证

使用服务API需要进行认证，目前SDK仅支持AK/SK认证方式。

使用AK/SK方式，需要用户提供AK和SK。

1. 注册并登录华为云管理控制台。
2. 在控制台中，鼠标移动至右上角的用户名处，在下拉列表中单击“我的凭证”。



3. 单击“访问密钥”页签，在页签中，单击“新增访问密钥”。

4. 在“身份验证”对话框中，输入当前用户的登录密码，通过邮箱或者手机进行验证，输入对应的验证码。
5. 单击“确定”，下载认证账号的AK/SK，请妥善保管AK/SK信息。

5 准备环境

5.1 配置 Java 环境

配置环境

在使用语音交互SDK时，需要准备的环境请参见**表 开发环境**。

说明

目前Java SDK不支持在android中使用。

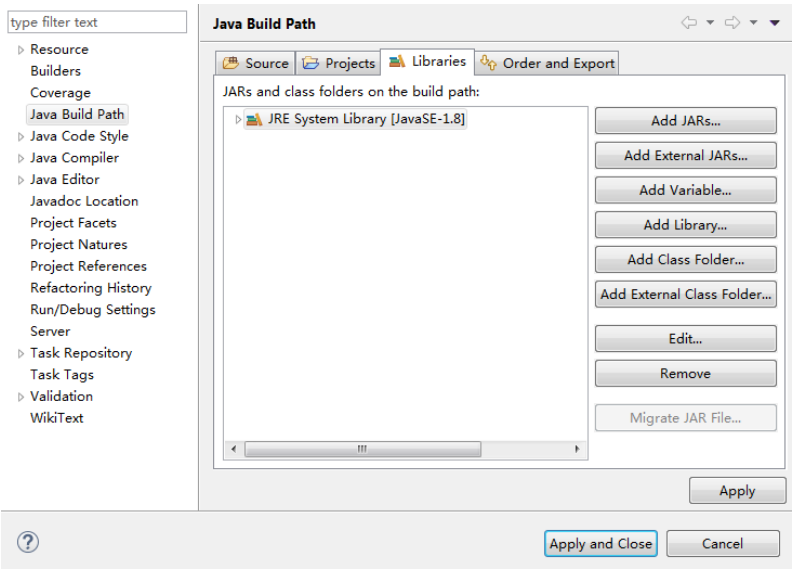
表 5-1 开发环境

准备项	说明
操作系统	Windows系统，推荐Windows 7及以上版本。
JDK	Java开发环境的基本配置。版本要求：强烈推荐使用1.8版本。
Eclipse	在**Eclipse官网**下载对应平台的Eclipse版本，比如：eclipse-jee-mars-R-win32-x86_64.zip。
Idea	在**Idea官网**下载对应平台的idea版本，比如：ideaIU-2023.2.2.exe。

导入 SDK

1. Eclipse导入SDK。
   a. 解压eclipse后，直接打开。同时**下载SDK**。
   b. 选择“Window -> Preferences -> Java -> installed JREs”配置正确的JRE路径。
   c. 新建工程，在工程下建立一个文件（New -> Folder），命名为lib。将下载的jar包拷贝至lib中。

d. 选中新建的工程，单击右键，下拉选择“Build Path -> Configure Build Path”，在“ Java Build Path” 对话框中，单击“Libraries ”页签，选择“Add JARs”。在打开的窗口中，选择刚放进lib的jar包。单击“OK”，导入完成。



2. Idea导入SDK。

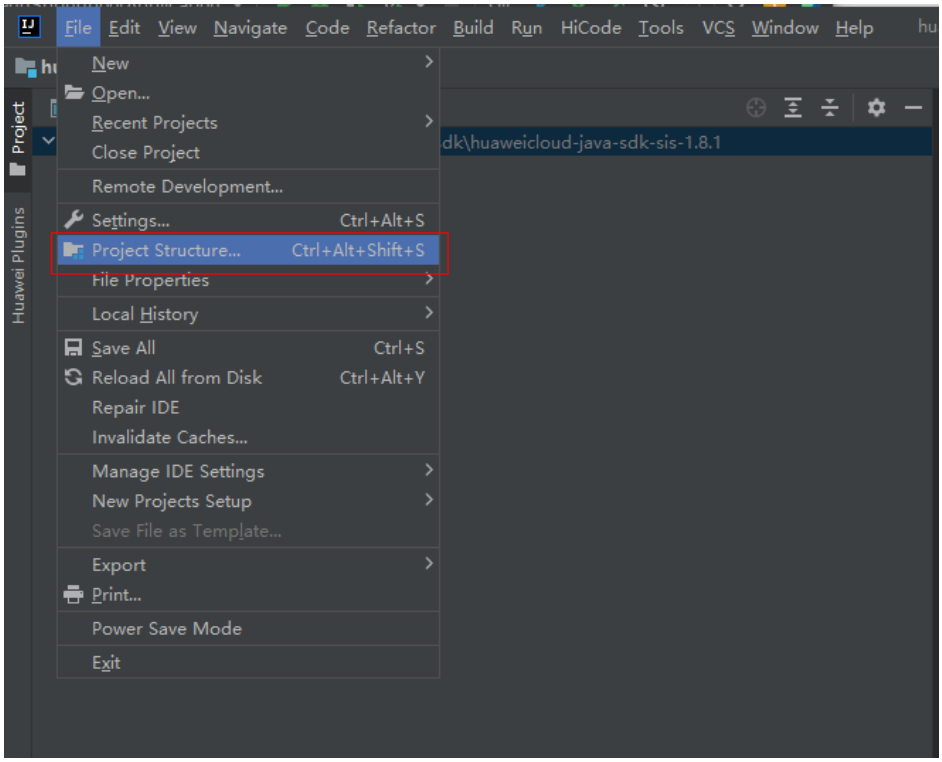
a. 安装idea后，打开idea软件。

b. 项目导入：“File -”>“Open -”>“选择项目SDK项目huaweicloud-java-sdk-sis-1.8.1”（空白项目时直接选择open），选择“Trust Project”。

图 5-1 项目导入

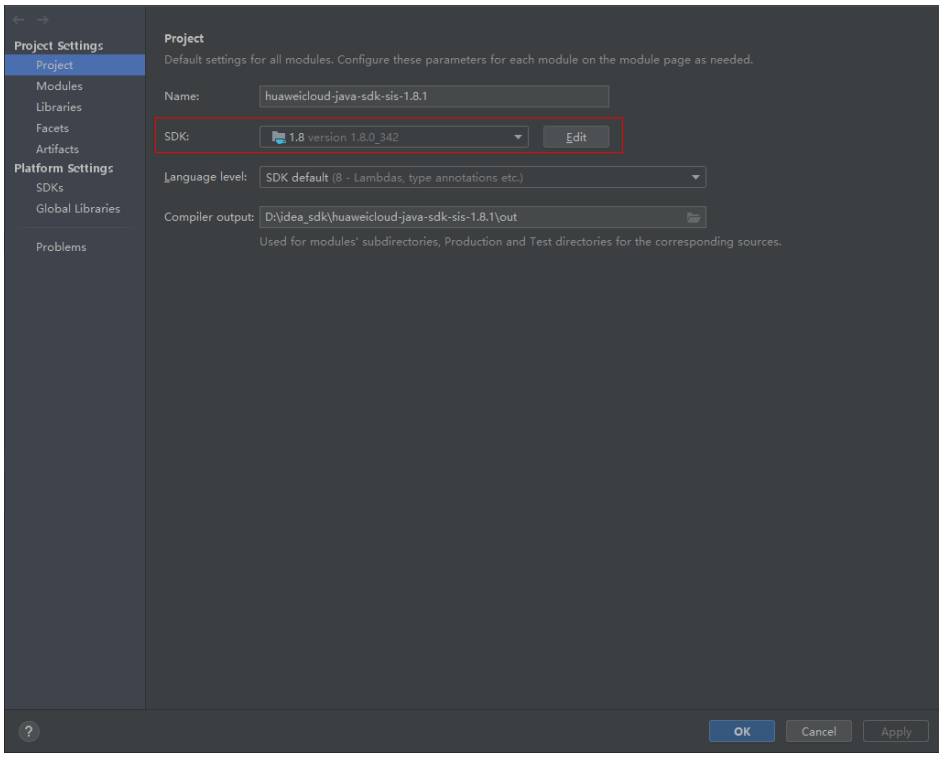
c. 配置java环境：“File -> Project Structure -> Project -> SDK”。

图 5-2 配置 java 环境 1



d. 选择SDK环境，例如：jdk1.8.0_342，单击“OK”即可。

图 5-3 配置 java 环境 2



5.2 配置 Python 环境

前提条件

- 确保已安装Python3，目前Python SDK仅支持Python3。
- 确保已安装setuptools、requests、websocket-client。

操作步骤

1. **下载SDK**，通过**pip-list**命令查看安装包。若未安装，则执行以下命令：
```
pip install setuptools
pip install requests
pip install websocket-client
```
2. 进入下载的Python SDK目录，在setup.py所在层目录执行 python setup.py install 命令，完成SDK安装。

版本说明

websocket-client 在1.x版本以后，新增了onclose接口的入参，导致和之前旧版本不兼容。sdk1.7.1版本已解决该兼容性问题，旧版本sdk如果在使用实时流遇到onclose参数报错，可考虑更新sdk解决问题。

5.3 配置 Android 环境

1. 配置环境。

   在使用语音交互SDK时，需要提前配置好android系统的开发环境，如**表5-2**所示。

   表 5-2 配置环境准备项

准备项	说明
JDK	Java开发环境的基础配置。版本要求:推荐使用1.8版本。
Gradle	在**Gradle官网**上下载好之后，配置环境变量即可。推荐使用Gradle7.x 版本。
AndroidSDK	在**AndroidSDK官网**上下载好之后，配好基础配置即可。
idea	在**idea官网**上下载相应版本即可(也可使用Android Studio开发，根据自己爱好自行选择)。

2. 将下载的 huaweicloud-android-sdk-sis-1.1.0.zip解压之后，使用idea打开该工程，根据指示，修改相关信息即可运行demo（目前该SDK仅支持在Android8 及以上版本系统中使用，推荐使用真机进行调试）。

3. 用户可将libs目录中的huaweicloud-android-sdk-sis-1.1.0.jar，集成到自己的项目中进行开发（demo仅用于指示作用）。

5.4 配置 CPP 环境（Windows）

- 当前Windows 版本 CPP SDK基于visual studio 运行，确保已安装visual studio 2017。
- 依赖curl、openssl、boost、websocketpp软件，相关安装包已打包至SDK中，使用方法详见SDk的使用说明”。

5.5 配置 CPP 环境（Linux）

- c++版本需要在11及以上
- cmake版本需要在3.14及以上
- gcc-c++ 版本需要在5.4.0及以上

Linux版本CPP SDK依赖g++,cmake，此三项需要提前在机器安装，才能完成sdk编译和运行.

CentOS

```
yum install gcc-c++
yum install cmake
```

Ubuntu

```
sudo apt-get install gcc
sudo apt-get install g++
sudo apt-get install cmake
```

三方库安装

本SDK所依赖的三方库包含

- openssl

- jsoncpp

- websocketpp

- glog

- gflags

- boost

依赖库均以源码形式存放在SDK根目录，SDK默认是开启所有依赖库安装，即运行cmake命令时，SDK会默认安装这些依赖库。

如果您在系统中已安装其中一些库，如openssl等，构建的时候可以选择跳过。如cmake .. -DOPENSSL=OFF , 跳过对openssl安装。如果所有依赖库之前已安装在系统中，则可以全部跳过。cmake .. -DWEBSOCKETPP=OFF -DJSONCPP=OFF -DBOOST=OFF -DGFLAGS=OFF -DGLOG=OFF -

DOPENSSL=OFF

详细使用请参考SDK压缩包中的说明文档。

6 Java SDK

6.1 一句话识别 Http 接口

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化AsrCustomizationClient，其参数包括AuthInfo和SisConfig。

表 6-1 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，具体请参考**地区和终端节点**。一般使用默认即可。

表 6-2 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为AsrCustomShortRequest，详见**表6-3**。

表 6-3 AsrCustomShortRequest

参数名称	是否必选	参数类型	描述
data	是	String	本地音频文件经过Base64编码后的字符串，音频文件时长不超过1min。
audioFormat	是	String	音频格式，具体信息请参见《API参考》中**一句话识别**章节。
property	是	String	属性字符串，语言_采样率_模型，如chinese_16k_general。具体信息请参见《API参考》中**一句话识别**章节。
addPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vocabularyId	否	String	热词表id，不使用则不填写。 创建热词表请参考《API参考》中**创建热词表**章节。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

响应参数

响应类为AsrCustomShortResponse，详见**表6-4**。调用失败处理方法请参见**错误码**。

表 6-4 AsrCustomShortResponse

参数名	是否必选	参数类型	说明
trace_id	是	String	服务内部的令牌，可用于在日志中追溯具体流程，调用失败无此字段。 在某些错误情况下可能没有此令牌字符串。
result	是	Object	调用成功表示识别结果，调用失败时无此字段。请参考**表6-5**。

表 6-5 Result

参数名	是否必选	参数类型	说明
text	是	String	调用成功表示识别出的内容。
score	是	Float	调用成功表示识别出的置信度，取值范围：0~1。
word_info	否	Array of objects	分词信息列表。

表 6-6 Word_info 数据结构

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.AsrCustomShortRequest;
import com.huawei.sis.bean.response.AsrCustomShortResponse;
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.client.AsrCustomizationClient;
import com.huawei.sis.exception.SisException;
import com.huawei.sis.util.IOUtils;
import java.util.List;
import com.huawei.sis.util.JsonUtils;


/**
 * 一句话识别
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
```

```
public class AsrCustomizationDemo {
  private static final int SLEEP_TIME = 500;
  private static final int MAX_POLLING_NUMS = 1000;

  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "";    // 区域，如cn-north-1、cn-north-4
  private String projectId = ""; // 项目id。登录管理控制台，鼠标移动到右上角的用户名上，在下拉列表中选择我的凭证，在项目列表中查看项目id。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。
  // 一句话识别参数
  private String path = "";             // 音频文件路径，如D:/test.wav等，sdk会将音频文件转化为base64编码
  private String pathAudioFormat = "";  // 文件格式，如wav等
  private String pathProperty = "chinese_16k_general";     // 属性字符串，language_sampleRate_domain, 16k模型推荐使用chinese_16k_general


  /**
   * 设置一句话识别参数，所有参数均有默认值，不配置也可使用
   *
   * @param request 一句话识别请求
   */
  private void setShortParameter(AsrCustomShortRequest request) {

    // 设置是否添加标点，默认是no
    request.setAddPunc("yes");
    // 设置是否将语音中的数字转写为阿拉伯数字，yes或no，默认yes
    request.setDigitNorm("no");
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }


  /**
   * 一句话识别demo
   */
  private void shortDemo() {
    try {
      // 1. 初始化AsrCustomizationClient
      // 定义authInfo，根据ak，sk，region，projectId
      AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
      // 设置config，主要与超时有关
      SisConfig config = getConfig();
      // 根据authInfo和config，构造AsrCustomizationClient
      AsrCustomizationClient asr = new AsrCustomizationClient(authInfo, config);

      // 2. 配置请求
      String data = IOUtils.getEncodeDataByPath(path);
```

```
    AsrCustomShortRequest request = new AsrCustomShortRequest(data, pathAudioFormat, pathProperty);
    // 设置请求参数，所有参数均为可选
    setShortParameter(request);

    // 3. 发送请求，获取响应
    AsrCustomShortResponse response = asr.getAsrShortResponse(request);
    // 设置企业id, 可选
    // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
    // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
    // AsrCustomShortResponse response = asr.getAsrShortResponse(headers, request);
    // 打印结果
    System.out.println(JsonUtils.obj2Str(response, true));

  } catch (SisException e) {
    e.printStackTrace();
    System.out.println("error_code:" + e.getErrorCode() + "\nerror_msg:" + e.getErrorMsg());
  }
 }

 public static void main(String[] args) {
   AsrCustomizationDemo demo = new AsrCustomizationDemo();
   demo.shortDemo();
 }

}
```

6.2 一句话识别 Websocket 接口

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 该功能为1.70及以上版本SDK新增功能，使用前请检查并更新SDK版本。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化SasrWebsocketClient，其参数包括AuthInfo、RasrListener、SisConfig。

表 6-7 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

表 6-8 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为SasrWebsocketRequest，详见**表 SasrWebsocketRequest**。

表 6-9 SasrWebsocketRequest

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，支持pcm，alaw，ulaw等，如pcm8k16bit，具体规格请参见《API参考》中**开始识别**章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_8k_common。
punc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
intermediateResult	否	String	是否显示中间结果，yes 或 no，默认no。
vocabularyId	否	String	热词表id，若没有则不填。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

响应参数

状态响应类为StateResponse，详见**表6-10**。

结果响应类为RasrResponse，详见**表6-11**。

调用失败处理方法请参见**错误码**。

表 6-10 StateResponse

参数名称	是否必选	参数类型	描述
state	是	String	识别状态，包括start、end、fail。
traceId	是	String	用于日志问题追溯。
description	是	String	状态描述。

表 6-11 RasrResponse

参数名	参数类型	说明
resp_type	String	参数值为RESULT，表示识别结果响应。
trace_id	String	服务内部的令牌，可用于在日志中追溯具体流程。
segments	Array of objects	多句结果。 请参考**表6-12**。

表 6-12 Segment

参数名	参数类型	说明
start_time	Integer	一句的起始时间戳，单位为ms。
end_time	Integer	一句的结束时间戳，单位为ms。
is_final	Boolen	true表示是最终结果， false表示为中间临时结果。
result	Object	调用成功表示识别结果，调用失败时无此字段。 请参考**表6-13**。

表 6-13 Result

参数名	参数类型	说明
text	String	识别结果。
score	Float	识别结果的置信度，取值范围：0~1。此值仅会在最终结果时被赋值，在中间结果时统一置为“0.0”。 **说明** 目前置信度作用不是太大，请勿过多依赖此值。

参数名	参数类型	说明
word_info	Array of Object	分词输出列表。

表 6-14 Word_info 数据结构

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.bean.RasrListener;
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.SasrWebsocketRequest;
import com.huawei.sis.bean.response.RasrResponse;
import com.huawei.sis.bean.response.StateResponse;
import com.huawei.sis.client.SasrWebsocketClient;
import com.huawei.sis.util.JsonUtils;

/**
 * 一句话识别 websocket demo
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class SasrWebsocketDemo {

  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "cn-north-4";         // 区域，如cn-north-1、cn-north-4
  private String projectId = "";                // 项目id，在我的凭证查看。参考https://support.huaweicloud.com/api-sis/sis_03_0008.html

  private String path = "";                     // 本地音频路径，如D:/test.wav, sdk也支持byte数组传送

  private String audioFormat = "pcm16k16bit";             // 音频格式，如pcm16k16bit，详见api文档或sdk文档
  private String property = "chinese_16k_common";       // 属性字符串，language_sampleRate_domain，如chinese_16k_common, 详见api文档

  /**
   * 一句话识别websocket版本参数设置，所有参数设置均为可选，均有默认值。用户根据需求设置参数。
   *
   * @param request request请求，包含各种参数
   */
  private void setParameters(SasrWebsocketRequest request) {
```

```
    // 1. 设置是否添加标点符号，yes 或 no， 默认"no"
    request.setAddPunc("yes");
    // 2. 设置是否显示中间结果，yes或no，默认“no”
    request.setIntermediateResult("no");
    // 3. 设置热词表id, 若没有则设置，否则会报错。
    // request.setVocabularyId("");
    // 4. 设置是否将音频中数字转写为阿拉伯数字，yes or no，默认yes
    request.setDigitNorm("no");
    // 5. 设置是否需要word_info，yes or no, 默认no
    request.setNeedWordInfo("no");
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new
ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }


  /**
   * 获取监听器，监听器的监听函数。
   *
   * @return RasrListener，用于监听websocket
   */
  private RasrListener getRasrListener() {
    RasrListener rasrListener = new RasrListener() {
      @Override
      /**
       * 连接成功回调
       */
      public void onTranscriptionConnect() {
        System.out.println("sasr websocket connected");
      }

      @Override
      /**
       * 断开连接回调
       */
      public void onTranscriptionClose() {
        System.out.println("sasr websocket closed");
      }

      @Override
      /**
       * 响应结果回调
       */
      public void onTranscriptionResponse(RasrResponse response) {
        printResponse(response);
      }

      @Override
      /**
       * 识别开始回调
       */
      public void onTranscriptionBegin(StateResponse response) {
        printResponse(response);
      }
```

```
    @Override
    /**
     * 识别结束回调
     */
    public void onSTranscriptionEnd(StateResponse response) {
      printResponse(response);
    }

    @Override
    /**
     * 识别出错回调
     */
    public void onTranscriptionFail(StateResponse response) {
      printResponse(response);
    }

    @Override
    public void onEvent(String event) {
      log.info("receive event {}", event);
    }
  };
  return rasrListener;
}


private void printResponse(Object response) {
  try {
    System.out.println(JsonUtils.obj2Str(response, true));
  } catch (Exception e) {
    e.printStackTrace();
  }
}

/**
 * 实时语音识别SDK的工作流程
 */
private void process() {
  // 1. 实现监听器接口listener，用户自定义收到响应的处理逻辑。
  RasrListener listener = getRasrListener();

  // 2. 初始化SasrWebsocketClient
  AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
  SasrWebsocketClient sasrWebsocketClient = new SasrWebsocketClient(authInfo, listener, getConfig());
  try {

    // 3. 配置参数
    // audioFormat为支持格式、property为属性字符串，具体填写请详细参考api文档
    SasrWebsocketRequest request = new SasrWebsocketRequest(audioFormat, property);
    setParameters(request);

    // 4 连接websocket
    sasrWebsocketClient.sasrConnect(request);
    // 设置企业id, 可选
    // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
    // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
    // sasrWebsocketClient.sasrConnect(headers, request);

    // 5. 发送开始请求、发送音频、发送end请求
    // 发送开始请求，即将开始请求连带配置发送至服务端
    sasrWebsocketClient.sendStart();

    // 也可以自己控制发送速率.byteLen为每次发送大小，sleepTime为每次发送后睡眠时间(ms)，一些非持续获取音频场景不需要睡眠，可设置为0.
    sasrWebsocketClient.sendAudio(path);
    // sasrWebsocketClient.sendAudio(path, byteLen, sleepTime);

    // 可直接发送byte流,即byte数组
    // byte[] data = IOUtils.getFileData(path);
```

```
        // sasrWebsocketClient.sendByte(data);
        // sasrWebsocketClient.sendByte(data, byteLen, sleepTime);

        // 发送结尾请求
        sasrWebsocketClient.sendEnd();


      } catch (Exception e) {
        e.printStackTrace();

      } finally {
        // 6. 关闭客户端。发送完毕后，此步一定要实施，否则服务端因为20s没有接收任何消息而报异常。
        sasrWebsocketClient.close();
      }

    }

    public static void main(String[] args) {
      SasrWebsocketDemo sasrWebsocketDemo = new SasrWebsocketDemo();
      sasrWebsocketDemo.process();
    }
}
```

6.3 录音文件识别

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 确保已存在待识别的音频文件并上传OBS或者有公网可访问服务器上（需保证可使用域名访问），示例音频可参考下载SDK压缩包文件。如果音频存放在OBS上，确保服务已授权访问OBS，可参考**配置OBS服务**。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化AsrCustomizationClient，其参数包括AuthInfo和SisConfig。

表 6-15 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

表 6-16 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为AsrCustomLongRequest，详见**表6-17**。

表 6-17 AsrCustomLongRequest

参数名称	是否必选	参数类型	描述
dataUrl	是	String	存放录音文件地址： ● 推荐使用华为云OBS：授权配置请参见**OBS配置**。 ● 您也可以把录音文件放在自行搭建服务器上，提供下载文件的地址。URL不能使用IP地址，只能使用域名，请尽量避免中文。
audioFormat	是	String	音频格式，具体信息请参见《API参考》中**录音文件识别**章节。
property	是	String	属性字符串，语言_采样率_模型，如chinese_8k_common。具体信息请参见《API参考》中**录音文件识别**章节。
addPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
callbackUrl	否	String	表示回调 url，用户用于接收识别结果的服务器地址， 不支持ip方式调用，url长度小于2048字节。服务请求方法为Post方式，请求体为Json格式。 ● 如果用户使用回调方式获取识别结果，需提交该参数，处理成功后用户服务器需返回状态码为200。 ● 如果用户使用轮询方式获取识别结果，则无需提交该参数。

参数名称	是否必选	参数类型	描述
needAnalysisInfo	否	Boolean	是否选择分析信息。当前仅对8k模型有效。 如果选择false，则声道、话者分离、情绪检测、速度信息均无效。默认false。
diarization	否	Boolean	是否需要话者分离，表示识别结果会包含role项，默认true。
channel	否	String	语音文件声道信息，可以为MONO（缺省)、LEFT_AGENT、RIGHT_AGENT。
emotion	否	Boolean	是否需要做情绪检测，默认true。
speed	否	Boolean	是否需要输出语速信息，默认true。
vocabularyId	否	String	热词表id，不使用则不填写。 创建热词表请参考《API参考》中**创建热词表**章节。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

响应参数

响应类为AsrCustomLongResponse，详见**表6-18**。调用失败处理方法请参见**错误码**。

表 6-18 AsrCustomLongResponse

参数名称	是否必选	参数类型	描述
status	是	String	描述返回状态。 • WAITING 等待识别。 • FINISHED识别已经完成。 • ERROR 识别过程中发生错误。
createTime	否	String	任务创建时间。格式如2018-12-04T13:10:29.310Z。
startTime	否	String	开始识别时间。格式如2018-12-04T13:10:29.310Z。
finishTime	否	String	识别完成时间。格式如2018-12-04T13:10:29.310Z。
audioDuration	否	Integer	提交音频时长，单位ms。
segments	否	Array of objects	识别结果, 多句结果的数组。 数据结构参见**表6-19**。

表 6-19 Segment

参数名	是否必选	参数类型	说明
start_time	是	Integer	一句的起始时间戳，单位ms。
end_time	是	Integer	一句的结束时间戳，单位ms。
result	是	Object	调用成功表示识别结果，调用失败时无此字段。数据结构参见表6-20。

表 6-20 Result

参数名	是否必选	参数类型	说明
text	是	String	识别结果文本。
analysis_info	否	Object	每一句的质检分析结果对象。 仅在识别配置中的need_analysis_info不为null时存在该返回结果。数据结构参见表6-21。
word_info	否	Array of Object	分词输出列表。

表 6-21 Analysisinfo

参数名	是否必选	参数类型	说明
role	否	String	角色类型，目前仅支持 AGENT（座席），USER（用户）。
emotion	否	String	情绪类型，目前仅支持NORMAL（正常），ANGRY（愤怒）。 在识别配置中emotion为true时存在。
speed	否	Float	语速信息，单位是每秒字数。 在识别配置中speed为true时存在。

表 6-22 Word_info 数据结构

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.AsrCustomLongRequest;
import com.huawei.sis.bean.response.AsrCustomLongResponse;
import com.huawei.sis.bean.request.AsrCustomShortRequest;
import com.huawei.sis.bean.response.AsrCustomShortResponse;
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.client.AsrCustomizationClient;
import com.huawei.sis.exception.SisException;
import com.huawei.sis.util.IOUtils;
import com.huawei.sis.util.JsonUtils;

/**
 * 录音文件识别Demo
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class AsrCustomizationDemo {
  private static final int SLEEP_TIME = 500;
  private static final int MAX_POLLING_NUMS = 1000;

  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "";    // 区域，如cn-north-1、cn-north-4
  private String projectId = ""; // 项目id。登录管理控制台，鼠标移动到右上角的用户名上，在下拉列表中选择我的凭证，在项目列表中查看项目id。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。

  /**
   *   todo 请正确填写音频格式和模型属性字符串
   *   1. 音频格式一定要相匹配。
   *      例如obs url是xx.wav, 则在录音文件识别格式是auto。
   *      例如音频是pcm格式，并且采样率为8k，则格式填写pcm8k16bit。
   *      如果返回audio_format is invalid 说明该文件格式不支持。
   *
   *   2. 音频采样率要与属性字符串的采样率要匹配。
   *      例如格式选择pcm16k16bit，属性字符串却选择chinese_8k_common, 则属于采样率不匹配。
   *      例如wav本身是16k采样率，属性选择chinese_8k_common, 同样属于采样率不匹配。
   *
   *   3. 用户可以通过使用热词，识别专业术语，增加语句识别准确率。
   */

  // 录音文件识别参数
  private String obsUrl = "";          // 音频文件OBS链接，录音文件识别目前仅支持传入OBS音频连接，或公网可访问url
  private String obsAudioFormat = "";  // 文件格式，如auto等
  private String obsProperty = "";     // 属性字符串，如chinese_8k_common等
  /**
   * 设置录音文件识别参数，所有参数均有默认值，不配置也可使用
   *
   * @param request 录音文件识别请求
   */
  private void setLongParameter(AsrCustomLongRequest request) {
    // 设置否是添加标点，yes 或no， 默认是no
    request.setAddPunc("yes");
    // 设置是否将语音中的数字转写为阿拉伯数字，yes或no，默认yes
    request.setDigitNorm("no");
    // 设置声道，MONO/LEFT_AGENT/RIGHT_AGENT, 默认是单声道MONO
    request.setChannel("MONO");
    // 设置是否需要分析，默认为false。当前仅支持8k采样率音频。当其设置为true时，话者分离、情绪检测，速
```

```
度、声道才生效。
    request.setNeedAnalysis(true);
    // 设置是否需要话者分离，若是，则识别结果包含role，默认true
    request.setDirization(true);
    // 设置是否需要情绪检测，默认true。
    request.setEmotion(true);
    // 设置是否需要速度。默认true。
    request.setSpeed(true);
    // 设置回调地址，设置后音频转写结果将直接发送至回调地址。请务必保证地址可连通,不支持ip地址。
    // request.setCallbackUrl("");
    // 设置热词id，不使用则不用填写
    // request.setVocabularyId("");
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new
ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }


  /**
   * 录音文件识别demo
   */
  private void longDemo() {
    try {
      // 1. 初始化AsrCustomizationClient
      // 定义authInfo，根据ak，sk，region,projectId.
      AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
      // 设置config，主要与超时有关
      SisConfig config = getConfig();
      // 根据authInfo和config，构造AsrCustomizationClient
      AsrCustomizationClient asr = new AsrCustomizationClient(authInfo, config);

      // 2. 生成请求
      AsrCustomLongRequest request = new AsrCustomLongRequest(obsUrl, obsAudioFormat, obsProperty);
      // 设置请求参数，所有参数均为可选
      setLongParameter(request);

      // 3. 提交任务，获取jobId
      String jobId = asr.submitJob(request);
      // 设置企业id, 可选
      // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
      // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
      // String jobId = asr.submitJob(headers, request);


      // 4 轮询jobId，获取最终结果。
      int count = 0;
      int successFlag = 0;
      AsrCustomLongResponse response = null;
      while (count < MAX_POLLING_NUMS) {
        System.out.println("正在进行第" + count + "次尝试");
        response = asr.getAsrLongResponse(jobId);
        String status = response.getStatus();
        if (status.equals("FINISHED")) {
          successFlag = 1;
```

```
        break;
      } else if (status.equals("ERROR")) {
        System.out.println("执行失败, 无法根据jobId获取结果");
        return;
      }
      try {
        Thread.sleep(SLEEP_TIME);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
      count++;
    }
    // 打印结果
    if (successFlag == 0) {
      System.out.println("已进行" + count + "次尝试，无法获取识别结果。 jobId为 " + jobId);
      return;
    }

    System.out.println(JsonUtils.obj2Str(response, true));
  } catch (SisException e) {
    e.printStackTrace();
    System.out.println("error_code:" + e.getErrorCode() + "\nerror_msg:" + e.getErrorMsg());
  }
 }

 public static void main(String[] args) {
  AsrCustomizationDemo demo = new AsrCustomizationDemo();
  // 录音文件识别
  demo.longDemo();
 }
}
```

回调服务示例

如果用户选择使用回调方式获取识别结果，假设用户设置的回调地址是https://address/v1/callback,这里以spring为示例。

1. 新建请求类

```
import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import java.util.List;

/**
 * 查询长语音转写结果
 */
@Getter
@Setter
@JsonIgnoreProperties(ignoreUnknown = true)
@ToString
public class QueryTranscriptionResp {
  @JsonProperty("job_id")
  private String jobId;

  @JsonProperty("status")
  private String status;

  @JsonProperty("create_time")
  private String createTime;

  @JsonProperty("start_time")
  private String startTime;

  @JsonProperty("finish_time")
```

```
    private String finishTime;

    @JsonProperty("segments")
    private List<Segment> segments;

    @JsonProperty("error_code")
    private String errorCode;

    @JsonProperty("error_msg")
    private String errorMsg;

    /**
     * Segments
     */
    @Getter
    @Setter
    @JsonIgnoreProperties(ignoreUnknown = true)
    public static class Segment {
      @JsonProperty("start_time")
      private long startTime;

      @JsonProperty("end_time")
      private long endTime;

      @JsonProperty("result")
      private Result result;
    }

    /**
     * Result
     */
    @Getter
    @Setter
    @JsonIgnoreProperties(ignoreUnknown = true)
    public static class Result {
      @JsonProperty("text")
      private String text;

      @JsonProperty("score")
      private double score;

      @JsonProperty("analysis_info")
      private AnalysisInfo analysisInfo;

      @JsonProperty("word_info")
      private List<WordInfo> wordInfo;
    }

    /**
     * AnalysisInfo
     */
    @Getter
    @Setter
    @JsonIgnoreProperties(ignoreUnknown = true)
    public static class AnalysisInfo {
      @JsonProperty("role")
      private String role;

      @JsonProperty("emotion")
      private String emotion;

      @JsonProperty("speed")
      private Double speed;
    }

    /**
     * WordInfo
     */
    @Getter
```

```
  @Setter
  @JsonIgnoreProperties(ignoreUnknown = true)
  public static class WordInfo {
    @JsonProperty("start_time")
    private Integer startTime;

    @JsonProperty("end_time")
    private Integer endTime;

    @JsonProperty("word")
    private String word;

  }
}
```

2. 新建Controller

```
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.util.StringUtils;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
public class CallbackController {

  @PostMapping("/v1/callback")
  public ResponseEntity<?> callback(@RequestBody QueryTranscriptionResp queryTranscriptionResp) {
    if (!StringUtils.isEmpty(queryTranscriptionResp.getErrorCode())
        || queryTranscriptionResp.getStatus().equals("ERROR")) {
      System.out.println("receive error resp"+queryTranscriptionResp.toString());
      return new ResponseEntity<>("error resp", HttpStatus.BAD_REQUEST);
    }
    List<QueryTranscriptionResp.Segment> segments = queryTranscriptionResp.getSegments();
    for (QueryTranscriptionResp.Segment segment : segments) {
      QueryTranscriptionResp.Result result = segment.getResult();
      System.out.println("result: " + result.getText());
    }
    return new ResponseEntity<>("", HttpStatus.OK);
  }
}
```

6.4 实时语音识别

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RasrClient，其参数包括AuthInfo、RasrListener、SisConfig。

RasrListener需要用户自定义实现监听逻辑，请参见**表6-23**和**表6-24**。

表 6-23 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

表 6-24 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为RasrRequest，详见**表6-25**。

表 6-25 RasrRequest

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，支持pcm，alaw，ulaw等，如pcm8k16bit，参见《API参考》中**开始识别**章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_16k_general，参见《API参考》中**开始识别**章节。
punc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vadHead	否	Integer	头部最大静音时间，[0, 60000]，默认10000ms。
vadTail	否	Integer	尾部最大静音时间，[0, 3000]，默认500ms。

参数名称	是否必选	参数类型	描述
maxSeconds	否	Integer	音频最长持续时间， [1, 60]，默认30s。
intermediateResult	否	String	是否显示中间结果，yes 或 no，默认no。例如分3次发送音频，选择no结果一次性返回，选择yes分三次返回。
vocabularyId	否	String	热词表id，若没有则不填。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

响应参数

状态响应类为StateResponse，详见**表6-26**。

结果响应类为RasrResponse，详见**表6-27**。

调用失败处理方法请参见**错误码**。

表 6-26 StateResponse

参数名称	是否必选	参数类型	描述
state	是	String	识别状态，包括start、end、fail。
traceId	是	String	用于日志问题追溯。
description	是	String	状态描述。

表 6-27 RasrResponse

参数名	参数类型	说明
resp_type	String	参数值为RESULT，表示识别结果响应。
trace_id	String	服务内部的令牌，可用于在日志中追溯具体流程。
segments	Array of objects	多句结果。 请参考**表6-28**。

表 6-28 Segment

参数名	参数类型	说明
start_time	Integer	一句的起始时间戳，单位为ms。
end_time	Integer	一句的结束时间戳，单位为ms。
is_final	Boolen	true表示是最终结果， false表示为中间临时结果。
result	Object	调用成功表示识别结果，调用失败时无此字段。 请参考**表6-29**。

表 6-29 Result

参数名	参数类型	说明
text	String	识别结果。
score	Float	识别结果的置信度，取值范围：0~1。此值仅会在最终结果时被赋值，在中间结果时统一置为“0.0”。 **说明** 目前置信度作用不是太大，请勿过多依赖此值。
word_info	Array of Object	分词输出列表。

表 6-30 Word_info 数据结构

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.RasrRequest;
import com.huawei.sis.bean.response.RasrResponse;
import com.huawei.sis.bean.response.StateResponse;
import com.huawei.sis.client.RasrClient;
import com.huawei.sis.bean.RasrListener;
import com.huawei.sis.util.JsonUtils;
```

```
/**
 * 实时语音识别demo
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class RasrDemo {
  private static final int DEFAULT_HEAD_SILENCE_TIME = 1000;
  private static final int DEFAULT_TAIL_SILENCE_TIME = 500;
  private static final int DEFAULT_CONTINUE_SECONDS = 30;

  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "";         // 区域，如cn-north-1、cn-north-4
  private String projectId = "";      // 项目id。登录管理控制台，鼠标移动到右上角的用户名上，在下拉列表中选择我的凭证，在项目列表中查看项目id。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。

  private String path = "";           // 本地音频路径，如D:/test.wav, 也可将音频文件、音频流转换为byte数组后进行传送。

  private String audioFormat = "";    // 音频格式，如pcm16k16bit
  private String property = "chinese_16k_general";       // 属性字符串，language_sampleRate_domain，16模型推荐使用chinese_16k_general

  /**
   * 实时语音识别参数设置，所有参数设置均为可选，均有默认值。用户根据需求设置参数。
   *
   * @param request request请求，包含各种参数
   */
  private void setParameters(RasrRequest request) {

    // 1. 设置是否添加标点符号，yes 或 no， 默认"no"
    request.setAddPunc("yes");
    // 2. 设置头部的最大静音时间，[0,60000], 默认10000ms
    request.setVadHead(DEFAULT_HEAD_SILENCE_TIME);
    // 3. 设置尾部最大静音时间，[0, 3000], 默认500ms，
    request.setVadTail(DEFAULT_TAIL_SILENCE_TIME);
    // 4. 设置最长持续时间，仅在continue-stream，sentence-stream模式下起作用，[1, 60], 默认30s
    request.setMaxSeconds(DEFAULT_CONTINUE_SECONDS);
    // 5. 设置是否显示中间结果，yes或no，默认“no”。例如分3次发送音频，选择no结果一次性返回，选择yes分三次返回。
    request.setIntermediateResult("no");
    // 6. 设置热词表id, 若没有则设置，否则会报错。
    // request.setVocabularyId("");
    // 7. 设置是否将音频中数字转写为阿拉伯数字，yes or no，默认yes
    request.setDigitNorm("no");
    // 8. 设置是否需要word_info，yes or no, 默认no
    request.setNeedWordInfo("no");
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置pingInterval，默认5000ms，当并发较大时，建议把此值设置大一些。如果不需要ping，可设置为-1
    // config.setPingInterval(-1);
```

```
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new
ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }


  /**
   * 获取监听器，监听器的监听函数。
   *
   * @return RasrListener，用于监听实时语音识别的开始、识别结果、结束以及失败响应
   */
  private RasrListener getRasrListener() {
    RasrListener rasrListener = new RasrListener() {
      @Override
      /**
       * 连接成功回调
       */
      public void onTranscriptionConnect() {
        System.out.println("websocket connected");
      }

      @Override
      /**
       * 断开连接回调
       */
      public void onTranscriptionClose() {
        System.out.println("websocket closed");
      }

      @Override
      /**
       * 响应结果回调
       */
      public void onTranscriptionResponse(RasrResponse response) {
        printResponse(response);
      }

      @Override
      /**
       * 识别开始回调
       */
      public void onTranscriptionBegin(StateResponse response) {
        printResponse(response);
      }

      @Override
      /**
       * 识别结束回调
       */
      public void onSTranscriptionEnd(StateResponse response) {
        printResponse(response);
      }

      @Override
      /**
       * 识别出错回调
       */
      public void onTranscriptionFail(StateResponse response) {
        printResponse(response);
      }

      @Override
      public void onVoiceStart() {
        log.info("voice start event");
      }

      @Override
```

```
    public void onVoiceEnd() {
      log.info("voice end event");
    }

    @Override
    public void onExcceededSilence() {
      log.error("exceeded silence event");
    }

    @Override
    public void onExceededAudio() {
      log.error("exceeded audio event");
    }

    @Override
    public void onEvent(String event) {
      log.warn("receive event {}", event);
    }
  };
  return rasrListener;
}


private void printResponse(Object response) {
  try {
    System.out.println(JsonUtils.obj2Str(response, true));
  } catch (Exception e) {
    e.printStackTrace();
  }
}

/**
 * 实时语音识别SDK的工作流程
 */
private void process() {
  // 1. 实现监听器接口RasrListener，用户自定义收到响应的处理逻辑。
  RasrListener rasrListener = getRasrListener();

  // 2. 初始化RasrClient
  AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
  RasrClient rasrClient = new RasrClient(authInfo, rasrListener, getConfig());
  try {


    // 3. 配置参数
    // audioFormat为支持格式、property为属性字符串
    RasrRequest request = new RasrRequest(audioFormat, property);
    setParameters(request);

    // 4 选择连接模式，目前实时语音识别提供三种接口，流式一句话、实时语音识别连续模式、实时语音识别单句模式
    // 选择1 流式一句话连接
    // rasrClient.shortStreamConnect(request);

    // 选择2，实时语音识别单句模式
    // rasrClient.sentenceStreamConnect(request);

    // 选择3，实时语音识别连续模式
    rasrClient.continueStreamConnect(request);
    // 设置企业id, 可选
    // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
    // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
    // rasrClient.continueStreamConnect(headers, request);

    // 5. 发送开始请求、发送音频、发送end请求
    // 发送开始请求，即将开始请求连带配置发送至服务端
    rasrClient.sendStart();

    // 发送数据，在实时语音连续模式下可多次发送。识别结果可以通过监听器获取
```

```
    // 可以自己控制发送速率.byteLen为每次发送大小，sleepTime为每次发送后睡眠时间(ms)，一些非持续获取音频场景不需要睡眠，可设置为0.
      rasrClient.sendAudio(path, 3200, 200);
      // rasrClient.sendAudio(path);
      // 可直接发送byte流,即byte数组
      // byte[] data = IOUtils.getFileData(path);
      // rasrClient.sendByte(data);
      // rasrClient.sendByte(data, byteLen, sleepTime);

      // 发送结尾请求
      rasrClient.sendEnd();



    } catch (Exception e) {
      e.printStackTrace();

    } finally {
      // 6. 关闭客户端。发送完毕后，此步一定要实施，否则服务端因为20s没有接收任何消息而报异常。
      rasrClient.close();
    }

  }

  public static void main(String[] args) {
    RasrDemo rasrDemo = new RasrDemo();
    rasrDemo.process();
  }
}
```

6.5 语音合成

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化TtsCustomizationClient，其参数包括AuthInfo和SisConfig。

表 6-31 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。

表 6-32 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为TtsCustomRequest，详见**表6-33**。

表 6-33 TtsCustomRequest

参数名称	是否必选	参数类型	描述
text	是	String	待合成的文本。
audio_format	否	String	待合成的音频格式，可选mp3，wav等，默认wav。具体信息请参见《API参考》中**语音合成**章节。
pitch	否	Integer	音高，[-500,500]，默认是0。
speed	否	Integer	语速，[-500,500]，默认是0。
volume	否	Integer	音量，[0,100]，默认是50。
sample_rate	否	String	采样率，支持“8000”、“16000”，默认“8000”。
property	否	String	特征字符串，{language}_{speaker}_{domain}，默认chinese_xiaoqi_common。具体信息请参见《API参考》中**语音合成**章节。
isSaved	否	Boolean	是否选择合成的音频数据保存到本地，默认不保存。
savePath	否	String	选择保存到本地的路径。路径需具体到文件，如D:/test.wav。

响应参数

响应类为TtsCustomResponse，详见**表6-33**。调用失败处理方法请参见**错误码**。

表 6-34 TtsResponse

参数名	是否必选	参数类型	说明
isSaved	否	String	是否将响应音频保存为本地文件。
savePath	否	String	保存本地的路径，如D:/test.wav。
result	是	Object	调用成功时为合成语音内容，请参考表6-35。 调用失败时无此字段。

表 6-35 Result

参数名	是否必选	参数类型	说明
data	是	String	合成后生成的语音数据，以Base64编码格式返回。用户如需生成音频，需要将Base64编码解码成byte数组，再保存为wav音频。

代码示例

如下示例仅供参考，最新代码请前往SDK（websocket）章节获取并运行。

```
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.TtsCustomRequest;
import com.huawei.sis.bean.response.TtsCustomResponse;
import com.huawei.sis.client.TtsCustomizationClient;
import com.huawei.sis.exception.SisException;
import com.huawei.sis.util.JsonUtils;

/**
 * 语音合成的demo
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class TtsCustomizationDemo {

  private static final int DEFAULT_PITCH = 0;
  private static final int DEFAULT_SPEED = 0;
  private static final int DEFAULT_VOLUME = 50;

  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");
```

```
  private String region = "";    // 区域，如cn-north-1、cn-north-4
  private String projectId = ""; // 项目id。登录管理控制台，鼠标移动到右上角的用户名上，在下拉列表中选择我的凭证，在项目列表中查看项目id。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。

  private String text = "";      // 待合成的文本
  private String path = "";      // 设置本地音频保存路径.可选择不保存到本地。需具体到文件，如D:/test.wav


  /**
   * 用于语音合成参数设置，例如发声人、音高、语速、音量、采样率、连接超时。所有参数均可以不设置，采用默认。
   *
   * @param request 语音合成请求
   */
  private void setParameter(TtsCustomRequest request) {

    // 设置语音格式，可选MP3，pcm等，默认wav
    request.setAudioFormat("wav");
    // 音高，[-500, 500], 默认0
    request.setPitch(DEFAULT_PITCH);
    // 语速，[-500, 500]，默认0
    request.setSpeed(DEFAULT_SPEED);
    // 音量，[0, 100]，默认50
    request.setVolume(DEFAULT_VOLUME);
    // 当前支持8000和16000，默认8000
    request.setSampleRate("8000");
    // 设置property，特征字符串，{language}_{speaker}_{domain}
    request.setProperty("chinese_xiaoyu_common");

    // 设置返回数据是否保存，默认不保存。若保存，则需要设置一下保存路径，如D:/1.wav
    request.setSaved(true);
    request.setSavePath(path);
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }

  /**
   * 根据文本和api，获取生成的音频数据
   */
  private void ttsCustomDemo() {
    try {
      // 1. 初始化TtsCustomizationClient
      // 定义authInfo，根据ak，sk，region, projectId.
      AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
      // 定义config，所有参数可选，设置超时时间。
      SisConfig config = getConfig();
      // 根据authInfo和config，构造TtsCustomizationClient
      TtsCustomizationClient tts = new TtsCustomizationClient(authInfo, config);

      // 2. 配置请求
      TtsCustomRequest request = new TtsCustomRequest(text);
      // 设置参数，所有参数均可选，如果要保存合成音频文件，需要在request设置
      setParameter(request);
```

```
    // 3. 发送请求，获取响应。具体结果可通过response.getXX获取。
    TtsCustomResponse response = tts.getTtsResponse(request);
    // 设置企业id, 可选
    // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
    // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
    // TtsCustomResponse response = tts.getTtsResponse(headers, request);

    System.out.println(JsonUtils.obj2Str(response, true));

  } catch (SisException e) {
    e.printStackTrace();
    System.out.println("error_code:" + e.getErrorCode() + "\nerror_msg:" + e.getErrorMsg());
  }

 }

 public static void main(String[] args) {
   TtsCustomizationDemo demo = new TtsCustomizationDemo();
   demo.ttsCustomDemo();
 }

}
```

6.6 热词管理

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化HotWordClient，其参数包括AuthInfo和SisConfig。

表 6-36 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。

表 6-37 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为HotWordRequest，详见**表6-38**。

表 6-38 HotWordRequest

参数名称	是否必选	参数类型	描述
name	是	String	热词表名，创建时不可重复。内容限制为字母，数字，下中划线和井号，长度不超过32字节。
language	是	String	热词表语言类型，目前支持汉语普通话“chinese_mandarin”。
contents	是	Array of String	热词库，单词库支持热词数上限10000。中文单个热词长度上限32字节。
description	否	String	热词表描述，长度不超过255字节。

响应参数

创建热词响应参数为String，表示热词表ID。调用失败处理方法请参见**错误码**。

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.HotWordRequest;
import com.huawei.sis.bean.response.HotWordsResponse;
import com.huawei.sis.bean.response.HotWordResponse;
import com.huawei.sis.client.HotWordClient;
import com.huawei.sis.exception.SisException;
import com.huawei.sis.util.JsonUtils;

import java.util.ArrayList;
import java.util.List;

/**
```

```
 * 热词demo
 *
 * 热词可在一句话识别、录音文件识别、实时语音识别使用。例如将地名和人名作为热词，则语音可以准确识别出人名和地名。
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class HotWordDemo {
  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "";    // 区域，如cn-north-1、cn-north-4
  private String projectId = ""; // 项目id。登录管理控制台，鼠标移动到右上角的用户名上，在下拉列表中选择我的凭证，在项目列表中查看项目id。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。

  private String name = "test";      // 创建热词表时，需要确保热词表名之前未创建过。如 test1
  private String vocabularyId = "";  // 热词表id，仅在更新、查询，删除中使用,创建时不需要。使用前一定要确保热词表id已存在。
  private List<String> hotWordList = new ArrayList<>(); // 用于存放热词表，每个热词表最多存放10000个热词。如["计算机", "网络"]

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }

  /**
   * 1. 热词使用包含创建、更新、查询、删除等，一个用户可以创建多个热词表，一个热词表可以包含多个热词。一个vocabularyId对应一个热词表。
   * 2. 目前支持一个用户最多创建10个热词表，一个热词表最多包含10000个热词。
   */
  private void hotWordDemo() {
    try {
      // 初始化client
      AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
      SisConfig config = getConfig();
      HotWordClient hotWordClient = new HotWordClient(authInfo, config);

      // option 1 创建热词表,可生成热词表id
      // name表示热词表表名，创建热词表时不可和已有表名重复
      // hotWordList表示热词表信息，用于存放热词
      hotWordList.add("测试");
      HotWordRequest hotWordRequest = new HotWordRequest(name, hotWordList);
      // 可选，热词表描述信息
      hotWordRequest.setDescription("test");
      // 可选，热词语言，目前仅支持中文 chinese_mandarin。
      hotWordRequest.setLanguage("chinese_mandarin");
      vocabularyId = hotWordClient.create(hotWordRequest);
      System.out.println("成功创建热词表，热词表id为" + vocabularyId);

      // option 2 根据热词表id 更新热词表。新的热词表会替换旧的热词表。使用前需确保热词表id已存在。
      hotWordList.add("华为");
```

```
    HotWordRequest updateRequest = new HotWordRequest(name, hotWordList);
    String updateVocabularyId = hotWordClient.update(updateRequest, vocabularyId);
    System.out.println("成功更新热词表，热词表id为" + updateVocabularyId);

    // option 3 查看热词表列表
    HotWordsResponse hotWordListResponse = hotWordClient.query();
    // 打印响应，可通过getXX方法获取具体参数
    System.out.println(JsonUtils.obj2Str(hotWordListResponse, true));

    // option 4 根据热词表id 查询热词表，使用前需确保热词表id已存在。
    HotWordResponse hotWordResponse = hotWordClient.query(vocabularyId);
    // 打印响应，可通过getXX方法获取具体参数
    System.out.println(JsonUtils.obj2Str(hotWordResponse, true));

    // option 5 根据vocabularyId 删除热词表，使用前需确保热词表id已存在。
    // 删除无返回结果，无报错异常即表明删除成功。
    hotWordClient.delete(vocabularyId);
  } catch (SisException e) {
    e.printStackTrace();
    System.out.println("error_code:" + e.getErrorCode() + "\terror_msg:" + e.getErrorMsg());
  }

 }

 public static void main(String[] args) {
   HotWordDemo hotWordDemo = new HotWordDemo();
   hotWordDemo.hotWordDemo();
 }
}
```

6.7 实时语音合成

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RttsClient，其参数包括AuthInfo和SisConfig。

表 6-39 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。

表 6-40 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。
websocketWaitTimeout	否	Integer	websocket最大等待超时，默认20000，单位ms

请求参数

请求类为RttsRequest，详见**表6-41**。

表 6-41 RttsRequest

名称	参数类型	是否必选	说明
command	String	是	需设置为START，表示开始识别请求。
text	String	是	待合成的文本，文本长度限制小于500字符。
config	Object	否	配置信息。请参考**表 config数据结构**。

表 6-42 Config

名称	参数类型	是否必选	说明
audio_format	String	否	语音格式头：pcm、alaw、ulaw。 默认：pcm
sample_rate	String	否	采样率：16000、8000。 默认：8000

名称	参数类型	是否必选	说明
property	String	否	语音合成特征字符串，组成形式为{language}_{speaker}_{domain}，即“语种_人员标识_领域”。 ● language取值范围： – chinese ● speaker取值范围： – xiaoqi 正式女生 – xiaoyu正式男生 – xiaoyan情感女生 – xiaowang童声 ● speaker（精品发音人）取值范围： – huaxiaomei温柔女声发音人，仅支持pcm – huaxiaofei朝气男声发音人，仅支持pcm ● domain取值范围： – common，通用领域 默认：chinese_xiaoyan_common 实时语音合成和语音合成属于同一种资源，按次计费。实时语音合成普通发音人，每100字计一次。精品发音人每50字计一次。
speed	Integer	否	语速。 取值范围：-500~500 默认值：0
pitch	Integer	否	音高。 取值范围：-500~500 默认值：0
volume	Integer	否	音量。 取值范围：0~100 默认值：50

响应参数

响应类为RttsDataResponse，详见**表6-43**。调用失败处理方法请参见**错误码**。

表 6-43 RttsDataResponse

参数名	是否必选	参数类型	说明
data	是	Array of Byte	音频文件的byte数组。

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.cloud.sdk.util.StringUtils;
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.bean.RttsListener;
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.RttsRequest;
import com.huawei.sis.bean.response.RttsDataResponse;
import com.huawei.sis.bean.response.StateResponse;
import com.huawei.sis.client.RttsClient;
import com.huawei.sis.util.JsonUtils;


import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;

/**
 * 实时语音合成Demo
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class RttsDemo {
  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "";         // 区域，如cn-north-1、cn-north-4
  private String projectId = "";      // 项目id，在我的凭证查看。参考https://support.huaweicloud.com/api-sis/sis_03_0008.html

  private String text = "";           // 待合成的文本
  private String path = "";           // 合成音频存储的路径


  public static void main(String[] args) {
    RttsDemo rttsDemo = new RttsDemo();
    rttsDemo.process();
  }

  /**
   * 实时语音合成参数设置，所有参数设置均为可选，均有默认值。用户根据需求设置参数。
   */
  private RttsRequest getRttsRequest() {
    RttsRequest request = new RttsRequest();
    request.setCommand("START");
    // 设置待合成文本，文本长度1-500字
```

```
    request.setText(text);
    RttsRequest.Config config = new RttsRequest.Config();
    // 设置发音人属性，{language}_{speaker}_{domain}, 详见api文档
    config.setPorperty("chinese_xiaoyan_common");
    // 设置合成音频格式，默认pcm
    config.setAudioFormat("pcm");
    // 设置合成音频采样率，当前支持8000和16000，默认8000
    config.setSampleRate("8000");
    // 设置合成音频音量大小，取值0-100，默认50
    config.setVolume(50);
    // 设置合成音频音高大小，取值-500-500，默认0
    config.setPitch(0);
    // 设置合成音频语速大小，取值-500-500，默认0
    config.setSpeed(0);
    request.setConfig(config);
    return request;
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置websocket等待超时时间，默认20000ms
    config.setWebsocketWaitTimeout(SisConstant.DEFAULT_WEBSOCKET_WAIT_TIME);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new
ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }


  private void printResponse(Object response) {
    try {
      System.out.println(JsonUtils.obj2Str(response, true));
    } catch (Exception e) {
      e.printStackTrace();
    }
  }

  /**
   * 实时语音转写SDK的工作流程
   * 1. RttsClient只能发送一次文本，如有多个文本需发送，需要多次新建RttsClient实例
   * 2. 实时语音合成会多次收到音频响应，默认格式为pcm。在demo中会把多次返回的结果拼接起来，存入文件
中。
   * 3. 当服务端完成合成任务后，会返回end响应。
   */
  private void process() {
    // 1. 实现监听器接口RttsListener，用户自定义收到响应的处理逻辑。
    RttsListener rttsListener = new MyRttsListener(path);

    // 2. 初始化RttsClient,每个client只能发送一次text，如需发送多次text，需要建立多个client
    AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
    RttsClient rttsClient = new RttsClient(authInfo, rttsListener, getConfig());

    // 3. 配置参数
    // audioFormat为支持格式、property为属性字符串，具体填写请详细参考api文档
    RttsRequest request = getRttsRequest();

    // 4. 发送待合成文本，等待结果
    try {
      rttsClient.synthesis(request);
```

```
    // 设置企业id, 可选
    // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
    // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
    // rttsClient.synthesis(headers, request);

  } catch (Exception e) {
    e.printStackTrace();
  }


}

public class MyRttsListener implements RttsListener {
  private String path;
  private FileOutputStream fos = null;

  public MyRttsListener() {
    super();
  }

  public MyRttsListener(String path) {
    this.path = path;
  }

  @Override
  public void onTranscriptionResponse(RttsDataResponse rttsDataResponse) {
    System.out.println("receive binary data " + rttsDataResponse.getData().length);
    if (fos == null) {
      return;
    }
    try {
      fos.write(rttsDataResponse.getData());
    } catch (IOException e) {
      e.printStackTrace();
    }

  }

  @Override
  public void onTranscriptionBegin(StateResponse response) {
    printResponse(response);
    try {
      if (StringUtils.isNullOrEmpty(path)) {
        return;
      }
      File f = new File(path);
      fos = new FileOutputStream(f);
    } catch (IOException e) {
      e.printStackTrace();
    }

  }

  @Override
  public void onSTranscriptionEnd(StateResponse response) {
    printResponse(response);
    close();
  }

  @Override
  public void onTranscriptionFail(StateResponse response) {
    printResponse(response);
    close();

  }

  private void close() {
    if (fos == null) {
      return;
```

```
    }
    try {
      fos.close();
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
 }
}
```

6.8 录音文件极速版

前提条件

- 确保已按照**配置Java环境**配置完毕。
- 确保已存在待识别的音频文件并上传OBS，示例音频可参考下载SDK压缩包文件，同时确保服务已授权访问OBS，可参考**配置OBS服务**。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化FlashLasrClient，其参数包括AuthInfo和SisConfig。

表 6-44 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

表 6-45 SisConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。

请求参数

请求类为FlashLasrRequest，详见表6-46。

表 6-46 FlashLasrRequest

参数	是否必选	参数类型	描述
audio_format	是	String	支持语音的格式，请参考表6-47。
property	是	String	所使用的模型特征串，通常是 “语种_采样率_领域”的形式，采样率需要与音频采样率保持一致，取值范围请参考表6-48。
add_punc	否	String	表示是否在识别结果中添加标点，取值为“yes”和“no”，默认为“no”。
digit_norm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为“yes” 和 “no”，默认为“yes”。
vocabulary_id	否	String	热词表id，不使用则不填写。 创建热词表信息请参考创建热词表。
need_word_info	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。
first_channel_only	否	String	表示是否在识别中只识别首个声道的音频数据，取值为“yes”和“no”，默认为“no”。
obs_bucket_name	否	String	表示在OBS对象桶名，使用前请先授权，操作方法请参见配置OBS访问权限。 obs_bucket_name长度大于等于3个字符，小于64个字符，不需要进行urlencode编码，如果包含中文，直接输入中文即可。 示例 obs url为https://test.obs.cn-north-4.myhuaweicloud.com/data/0601/test.wav 则obs_bucket_name=test，obs_bucket_key=data/0601/test.wav

参数	是否必选	参数类型	描述
obs_object_key	否	String	表示OBS对象桶中的对象的键值，长度小于1024个字符，不需要进行urlencode编码，如果包含中文，直接输入中文即可。 示例 obs url为https://test.obs.cn-north-4.myhuaweicloud.com/data/0601/test.wav 则obs_bucket_name=test，obs_bucket_key=data/0601/test.wav

表 6-47 audio_format

audio_format取值	描述
wav	wav格式音频
mp3	mp3格式音频
m4a	m4a格式音频
aac	aac格式音频
opus	ops格式音频。

表 6-48 property

property取值	描述
chinese_8k_common	支持采样率为8k的中文普通话语音识别。
chinese_16k_conversation	支持采样率为16k的会议场景的中文普通话语音识别。

响应参数

响应类为FlashLasrResponse, 详见**表6-49**。调用失败处理方法请参见**错误码**。

表 6-49 FlashLasrResponse

参数	是否必选	参数类型	描述
trace_id	是	String	可用于在日志中追溯具体流程，调用失败无此字段。 在某些错误情况下可能没有此令牌字符串。
audio_duration	是	Integer	音频时长，单位毫秒
flash_result	是	Array of **FlashResult** objects	调用成功表示识别结果，调用失败时无此字段。

表 6-50 FlashResult

参数	是否必选	参数类型	描述
channel_id	否	Integer	声道Id
sentences	否	Array of **Sentences** objects	分句信息列表

表 6-51 Sentences

参数	是否必选	参数类型	描述
start_time	否	Integer	一句话开始时间，单位毫秒
result	否	**Result** object	分句结果信息
end_time	否	Integer	一句话结束时间，单位毫秒

表 6-52 Result

参数	是否必选	参数类型	描述
text	是	String	调用成功表示识别出的内容。
score	是	Double	调用成功表示识别出的置信度（0-1之间）。
word_info	否	Array of **WordInfo** objects	分词信息列表

表 6-53 WordInfo

参数	是否必选	参数类型	描述
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import com.huawei.sis.bean.AuthInfo;
import com.huawei.sis.bean.SisConfig;
import com.huawei.sis.bean.SisConstant;
import com.huawei.sis.bean.request.FlashLasrRequest;
import com.huawei.sis.bean.response.FlashLasrResponse;
import com.huawei.sis.client.FlashLasrClient;
import com.huawei.sis.exception.SisException;
import com.huawei.sis.util.JsonUtils;

/**
 * 录音文件极速版Demo
 *
 * Copyright 2021 Huawei Technologies Co.,Ltd.
 */
public class FlashLasrDemo {
  // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
  // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
  private String ak = System.getenv("HUAWEICLOUD_SDK_AK");
  private String sk = System.getenv("HUAWEICLOUD_SDK_SK");


  private String region = "";    // 区域，如cn-north-1、cn-north-4
  private String projectId = ""; // 项目id，在我的凭证查看。参考https://support.huaweicloud.com/api-sis/sis_03_0008.html

  private String obsBucketName = "";      // obs桶名
  private String obsObjectKey = "";       // obs对象的key
  private String audioFormat = "";        // 文件格式，如wav等， 支持格式详见api文档
  private String property = "";           // 属性字符串，language_sampleRate_domain, 如chinese_8k_common, 详见api文档


  /**
   * 设置录音文件识别极速版参数
   *
   * @param request 录音文件极速版请求
   */
  private void setShortParameter(FlashLasrRequest request) {
    // 以下参数必选
    // 设置桶名，必选
    request.setObsBucketName(obsBucketName);
    // 设置桶内对象名，必选
    request.setObsObjectKey(obsObjectKey);
    // 设置格式，必选
    request.setAudioFormat(audioFormat);
    // 设置属性，必选
    request.setProperty(property);

    // 以下参数可选
```

```
    // 设置是否添加标点，默认是no
    request.setAddPunc("yes");
    // 设置热词id，详见api文档，若热词id不存在，则会报错
//     request.setVocabularyId("");
    // 设置是否将音频中数字转写为阿拉伯数字，yes or no，默认yes
    request.setDigitNorm("no");
    // 设置是否需要word_info，yes or no, 默认no
    request.setNeedWordInfo("no");
    // 设置是否只识别首个声道的音频数据，默认no
    request.setFirstChannelOnly("no");
  }

  /**
   * 定义config，所有参数可选，设置超时时间等。
   *
   * @return SisConfig
   */
  private SisConfig getConfig() {
    SisConfig config = new SisConfig();
    // 设置连接超时，默认10000ms
    config.setConnectionTimeout(SisConstant.DEFAULT_CONNECTION_TIMEOUT);
    // 设置读取超时，默认10000ms
    config.setReadTimeout(SisConstant.DEFAULT_READ_TIMEOUT);
    // 设置代理, 一定要确保代理可用才启动此设置。 代理初始化也可用不加密的代理，new
ProxyHostInfo(host, port);
    // ProxyHostInfo proxy = new ProxyHostInfo(host, port, username, password);
    // config.setProxy(proxy);
    return config;
  }

  /**
   * 录音文件极速版demo。
   */
  private void flashLasrDemo() {
    try {

      // 1. 初始化FlashLasrClient
      // 定义authInfo，根据ak，sk，region，projectId
      AuthInfo authInfo = new AuthInfo(ak, sk, region, projectId);
      // 设置config，主要与超时有关
      SisConfig config = getConfig();
      // 根据authInfo和config，构造FlashLasrClient
      FlashLasrClient flashLasrClient = new FlashLasrClient(authInfo, config);

      // 2. 配置请求
      FlashLasrRequest request = new FlashLasrRequest();
      setShortParameter(request);

      // 3. 发送请求，获取响应。具体结果可通过response.getXX获取。
      FlashLasrResponse response = flashLasrClient.getFlashLasrResponse(request);
       // 设置企业id, 可选
      // Map<String, String> headers = OKHttpClientUtils.getJsonHeaders();
      // headers.put(SisConstant.ENTERPRISE_PROJECT_ID_KEY, "your enterprise_id");
      System.out.println(JsonUtils.obj2Str(response, true));
    } catch (SisException e) {
      e.printStackTrace();
      System.out.println("error_code:" + e.getErrorCode() + "\nerror_msg:" + e.getErrorMsg());
    }
  }


  public static void main(String[] args) {
    FlashLasrDemo demo = new FlashLasrDemo();
    demo.flashLasrDemo();
  }
}
```

7 Python SDK

7.1 一句话识别 Http 接口

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化AsrCustomizationClient详见**表 AsrCustomizationClient初始化参数**。

表 7-1 AsrCustomizationClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
service_endpoint	否	String	终端节点，一般使用默认即可。
sis_config	否	Object	详见**表7-2**。

表 7-2 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	Integer	连接超时，默认10，单位s。
read_timeout	否	Integer	读取超时，默认10，单位s。
proxy	否	List	[host, port] 或 [host, port, username, password]。

请求参数

请求类为AsrCustomShortRequest，详见**表7-3**。

表 7-3 AsrCustomShortRequest

参数名称	是否必选	参数类型	描述
data	是	String	本地音频文件经过Base64编码后的字符串，音频文件时长不超过1min。
audio_format	是	String	音频格式，具体信息请参见《API参考》中**一句话识别**章节。
model_property	是	String	属性字符串，语言_采样率_模型，如chinese_16k_general。具体信息请参见《API参考》中**一句话识别**章节。
add_punc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digit_norm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vocabulary_id	否	String	热词表id，不使用则不填写。 创建热词表请参考《API参考》中**创建热词表**章节。
need_word_info	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

响应参数

Python SDK响应结果为Json格式，**表7-4**。调用失败处理方法请参见**错误码**。

表 7-4 响应结果

参数名称	是否必选	参数类型	描述
result	是	Object	详见表7-5。
trace_id	是	String	用于后台日志问题追溯。

表 7-5 Result

参数名称	是否必选	参数类型	描述
text	是	String	识别结果。
score	是	Float	识别结果置信度评分。
word_info	否	Array of objects	分词信息列表。

表 7-6 Word_info 数据结构

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往SDK（websocket）章节获取并运行。

```
# -*- coding: utf-8 -*-
from huaweicloud_sis.client.asr_client import AsrCustomizationClient
from huaweicloud_sis.bean.asr_request import AsrCustomShortRequest
from huaweicloud_sis.exception.exceptions import ClientException
from huaweicloud_sis.exception.exceptions import ServerException
from huaweicloud_sis.utils import io_utils
from huaweicloud_sis.bean.sis_config import SisConfig
import json
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK。
ak = os.getenv("HUAWEICLOUD_SIS_AK")            # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")            # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/
```

```
sis_03_0008.html
region = ''        # region，如cn-north-4
"""
    todo 请正确填写音频格式和模型属性字符串
    1. 音频格式一定要相匹配。
        例如wav音频，格式是wav。具体参考api文档。
        例如音频是pcm格式，并且采样率为8k，则格式填写pcm8k16bit。
        如果返回audio_format is invalid 说明该文件格式不支持。具体支持哪些音频格式，需要参考一些api文
档。
    2. 音频采样率要与属性字符串的采样率要匹配。
        例如格式选择pcm16k16bit，属性字符串却选择chinese_8k_common, 则会返回'audio_format' is not
match model
        例如wav本身是16k采样率，属性选择chinese_8k_common, 同样会返回'audio_format' is not match
model
"""
# 一句话识别参数，以音频文件的base64编码传入，1min以内音频
path = ''                              # 文件位置, 需要具体到文件，如D:/test.wav
path_audio_format = ''                 # 音频格式，如wav等，详见api文档
path_property = 'chinese_16k_general'   # language_sampleRate_domain, 如chinese_16k_general，详见api文
档
def sasr_example():
    """ 一句话识别示例 """
    # step1 初始化客户端
    config = SisConfig()
    config.set_connect_timeout(10)  # 设置连接超时
    config.set_read_timeout(10)  # 设置读取超时
    # 设置代理，使用代理前一定要确保代理可用。 代理格式可为[host, port] 或 [host, port, username,
password]
    # config.set_proxy(proxy)
    asr_client = AsrCustomizationClient(ak, sk, region, project_id, sis_config=config)
    # step2 构造请求
    data = io_utils.encode_file(path)
    asr_request = AsrCustomShortRequest(path_audio_format, path_property, data)
    # 所有参数均可不设置，使用默认值
    # 设置是否添加标点，yes or no，默认no
    asr_request.set_add_punc('yes')
    # 设置是否将语音中数字转写为阿拉伯数字，yes or no，默认yes
    asr_request.set_digit_norm('yes')
    # 设置是否添加热词表id，没有则不填
    # asr_request.set_vocabulary_id(None)
    # 设置是否需要word_info，yes or no, 默认no
    asr_request.set_need_word_info('no')
    # step3 发送请求，返回结果,返回结果为json格式
    result = asr_client.get_short_response(asr_request)
    # use enterprise_project_Id
    # headers = {'Enterprise-Project-Id': 'your enterprise project id', 'Content-Type': 'application/json'}
    # result = asr_client.get_short_response(asr_request, headers)
    print(json.dumps(result, indent=2, ensure_ascii=False))
if __name__ == '__main__':
    try:
        sasr_example()
    except ClientException as e:
        print(e)
    except ServerException as e:
        print(e)
```

7.2 一句话识别 Websocket 接口

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 该功能为1.70及以上版本SDK新增功能，使用前请检查并更新SDK版本。

- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化SasrWebsocketClient详见**表 SasrWebsocketClient初始化参数**。

表 7-7 SasrWebsocketClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，请参考**AK/SK认证**。
sk	是	String	用户的sk，请参考**AK/SK认证**。
use_aksk	是	Boolean	使用ak、sk要填写true。
region	是	String	区域，如：cn-north-4。具体请参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
callback	是	Object	回调类RasrCallBack，用于监听Websocket连接、响应、断开、错误等。
config	否	Object	详见**表 SisConfig**。
service_endpoint	否	String	终端节点，一般使用默认即可。

表 7-8 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	Integer	连接超时，默认10，单位s。
read_timeout	否	Integer	读取超时，默认10，单位s。
connect_lost_timeout	否	Integer	连接失效超时，默认4，单位s。一般不要修改这个参数。

请求参数

请求类为SasrWebsocketRequest，详见**表 SasrWebsocketRequest**。

表 7-9 SasrWebsocketRequest

参数名称	是否必选	参数类型	描述
audio_format	是	String	音频格式，支持pcm，alaw，ulaw等，如pcm8k16bit，具体规格请参见《API参考》中**开始识别**章节。
model_property	是	String	属性字符串，language_sampleRate_domain， 如chinese_8k_common。
add_punc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digit_norm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
interim_results	否	String	是否显示中间结果，yes 或 no，默认no。
vocabulary_id	否	String	热词表id，若没有则不填。
need_word_info	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

响应参数

Python SDK响应结果为Json格式，详见**表7-10**。调用失败处理方法请参见**错误码**。

表 7-10 响应结果

参数名称	是否必选	参数类型	描述
resp_type	是	String	参数值为RESULT，表示识别结果响应。
trace_id	是	String	服务内部的令牌，可用于在日志中追溯具体流程。
segments	是	Array of objects	多句结果。详见**表7-11**。

表 7-11 Segment

参数名称	是否必选	参数类型	描述
start_time	是	Integer	一句的起始时间戳，单位为ms。

参数名称	是否必选	参数类型	描述
end_time	是	Integer	一句的结束时间戳，单位为ms。
is_final	是	Boolen	true表示是最终结果， false表示为中间临时结果。
result	是	Object	调用成功表示识别结果，详见表7-12。

表 7-12 Result

参数名称	是否必选	参数类型	描述
text	是	String	识别结果。
score	是	Float	识别结果的置信度（0-1之间）。此值仅会在最终结果时被赋值，在中间结果时统一置为“0.0”。
word_info	否	Array of objects	分词信息列表。

表 7-13 Word_info

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往SDK（websocket）章节获取并运行。

```
# -*- coding: utf-8 -*-
from huaweicloud_sis.client.asr_client import SasrWebsocketClient
from huaweicloud_sis.bean.asr_request import SasrWebsocketRequest
from huaweicloud_sis.bean.callback import RasrCallBack
from huaweicloud_sis.bean.sis_config import SisConfig
import json
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK。
ak = os.getenv("HUAWEICLOUD_SIS_AK")            # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")            # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
```

```
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/
sis_03_0008.html
region = 'cn-north-4'  # region，如cn-north-4
# 一句话识别参数
path = ''           # 需要发送音频路径，如D:/test.pcm, 同时sdk也支持byte流发送数据。
audio_format = ''   # 音频支持格式，如pcm16k16bit，详见api文档
property = ''       # 属性字符串，language_sampleRate_domain, 如chinese_16k_common, 采样率要和音频一
致。详见api文档
class MyCallback(RasrCallBack):
    """ 回调类，用户需要在对应方法中实现自己的逻辑，其中on_response必须重写 """
    def on_open(self):
        """ websocket连接成功会回调此函数 """
        print('websocket connect success')
    def on_start(self, message):
        """
            websocket 开始识别回调此函数
        :param message: 传入信息
        :return: -
        """
        print('webscoket start to recognize, %s' % message)
    def on_response(self, message):
        """
            websockert返回响应结果会回调此函数
        :param message: json格式
        :return: -
        """
        print(json.dumps(message, indent=2, ensure_ascii=False))
    def on_end(self, message):
        """
            websocket 结束识别回调此函数
        :param message: 传入信息
        :return: -
        """
        print('websocket is ended, %s' % message)
    def on_close(self):
        """ websocket关闭会回调此函数 """
        print('websocket is closed')
    def on_error(self, error):
        """
            websocket出错回调此函数
        :param error: 错误信息
        :return: -
        """
        print('websocket meets error, the error is %s' % error)
    def on_event(self, event):
        """
            出现事件的回调
        :param event: 事件名称
        :return: -
        """
        print('receive event %s' % event)
def sasr_websocket_example():
    """ 一句话识别 websocket demo """
    # step1 初始化SasrWebsocketClient, 暂不支持使用代理
    my_callback = MyCallback()
    config = SisConfig()
    # 设置连接超时,默认是10
    config.set_connect_timeout(10)
    # 设置读取超时, 默认是10
    config.set_read_timeout(10)
    # 设置connect lost超时，一般在普通并发下，不需要设置此值。默认是10
    config.set_connect_lost_timeout(10)
    # websocket暂时不支持使用代理
    sasr_websocket_client = SasrWebsocketClient(ak=ak, sk=sk, use_aksk=True, region=region,
project_id=project_id,
                                                callback=my_callback, config=config)
    try:
        # step2 构造请求
        request = SasrWebsocketRequest(audio_format, property)
```



```
        # 所有参数均可不设置，使用默认值
        request.set_add_punc('yes')  # 设置是否添加标点， yes or no， 默认no
        request.set_interim_results('no')  # 设置是否返回中间结果，yes or no，默认no
        request.set_digit_norm('no')  # 设置是否将语音中数字转写为阿拉伯数字，yes or no，默认yes
        # request.set_vocabulary_id('')     # 设置热词表id，若不存在则不填写，否则会报错
        request.set_need_word_info('no')  # 设置是否需要word_info，yes or no, 默认no
        # step3 连接服务端
        sasr_websocket_client.sasr_stream_connect(request)
        # use enterprise_project_Id
        # headers = {'Enterprise-Project-Id': 'your enterprise project id'}
        # sasr_websocket_client.sasr_stream_connect(request, headers)
        # step4 发送音频
        sasr_websocket_client.send_start()
        # 连续模式下，可多次发送音频，发送格式为byte数组
        with open(path, 'rb') as f:
            data = f.read()
            sasr_websocket_client.send_audio(data)  # 可选byte_len和sleep_time参数，建议使用默认值
        sasr_websocket_client.send_end()
    except Exception as e:
        print('sasr websocket error', e)
    finally:
        # step5 关闭客户端，使用完毕后一定要关闭，否则服务端20s内没收到数据会报错并主动断开。
        sasr_websocket_client.close()
if __name__ == '__main__':
    sasr_websocket_example()
```

7.3 录音文件识别

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 确保已存在待识别的音频文件并上传OBS或者有公网可访问服务器上（需保证可使用域名访问），示例音频可参考下载SDK压缩包文件。如果音频存放在OBS上，确保服务已授权访问OBS，可参考**配置OBS服务**。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化AsrCustomizationClient详见**表 AsrCustomizationClient初始化参数**。

表 7-14 AsrCustomizationClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如：cn-north-4。具体请参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
service_endpoint	否	String	终端节点，一般使用默认即可。

参数名称	是否必选	参数类型	描述
sis_config	否	Object	详见表7-15。

表 7-15 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	Integer	连接超时，默认10，单位s。
read_timeout	否	Integer	读取超时，默认10，单位s。
proxy	否	List	[host, port] 或 [host, port, username, password]。

请求参数

请求类为AsrCustomLongRequest，详见表7-16。

表 7-16 AsrCustomLongRequest

参数名称	是否必选	参数类型	描述
data_url	是	String	存放录音文件地址： • 推荐使用华为云OBS：授权配置请参见OBS配置。 • 您也可以把录音文件放在自行搭建服务器上，提供下载文件的地址。URL不能使用IP地址，只能使用域名，请尽量避免中文
audio_format	是	String	音频格式，具体信息请参见《API参考》中录音文件识别章节。
model_property	是	String	属性字符串，语言_采样率_模型，如chinese_8k_common。具体信息请参见《API参考》中录音文件识别章节。
add_punc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digit_norm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。

参数名称	是否必选	参数类型	描述
need_analysis_info	否	Boolean	是否选择分析信息。 如果选择false，则声道、话者分离、情绪检测、速度信息均无效。默认false。
diarization	否	Boolean	是否需要话者分离，表示识别结果会包含role项，默认true。
channel	否	String	语音文件声道信息，可以为MONO（缺省），LEFT_AGENT， RIGHT_AGENT。默认MONO。
emotion	否	Boolean	是否需要做情绪检测，默认true。
speed	否	Boolean	是否需要输出语速信息，默认true。
vocabulary_id	否	String	热词表id，不使用则不填写。 创建热词表请参考《API参考》中**创建热词表**章节。
word_info	否	Array of objects	分词信息列表。

响应参数

Python SDK响应结果为Json格式，详见**表7-17**。调用失败处理方法请参见**错误码**。

表 7-17 响应结果

参数名称	是否必选	参数类型	描述
status	否	String	当前识别状态。具体状态如下所示： WAITING 等待识别。 FINISHED 识别已经完成。 ERROR 识别过程中发生错误。
create_time	否	String	任务创建时间, 遵循 RFC 3339格式。 格式示例：2018-12-04T13:10:29.310Z。
start_time	否	String	开始识别时间, 遵循 RFC 3339格式。 当status为FINISHED或ERROR时存在。 格式示例：2018-12-04T13:10:29.310Z。
finish_time	否	String	识别完成时间, 遵循 RFC 3339格式。 当status为FINISHED或ERROR时存在。 格式示例：2018-12-04T13:10:29.310Z。
audio_duration	否	Integer	提交音频时长，单位ms。

参数名称	是否必选	参数类型	描述
segments	否	Array of objects	识别结果, 多句结果的数组。 数据结构参见表7-18。

表 7-18 Segment

参数名	是否必选	参数类型	说明
start_time	是	Integer	一句的起始时间戳，单位ms。
end_time	是	Integer	一句的结束时间戳，单位ms。
result	是	Object	调用成功表示识别结果，调用失败时无此字段。详见表7-19。

表 7-19 Result

参数名	是否必选	参数类型	说明
text	是	String	识别结果文本。
analysis_info	否	Object	每一句的质检分析结果对象。 仅在识别配置中的need_analysis_info不为null时存在该返回结果。详见表7-20。
word_info	否	Array of Object	分词输出列表。

表 7-20 Analysis_info

参数名	是否必选	参数类型	说明
role	否	String	角色类型，目前仅支持 AGENT(座席)，USER(用户)。
emotion	否	String	情绪类型，目前仅支持NORMAL(正常)，ANGRY(愤怒)。 在识别配置中emotion为true时存在。
speed	否	Float	语速信息，单位是"每秒字数"。 在识别配置中speed为true时存在。

表 7-21 Word_info 数据结构

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
# -*- coding: utf-8 -*-
from huaweicloud_sis.client.asr_client import AsrCustomizationClient
from huaweicloud_sis.bean.asr_request import AsrCustomLongRequest
from huaweicloud_sis.exception.exceptions import ClientException
from huaweicloud_sis.exception.exceptions import ServerException
from huaweicloud_sis.bean.sis_config import SisConfig
import json
import time
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK
ak = os.getenv("HUAWEICLOUD_SIS_AK")              # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")              # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/sis_03_0008.html
region = ''         # region，如cn-north-4
"""
    todo 请正确填写音频格式和模型属性字符串
    1. 音频格式一定要相匹配.
        例如wav音频，格式是auto。具体参考api文档。
        例如音频是pcm格式，并且采样率为8k，则格式填写pcm8k16bit。
        如果返回audio_format is invalid 说明该文件格式不支持。具体支持哪些音频格式，需要参考一些api文档。

    2. 音频采样率要与属性字符串的采样率要匹配。
        例如格式选择pcm16k16bit，属性字符串却选择chinese_8k_common, 则会返回'audio_format' is not match model
        例如wav本身是16k采样率，属性选择chinese_8k_common, 同样会返回'audio_format' is not match model
"""
# 录音文件识别参数，音频文件以obs连接方式传入（即先需要将音频传送到华为云的obs）
obs_url = ''                # 音频obs连接
obs_audio_format = ''       # 音频格式，如auto等，详见api文档
obs_property = ''           # language_sampleRate_domain, 如chinese_8k_common，详见api文档
def lasr_example():
    """ 录音文件识别示例 """
    # step1 初始化客户端
    config = SisConfig()
    config.set_connect_timeout(10)       # 设置连接超时
    config.set_read_timeout(10)         # 设置读取超时
    # 设置代理，使用代理前一定要确保代理可用。 代理格式可为[host, port] 或 [host, port, username, password]
    # config.set_proxy(proxy)
    asr_client = AsrCustomizationClient(ak, sk, region, project_id,  sis_config=config)
```

```
    # step2 构造请求
    asrc_request = AsrCustomLongRequest(obs_audio_format, obs_property, obs_url)
    # 所有参数均可不设置，使用默认值
    # 设置是否添加标点，yes or no，默认no
    asrc_request.set_add_punc('yes')
    # 设置是否将语音中数字转写为阿拉伯数字，yes or no，默认yes
    asrc_request.set_digit_norm('yes')
    # 设置 是否需要分析信息，True or False, 默认False。 只有need_analysis_info生效，diarization、channel、
emotion、speed才会生效
    # 目前仅支持8k模型，详见api文档
    asrc_request.set_need_analysis_info(True)
    # 设置是否需要话者分离，默认True，需要need_analysis_info设置为True才生效。
    asrc_request.set_diarization(True)
    # 设置声道信息, 一般都是单声道，默认为MONO，需要need_analysis_info设置为True才生效
    asrc_request.set_channel('MONO')
    # 设置是否返回感情信息, 默认True，需要need_analysis_info设置为True才生效。
    asrc_request.set_emotion(True)
    # 设置是否需要返回语速信息，默认True，需要need_analysis_info设置为True才生效。
    asrc_request.set_speed(True)
    # 设置回调地址，设置后音频转写结果将直接发送至回调地址。请务必保证地址可连通。
    # asrc_request.set_callback_url('')
    # 设置是否添加热词表id，没有则不填
    # asrc_request.set_vocabulary_id(None)
    # 设置是否需要word_info，yes or no, 默认no
    asrc_request.set_need_word_info('no')
    # step3 发送请求，获取job_id
    job_id = asr_client.submit_job(asrc_request)
    # use enterprise_project_Id
    # headers = {'Enterprise-Project-Id': 'your enterprise project id', 'Content-Type': 'application/json'}
    # job_id = asr_client.submit_job(asrc_request, headers)
    # step4 根据job_id轮询，获取结果。
    status = 'WAITING'
    count = 0  # 每2s查询一次，尝试2000次，即4000s。如果音频很长，可适当考虑加长一些。
    while status != 'FINISHED' and count < 2000:
        print(count, ' query')
        result = asr_client.get_long_response(job_id)
        status = result['status']
        if status == 'ERROR':
            print('录音文件识别执行失败, %s' % json.dump(result))
            break
        time.sleep(2)
        count += 1
    if status != 'FINISHED':
        print('录音文件识别未在 %d 内获取结果，job_id 为%s' % (count, job_id))
    # result为json格式
    print(json.dumps(result, indent=2, ensure_ascii=False))
if __name__ == '__main__':
    try:
        lasr_example()
    except ClientException as e:
        print(e)
    except ServerException as e:
        print(e)
```

7.4 实时语音识别

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RasrClient详见**表 RasrClient初始化参数**。

表 7-22 RasrClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，请参考**AK/SK认证**。
sk	是	String	用户的sk，请参考**AK/SK认证**。
use_aksk	是	Boolean	使用ak、sk要填写true。
region	是	String	区域，如：cn-north-4。具体请参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
callback	是	Object	回调类RasrCallBack，用于监听Websocket连接、响应、断开、错误等。参考**代码示例**。
config	否	Object	详见**表7-23**。
service_endpoint	否	String	终端节点，一般使用默认即可。

表 7-23 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	Integer	连接超时，默认10，单位s。
read_timeout	否	Integer	读取超时，默认10，单位s。
connect_lost_timeout	否	Integer	连接失效超时，默认4，单位s。一般不要修改这个参数。

请求参数

请求类为RasrRequest，详见**表7-24**。

表 7-24 RasrRequest

参数名称	是否必选	参数类型	描述
audio_format	是	String	音频格式，支持pcm，alaw，ulaw等，如pcm8k16bit，参见《API参考》中开始识别章节。
model_property	是	String	属性字符串，language_sampleRate_domain， 如chinese_16k_general，参见《API参考》中开始识别章节。
add_punc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digit_norm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vad_head	否	Integer	头部最大静音时间，[0, 60000]，默认10000ms。
vad_tail	否	Integer	尾部最大静音时间，[0, 3000]，默认500ms。
max_seconds	否	Integer	音频最长持续时间， [1, 60]，默认30s。
interim_results	否	String	是否显示中间结果，yes 或 no，默认no。例如分3次发送音频，选择no结果一次性返回，选择yes分三次返回。
vocabulary_id	否	String	热词表id，若没有则不填。
need_word_info	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。
need_smooth	否	String	是否需要对识别结果进行顺滑（过滤语气词或者吞吐词），取值为“yes”和“no”。仅支持中文ASR的顺滑。

响应参数

Python SDK响应结果为Json格式，详见表7-25。调用失败处理方法请参见错误码。

表 7-25 响应结果

参数名称	是否必选	参数类型	描述
resp_type	是	String	参数值为RESULT，表示识别结果响应。
trace_id	是	String	服务内部的令牌，可用于在日志中追溯具体流程。
segments	是	Array of objects	多句结果。详见表7-26。

表 7-26 Segment

参数名称	是否必选	参数类型	描述
start_time	是	Integer	一句的起始时间戳，单位为ms。
end_time	是	Integer	一句的结束时间戳，单位为ms。
is_final	是	Boolen	true表示是最终结果， false表示为中间临时结果。
result	是	Object	调用成功表示识别结果，详见表7-27。

表 7-27 Result

参数名称	是否必选	参数类型	描述
text	是	String	识别结果。
score	是	Float	识别结果的置信度（0-1之间）。此值仅会在最终结果时被赋值，在中间结果时统一置为“0.0”。
word_info	否	Array of objects	分词信息列表。

表 7-28 Word_info

参数名	是否必选	参数类型	说明
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往SDK（websocket）章节获取并运行。

```
# -*- coding: utf-8 -*-
from huaweicloud_sis.client.rasr_client import RasrClient
from huaweicloud_sis.bean.rasr_request import RasrRequest
from huaweicloud_sis.bean.callback import RasrCallBack
from huaweicloud_sis.bean.sis_config import SisConfig
import json
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK
```

```
ak = os.getenv("HUAWEICLOUD_SIS_AK")              # 从环境变量获取ak 参考https://
support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")              # 从环境变量获取sk 参考https://support.huaweicloud.com/
sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/
sis_03_0008.html
region = 'cn-north-4'         # region，如cn-north-4
"""
    todo 请正确填写音频格式和模型属性字符串
    1. 音频格式一定要相匹配.
        例如音频是pcm格式，并且采样率为8k，则格式填写pcm8k16bit。
        如果返回audio_format is invalid 说明该文件格式不支持。具体支持哪些音频格式，需要参考一些api文
档。
    2. 音频采样率要与属性字符串的采样率要匹配。
        例如格式选择pcm16k16bit，属性字符串却选择chinese_8k_common, 则会返回'audio_format' is not
match model
"""
# 实时语音识别参数
path = ''  # 需要发送音频路径，如D:/test.pcm, 同时sdk也支持byte流发送数据。
audio_format = 'pcm16k16bit'  # 音频支持格式，如pcm16k16bit，详见api文档
property = 'chinese_16k_general'  # 属性字符串，language_sampleRate_domain, 如chinese_16k_general, 采
样率要和音频一致。详见api文档
class MyCallback(RasrCallBack):
    """ 回调类，用户需要在对应方法中实现自己的逻辑，其中on_response必须重写 """
    def on_open(self):
        """ websocket连接成功会回调此函数 """
        print('websocket connect success')
    def on_start(self, message):
        """
            websocket 开始识别回调此函数
        :param message: 传入信息
        :return: -
        """
        print('webscoket start to recognize, %s' % message)
    def on_response(self, message):
        """
            websockert返回响应结果会回调此函数
        :param message: json格式
        :return: -
        """
        print(json.dumps(message, indent=2, ensure_ascii=False))
    def on_end(self, message):
        """
            websocket 结束识别回调此函数
        :param message: 传入信息
        :return: -
        """
        print('websocket is ended, %s' % message)
    def on_close(self):
        """ websocket关闭会回调此函数 """
        print('websocket is closed')
    def on_error(self, error):
        """
            websocket出错回调此函数
        :param error: 错误信息
        :return: -
        """
        print('websocket meets error, the error is %s' % error)
    def on_event(self, event):
        """
            出现事件的回调
        :param event: 事件名称
        :return: -
        """
        print('receive event %s' % event)
def rasr_example():
    """ 实时语音识别demo """
```

```
    # step1 初始化RasrClient, 暂不支持使用代理
    my_callback = MyCallback()
    config = SisConfig()
    # 设置连接超时,默认是10
    config.set_connect_timeout(10)
    # 设置读取超时, 默认是10
    config.set_read_timeout(10)
    # 设置connect lost超时，一般在普通并发下，不需要设置此值。默认是10
    config.set_connect_lost_timeout(10)
    # websocket暂时不支持使用代理
    rasr_client = RasrClient(ak=ak, sk=sk, use_aksk=True, region=region, project_id=project_id,
callback=my_callback,
                             config=config)
    try:
        # step2 构造请求
        request = RasrRequest(audio_format, property)
        # 所有参数均可不设置，使用默认值
        request.set_add_punc('yes')  # 设置是否添加标点， yes or no， 默认no
        request.set_vad_head(10000)  # 设置有效头部， [0, 60000], 默认10000
        request.set_vad_tail(500)  # 设置有效尾部，[0, 3000]， 默认500
        request.set_max_seconds(30)  # 设置一句话最大长度，[1, 60], 默认30
        request.set_interim_results('no')  # 设置是否返回中间结果，yes or no，默认no
        request.set_digit_norm('no')  # 设置是否将语音中数字转写为阿拉伯数字，yes or no，默认yes
        # request.set_vocabulary_id('')     # 设置热词表id，若不存在则不填写，否则会报错
        request.set_need_word_info('no')  # 设置是否需要word_info，yes or no, 默认no
        request.set_need_smooth('no')     # 设置是否需要打开文本顺滑，默认no
        # step3 选择连接模式
        # rasr_client.short_stream_connect(request)       # 流式一句话模式
        # rasr_client.sentence_stream_connect(request)    # 实时语音识别单句模式
        rasr_client.continue_stream_connect(request)  # 实时语音识别连续模式
        # use enterprise_project_Id
        # headers = {'Enterprise-Project-Id': 'your enterprise project id'}
        # rasr_client.continue_stream_connect(request, headers)
        # step4 发送音频
        rasr_client.send_start()
        # 连续模式下，可多次发送音频，发送格式为byte数组
        with open(path, 'rb') as f:
            data = f.read()
            rasr_client.send_audio(data)  # 可选byte_len和sleep_time参数，建议使用默认值
        rasr_client.send_end()
    except Exception as e:
        print('rasr error', e)
    finally:
        # step5 关闭客户端，使用完毕后一定要关闭，否则服务端20s内没收到数据会报错并主动断开。
        rasr_client.close()
if __name__ == '__main__':
    rasr_example()
```

7.5 语音合成

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化TtsCustomizationClient详见**表 TtsCustomizationClient初始化参数**。

表 7-29 TtsCustomizationClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如：cn-north-4。具体请参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
service_endpoint	否	String	终端节点，一般使用默认即可。
sis_config	否	Object	详见**表7-30**。

表 7-30 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	String	连接超时，默认10，单位s。
read_timeout	否	String	读取超时，默认10，单位s。
proxy	否	List	[host, port] 或 [host, port, username, password]。

请求参数

请求类为TtsCustomRequest，详见**表7-31**。

表 7-31 TtsCustomRequest

参数名称	是否必选	参数类型	描述
text	是	String	待合成的文本。
audio_format	否	String	待合成的音频格式，可选mp3，wav等，默认wav。具体信息请参见《API参考》**语音合成**章节。
pitch	否	Integer	音高，[-500,500]，默认是0。
speed	否	Integer	语速，[-500,500]，默认是0。
volume	否	Integer	音量，[0,100]，默认是50。

参数名称	是否必选	参数类型	描述
sample_rate	否	String	采样率，支持“8000”、“16000”，默认“8000”。
model_property	否	String	特征字符串，{language}_{speaker}_{domain}，默认chinese_xiaoyan_common。具体信息请参见《API参考》中**语音合成**章节。
saved	否	Boolean	是否选择合成的音频数据保存到本地，默认不保存。
saved_path	否	String	选择保存到本地的路径，需要具体到音频文件，如D:/test.wav。

响应参数

Python SDK响应结果为Json格式，详见**表7-32**。调用失败处理方法请参见**错误码**。

表 7-32 响应结果

参数名称	是否必选	参数类型	描述
result	是	Object	调用成功时为合成语音内容，请参考**表7-33**。
trace_id	是	String	用于后台日志问题追溯。
is_saved	否	Boolean	是否保存为本地音频。
saved_path	否	String	保存音频的本地路径，只有在请求时saved参数设置为true才生效。

表 7-33 Result

参数名称	是否必选	参数类型	说明
data	是	String	合成后生成的语音数据，以Base64编码格式返回。

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
# -*- coding: utf-8 -*-

from huaweicloud_sis.client.tts_client import TtsCustomizationClient
from huaweicloud_sis.bean.tts_request import TtsCustomRequest
from huaweicloud_sis.bean.sis_config import SisConfig
from huaweicloud_sis.exception.exceptions import ClientException
from huaweicloud_sis.exception.exceptions import ServerException
```

```
import json
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK/HUAWEICLOUD_SIS_PROJECT_ID。
ak = os.getenv("HUAWEICLOUD_SIS_AK")              # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")              # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/sis_03_0008.html

def ttsc_example():
    """ 语音合成demo """


    region = ''         # region，如cn-north-4
    text = ''           # 待合成文本，不超过500字
    path = ''           # 保存路径，如D:/test.wav。 可在设置中选择不保存本地

    # step1 初始化客户端
    config = SisConfig()
    config.set_connect_timeout(10)       # 设置连接超时，单位s
    config.set_read_timeout(10)          # 设置读取超时，单位s
    # 设置代理，使用代理前一定要确保代理可用。 代理格式可为[host, port] 或 [host, port, username, password]
    # config.set_proxy(proxy)
    ttsc_client = TtsCustomizationClient(ak, sk, region, project_id, sis_config=config)

    # step2 构造请求
    ttsc_request = TtsCustomRequest(text)
    # 设置请求，所有参数均可不设置，使用默认参数
    # 设置属性字符串， language_speaker_domain, 默认chinese_xiaoyan_common, 参考api文档
    ttsc_request.set_property('chinese_xiaoyan_common')
    # 设置音频格式，默认wav，可选mp3和pcm
    ttsc_request.set_audio_format('wav')
    # 设置采样率，8000 or 16000, 默认8000
    ttsc_request.set_sample_rate('8000')
    # 设置音量，[0, 100]，默认50
    ttsc_request.set_volume(50)
    # 设置音高, [-500, 500], 默认0
    ttsc_request.set_pitch(0)
    # 设置音速, [-500, 500], 默认0
    ttsc_request.set_speed(0)
    # 设置是否保存，默认False
    ttsc_request.set_saved(True)
    # 设置保存路径，只有设置保存，此参数才生效
    ttsc_request.set_saved_path(path)

    # step3 发送请求，返回结果。如果设置保存，可在指定路径里查看保存的音频。
    result = ttsc_client.get_ttsc_response(ttsc_request)
    # use enterprise_project_Id
    # headers = {'Enterprise-Project-Id': 'your enterprise project id', 'Content-Type': 'application/json'}
    # result = ttsc_client.get_ttsc_response(ttsc_request, headers)
    print(json.dumps(result, indent=2, ensure_ascii=False))


if __name__ == '__main__':
    try:
        ttsc_example()
    except ClientException as e:
        print(e)
    except ServerException as e:
        print(e)
```

7.6 热词管理

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化HotWordClient，详见**表 HotWordClient初始化参数**。

表 7-34 HotWordClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如：cn-north-4。具体请参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
service_endpoint	否	String	终端节点，一般使用默认即可。
sis_config	否	Object	详见**表7-35**。

表 7-35 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	String	连接超时，默认10，单位s。
read_timeout	否	String	读取超时，默认10，单位s。
proxy	否	List	[host, port] 或 [host, port, username, password]。

请求参数

请求类为HotWordRequest，详见**表7-36**。

表 7-36 HotWordRequest

参数名称	是否必选	参数类型	描述
name	是	String	热词表名，创建时不可重复。内容限制为字母，数字，下中划线和井号，长度不超过32字节。
language	是	String	热词表语言类型，目前支持汉语普通话“chinese_mandarin”。
contents	是	Array of String	热词库，单词库支持热词数上限10000。中文单个热词长度上限32字节。
description	否	String	热词表描述，长度不超过255字节。

响应参数

创建热词响应参数为Json格式，详见**表7-37**。调用失败处理方法请参见**错误码**。

表 7-37 创建热词响应

参数名称	是否必选	参数类型	描述
vocabulary_id	是	String	调用成功则返回热词表ID。

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
# -*- coding: utf-8 -*-

from huaweicloud_sis.client.hot_word_client import HotWordClient
from huaweicloud_sis.bean.hot_word_request import HotWordRequest
from huaweicloud_sis.exception.exceptions import ClientException
from huaweicloud_sis.exception.exceptions import ServerException
from huaweicloud_sis.bean.sis_config import SisConfig
import json
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK
ak = os.getenv("HUAWEICLOUD_SIS_AK")            # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")            # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/sis_03_0008.html
region = ''         # region，如cn-north-4
# 热词参数
```

```
name = ''            # 创建热词时，需要保证name在此之前没有被创建使用过。如 test1
word_list = list()  # 用于存放热词表。每个热词表最多可以存放10000个热词。如["计算机", "网络"]
vocabulary_id = ''  # 用于更新指定热词表id信息，查询指定热词表id信息，删除指定热词表id信息。使用前要保证热词表id存在，否则就不要使用。

def hot_word_example():
    """
        1. 热词使用包含创建、更新、查询、删除等，一个用户可以创建多个热词表，一个热词表可以包含多个热词。一个vocabulary_id对应一个热词表。
        2. 目前支持一个用户最多创建10个热词表，一个热词表最多包含10000个热词。
        3. 热词可在一句话识别、录音文件识别、实时语音识别使用。例如将地名和人名作为热词，则语音可以准确识别出人名和地名。
    :return: 无
    """
    # 初始化客户端
    config = SisConfig()
    config.set_connect_timeout(10)       # 设置连接超时
    config.set_read_timeout(10)          # 设置读取超时
    # 设置代理，使用代理前一定要确保代理可用。 代理格式可为[host, port] 或 [host, port, username, password]
    # config.set_proxy(proxy)
    hot_word_client = HotWordClient(ak, sk, region, project_id, sis_config=config)

    # option 1 创建热词表
    word_list.append('测试')
    create_request = HotWordRequest(name, word_list)
    # 可选，热词语言，目前仅支持中文 chinese_mandarin。
    create_request.set_language('chinese_mandarin')
    # 可选，热词表描述信息
    create_request.set_description('test')
    create_result = hot_word_client.create(create_request)
    # 返回结果为json格式
    print('成功创建热词表')
    print(json.dumps(create_result, indent=2, ensure_ascii=False))

    # option 2 根据热词表id 更新热词表。新的热词表会替换旧的热词表。使用前需确保热词表id已存在。
    word_list.append('计算机')
    update_request = HotWordRequest('test2', word_list)
    update_result = hot_word_client.update(update_request, vocabulary_id)
    # 返回结果为json格式
    print('成功更新热词表', vocabulary_id)
    print(json.dumps(update_result, indent=2, ensure_ascii=False))

    # option 3 查看热词表列表
    query_list_result = hot_word_client.query_list()
    print(json.dumps(query_list_result, indent=2, ensure_ascii=False))

    # option 4 根据热词表id查询具体热词表信息，使用前需确保热词表id已存在。
    query_result = hot_word_client.query_by_vocabulary_id(vocabulary_id)
    print(json.dumps(query_result, indent=2, ensure_ascii=False))

    # option 5 根据热词表id删除热词表，使用前需确保热词表id已存在。
    delete_result = hot_word_client.delete(vocabulary_id)
    if delete_result is None:
        print('成功删除热词表', vocabulary_id)
    else:
        print(json.dumps(delete_result, indent=2, ensure_ascii=False))


if __name__ == '__main__':
    try:
        hot_word_example()
    except ClientException as e:
        print(e)
    except ServerException as e:
        print(e)
```

7.7 实时语音合成

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RttsClient详见**表 RttsClient初始化参数**。

表 7-38 RttsClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如：cn-north-4。具体请参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
service_endpoint	否	String	终端节点，一般使用默认即可。
sis_config	否	Object	详见**表7-39**。

表 7-39 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	Integer	连接超时，默认10，单位s。
read_timeout	否	Integer	读取超时，默认10，单位s。
websocket_wait_time	否	Integer	websocket最大等待时间，默认20，单位s。
proxy	否	List	[host, port] 或 [host, port, username, password]。

请求参数

请求类为RttsRequest，详见表7-40。

表 7-40 RttsRequest

参数名称	是否必选	参数类型	描述
text	是	String	待合成的文本。1-500字
audio_format	否	String	语音格式头：pcm、alaw、ulaw。 默认：pcm
pitch	否	Integer	音高，[-500,500]，默认是0。
speed	否	Integer	语速，[-500,500]，默认是0。
volume	否	Integer	音量，[0,100]，默认是50。
sample_rate	否	String	采样率，支持“8000”、“16000”，默认“8000”。
property	否	String	语音合成特征字符串，组成形式为{language}_{speaker}_{domain}，即“语种_人员标识_领域”。 ● language取值范围： – chinese ● speaker取值范围： – xiaoqi 正式女生 – xiaoyu正式男生 – xiaoyan情感女生 – xiaowang童声 ● speaker（精品发音人）取值范围： – huaxiaomei温柔女声发音人，仅支持pcm – huaxiaofei朝气男声发音人，仅支持pcm ● domain取值范围： – common，通用领域 默认：chinese_xiaoyan_common 实时语音合成和语音合成属于同一种资源，按次计费。实时语音合成普通发音人，每100字计一次。精品发音人每50字计一次。

响应参数

Python SDK响应结果为byte数组，保存合成音频数据。详见代码示例。调用失败处理方法请参见错误码。

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
# -*- coding: utf-8 -*-
from huaweicloud_sis.client.rtts_client import RttsClient
from huaweicloud_sis.bean.rtts_request import RttsRequest
from huaweicloud_sis.bean.callback import RttsCallBack
from huaweicloud_sis.bean.sis_config import SisConfig
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK
ak = os.getenv("HUAWEICLOUD_SIS_AK")              # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")              # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/sis_03_0008.html
region = ''  # region，如cn-north-4
text = ''  # 待合成的文本
path = ''  # 待合成的音频保存路径，如test.pcm
class MyCallback(RttsCallBack):
    """ 回调类，用户需要在对应方法中实现自己的逻辑，其中on_response必须重写 """
    def __init__(self, save_path):
        self._f = open(save_path, 'wb')
    def on_open(self):
        """ websocket连接成功会回调此函数 """
        print('websocket connect success')
    def on_start(self, message):
        """
            websocket 开始识别回调此函数
        :param message: 传入信息
        :return: -
        """
        print('webscoket start to recognize, %s' % message)
    def on_response(self, data):
        """
            回调返回的音频合成数据，byte数组格式
        :param data byte数组，合成的音频数据
        :return: -
        """
        print('receive data %d' % len(data))
        self._f.write(data)
    def on_end(self, message):
        """
            websocket 结束识别回调此函数
        :param message: 传入信息
        :return: -
        """
        print('websocket is ended, %s' % message)
        self._f.close()
    def on_close(self):
        """ websocket关闭会回调此函数 """
        print('websocket is closed')
        self._f.close()
    def on_error(self, error):
        """
            websocket出错回调此函数
        :param error: 错误信息
        :return: -
        """
        print('websocket meets error, the error is %s' % error)
        self._f.close()
def rtts_example():
```

```
    """
        实时语音合成demo
        1. RttsClient 只能发送一次文本，如果需要多次发送文本，需要新建多个RttsClient 和 callback
        2. 识别完成后服务端会返回end响应。
        3. 当识别出现问题时，会触发on_error回调，同时会关闭websocket。
        4. 实时语音合成会多次返回结果，demo的处理方式是将多次返回结果集合在一个音频文件里。
    """
    # step1 初始化RttsClient, 暂不支持使用代理
    my_callback = MyCallback(path)
    config = SisConfig()
    # 设置连接超时,默认是10
    config.set_connect_timeout(10)
    # 设置读取超时, 默认是10
    config.set_read_timeout(10)
    # 设置websocket等待时间
    config.set_websocket_wait_time(20)
    # websocket暂时不支持使用代理
    rtts_client = RttsClient(ak=ak, sk=sk, use_aksk=True, region=region, project_id=project_id,
callback=my_callback,
                             config=config)
    # step2 构造请求
    rtts_request = RttsRequest(text)
    # 设置属性字符串， language_speaker_domain, 默认chinese_xiaoyan_common, 参考api文档
    rtts_request.set_property('chinese_xiaoyan_common')
    # 设置音频格式为pcm
    rtts_request.set_audio_format('pcm')
    # 设置采样率，8000 or 16000, 默认8000
    rtts_request.set_sample_rate('8000')
    # 设置音量，[0, 100]，默认50
    rtts_request.set_volume(50)
    # 设置音高, [-500, 500], 默认0
    rtts_request.set_pitch(0)
    # 设置音速, [-500, 500], 默认0
    rtts_request.set_speed(0)
    # step3 合成
    rtts_client.synthesis(rtts_request)
    # use enterprise_project_Id
    # headers = {'Enterprise-Project-Id': 'your enterprise project id'}
    # rtts_client.synthesis(rtts_request, headers)
if __name__ == '__main__':
    rtts_example()
```

7.8 录音文件极速版

前提条件

- 确保已按照**配置Python环境**配置完毕，Python SDK仅支持Python3。
- 确保已存在待识别的音频文件。如果需要请在下载的SDK压缩包中获取示例音频。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化FlashLasrClient详见**表 FlashLasrClient初始化参数**。

表 7-41 FlashLasrClient 初始化参数

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。

参数名称	是否必选	参数类型	描述
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
project_id	是	String	项目ID，同region一一对应，参考**获取项目ID**。
service_endpoint	否	String	终端节点，一般使用默认即可。
sis_config	否	Object	详见**表7-42**。

表 7-42 SisConfig

参数名称	是否必选	参数类型	描述
connect_timeout	否	Integer	连接超时，默认10，单位s。
read_timeout	否	Integer	读取超时，默认10，单位s。
proxy	否	List	[host, port] 或 [host, port, username, password]。

请求参数

请求类为FlashLasrRequest，详见**表7-43**。

表 7-43 FlashLasrRequest

参数	是否必选	参数类型	描述
audio_format	是	String	支持语音的格式，请参考**表audio_format取值范围**。
property	是	String	所使用的模型特征串，通常是 “语种_采样率_领域”的形式，采样率需要与音频采样率保持一致，取值范围请参考**表property取值范围**。
add_punc	否	String	表示是否在识别结果中添加标点，取值为“yes”和“no”，默认为“no”。
digit_norm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为“yes” 和 “no”，默认为“yes”。

参数	是否必选	参数类型	描述
vocabulary_id	否	String	热词表id，不使用则不填写。 创建热词表信息请参考**创建热词表**。
need_word_info	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。
first_channel_only	否	String	表示是否在识别中只识别首个声道的音频数据，取值为“yes”和“no”，默认为“no”。
obs_bucket_name	否	String	表示在OBS对象桶名，使用前请先授权，操作方法请参见**配置OBS访问权限**。 obs_bucket_name长度大于等于3个字符，小于64个字符，不需要进行urlencode编码，如果包含中文，直接输入中文即可。 示例 obs url为https://test.obs.cn-north-4.myhuaweicloud.com/data/0601/test.wav 则obs_bucket_name=test，obs_bucket_key=data/0601/test.wav
obs_object_key	否	String	表示OBS对象桶中的对象的键值，长度小于1024个字符，不需要进行urlencode编码，如果包含中文，直接输入中文即可。 示例 obs url为https://test.obs.cn-north-4.myhuaweicloud.com/data/0601/test.wav 则obs_bucket_name=test，obs_bucket_key=data/0601/test.wav

表 7-44 audio_format 取值范围

audio_format取值	描述
wav	wav格式音频
mp3	mp3格式音频
m4a	m4a格式音频
aac	aac格式音频
opus	ops格式音频。

表 7-45 property 取值范围

property取值	描述
chinese_8k_common	支持采样率为8k的中文普通话语音识别。
chinese_16k_conversation	支持采样率为16k的会议场景的中文普通话语音识别。

响应参数

响应类为FlashLasrResponse, 详见**表7-46**。调用失败处理方法请参见**错误码**。

表 7-46 FlashLasrResponse

参数	是否必选	参数类型	描述
trace_id	是	String	服务内部的令牌，可用于在日志中追溯具体流程，调用失败无此字段。 在某些错误情况下可能没有此令牌字符串。
audio_duration	是	Integer	音频时长，单位毫秒
flash_result	是	Array of **FlashResult** objects	调用成功表示识别结果，调用失败时无此字段。

表 7-47 FlashResult

参数	是否必选	参数类型	描述
channel_id	否	Integer	声道Id
sentences	否	Array of **Sentences** objects	分句信息列表

表 7-48 Sentences

参数	是否必选	参数类型	描述
start_time	否	Integer	一句话开始时间，单位毫秒
result	否	**Result** object	分句结果信息
end_time	否	Integer	一句话结束时间，单位毫秒

表 7-49 Result

参数	是否必选	参数类型	描述
text	是	String	调用成功表示识别出的内容。
score	是	Double	调用成功表示识别出的置信度（0-1之间）。
word_info	否	Array of **WordInfo** objects	分词信息列表

表 7-50 WordInfo

参数	是否必选	参数类型	描述
start_time	否	Integer	起始时间
end_time	否	Integer	结束时间
word	否	String	分词

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
# -*- coding: utf-8 -*-
from huaweicloud_sis.client.flash_lasr_client import FlashLasrClient
from huaweicloud_sis.bean.flash_lasr_request import FlashLasrRequest
from huaweicloud_sis.exception.exceptions import ClientException
from huaweicloud_sis.exception.exceptions import ServerException
from huaweicloud_sis.bean.sis_config import SisConfig
import json
import os
# 鉴权参数
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SIS_AK/HUAWEICLOUD_SIS_SK
ak = os.getenv("HUAWEICLOUD_SIS_AK")            # 从环境变量获取ak 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert ak is not None, "Please add ak in your develop environment"
sk = os.getenv("HUAWEICLOUD_SIS_SK")            # 从环境变量获取sk 参考https://support.huaweicloud.com/sdkreference-sis/sis_05_0003.html
assert sk is not None, "Please add sk in your develop environment"
project_id = ""     # project id 同region一一对应，参考https://support.huaweicloud.com/api-sis/sis_03_0008.html
region = ''         # region，如cn-north-4
obs_bucket_name = ''    # obs桶名
obs_object_key = ''     # obs对象的key
audio_format = ''       # 文件格式，如wav等， 支持格式详见api文档
property = ''           # 属性字符串，language_sampleRate_domain, 如chinese_8k_common, 详见api文档
def flash_lasr_example():
    """ 录音文件极速版示例 """
    # step1 初始化客户端
    config = SisConfig()
    config.set_connect_timeout(10)  # 设置连接超时
    config.set_read_timeout(10)  # 设置读取超时
    # 设置代理，使用代理前一定要确保代理可用。 代理格式可为[host, port] 或 [host, port, username,
```

```
password]
    # config.set_proxy(proxy)
    client = FlashLasrClient(ak, sk, region, project_id, sis_config=config)
    # step2 构造请求
    asr_request = FlashLasrRequest()
    # 以下参数必选
    # 设置存放音频的桶名，必选
    asr_request.set_obs_bucket_name(obs_bucket_name)
    # 设置桶内音频对象名，必选
    asr_request.set_obs_object_key(obs_object_key)
    # 设置格式，必选
    asr_request.set_audio_format(audio_format)
    # 设置属性，必选
    asr_request.set_property(property)
    # 以下参数可选
    # 设置是否添加标点，yes or no，默认no
    asr_request.set_add_punc('yes')
    # 设置是否将语音中数字转写为阿拉伯数字，yes or no，默认yes
    asr_request.set_digit_norm('yes')
    # 设置是否添加热词表id，没有则不填
    # asr_request.set_vocabulary_id(None)
    # 设置是否需要word_info，yes or no, 默认no
    asr_request.set_need_word_info('no')
    # 设置是否只识别首个声道的音频数据，默认no
    asr_request.set_first_channel_only('no')
    # step3 发送请求，返回结果,返回结果为json格式
    result = client.get_flash_lasr_result(asr_request)
    # use enterprise_project_Id
    # headers = {'Enterprise-Project-Id': 'your enterprise project id', 'Content-Type': 'application/json'}
    # result = client.get_flash_lasr_result(asr_request, headers)
    print(json.dumps(result, indent=2, ensure_ascii=False))
if __name__ == '__main__':
    try:
        flash_lasr_example()
    except ClientException as e:
        print(e)
    except ServerException as e:
        print(e)
```

8 iOS SDK

8.1 一句话识别

前提条件

- 确保已经按照配置好iOS开发环境。
- 已经保存好1分钟内音频文件，建议使用16k16bit进行录音并保存为wav格式。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化SASRClient，参数为AuthInfo，详见**表8-1**。

表 8-1 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。

请求参数

请求类为SASRConfig和语音数据data，详见**表8-2**。

表 8-2 SASRConfig

参数名称	是否必选	参数类型	描述
config	是	**Config** object	配置信息。
data	是	String	本地音频文件经过Base64编码后的字符串，音频文件时长小于60s。

表 8-3 Config

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，具体信息请参见《API参考》中**一句话识别**章节。
property	是	String	属性字符串，语言_采样率_模型，如chinese_16k_general。具体信息请参见《API参考》中**一句话识别**章节。
addPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vocabularyId	否	String	热词表id，不使用则不填写。 创建热词表请参考《API参考》中**创建热词表**章节。

响应参数

响应类为SASRResponse，详见**表8-4**。调用失败处理方法请参见**错误码**。

表 8-4 SASRResponse

参数名	是否必选	参数类型	说明
traceId	是	String	服务内部的令牌，可用于在日志中追溯具体流程，调用失败无此字段。 在某些错误情况下可能没有此令牌字符串。
result	是	SASRResult	调用成功表示识别结果，调用失败时无此字段。请参考**表8-5**。

表 8-5 SASRResult

参数名	是否必选	参数类型	说明
text	是	String	调用成功表示识别出的内容。
score	是	Float	调用成功表示识别出的置信度，取值范围：0~1。

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
import SwiftUI
import AVFoundation
import SIS

struct Config {


    static let region = "cn-north-4"
    static let projectId = ""
}

class HTTPClientDelegate: HTTPDelegate, ObservableObject {

    @Published var result = ""
    func onMessage(response: SASRResponse) {
        self.result = response.result.text

    }

    func onError(response: SASRErrorResponse) {
        self.result = response.errorMsg
    }
}

enum STATE {
    case IDLE
    case RECORDING
}

struct SASRView: View {
    @ObservedObject var delegate = HTTPClientDelegate()
    @State var client: SASRClient?
    @State var recordStatus = STATE.IDLE
    @State var recorder: AudioFileRecorder?

    var body: some View {
        VStack {
            Button("开始录音") {
                do {
                    try AVAudioSession.sharedInstance().setCategory(.record)
                    try AVAudioSession.sharedInstance().setActive(true)
                } catch {
                    self.delegate.result = "初始化录音失败"
                    return
                }
                //认证用的AK和SK硬编码在代码中或明文存储都有很大安全风险，建议在配置文件或环境变量中密文存放，使用时解密，确保安全。
                //本示例以AK和SK保存在环境变量中来实现身份验证为例，运行本示例请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
                let ak = ProcessInfo.processInfo.environment["HUAWEICLOUD_SDK_AK"]!
                let sk = ProcessInfo.processInfo.environment["HUAWEICLOUD_SDK_SK"]!
                let authInfo = AuthInfo(ak: ak, sk: sk, region: Config.region, projectId: Config.projectId)
```

```
                self.client = SASRClient(auth: authInfo)
                self.client!.delegate = self.delegate
                self.recorder = AudioFileRecorder()
                self.recorder?.start()
                self.delegate.result = ""
                self.recordStatus = .RECORDING
            }
            .buttonStyle(.borderedProminent)
            .disabled(self.recordStatus == .RECORDING)

            Button("停止录音") {
                self.recorder?.stop()
                let filePath = self.recorder?.filePath
                let binData = try! Data(contentsOf: URL(fileURLWithPath: filePath!))
                let base64Data = binData.base64EncodedData()
                let strData = String(decoding: base64Data, as: UTF8.self)
                var config = SASRConfig()
                config.addPunc = "yes"
                config.digitNorm = "no"
                let sasrRequest = SASRRequest(config: config, data: strData)
                self.client!.transcribe(request: sasrRequest)
                self.recordStatus = .IDLE
            }
            .buttonStyle(.borderedProminent)
            .disabled(self.recordStatus == .IDLE)
            Text(delegate.result)
        }
        .padding()
    }
}

struct ContentView_Previews: PreviewProvider {
    static var previews: some View {
        SASRView()
    }
}
```

8.2 实时语音识别连续模式

前提条件

- 确保已经按照配置好iOS开发环境。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RASRClient，参数为AuthInfo和RASRConfig。

表 8-6 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。

表2 RASRConfig

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，SDK内置录音功能只支持pcm16k16bit，参见《API参考》中**开始识别**章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_16k_general，参见《API参考》中**开始识别**章节。
addPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vocabularyId	否	String	热词表id，不使用则不填写。 创建热词表请参考《API参考》中**创建热词表**章节。
interimResults	否	String	是否输出中间结果，可以为yes或no。默认为no，表示不输出中间结果。

响应参数

结果响应类为RASRResponse，详见**表8-7**。调用失败处理方法请参见**错误码**。

表 8-7 RASRResponse

参数名	参数类型	说明
respType	String	参数值为RESULT，表示识别结果响应。
traceId	String	服务内部的令牌，可用于在日志中追溯具体流程。
segments	Array of RASRSentence	多句结果，请参考**表8-8**。
errorCode	String	错误码。
errorMsg	String	错误描述。

表 8-8 RASRSentence

参数名	参数类型	说明
startTime	Integer	一句的起始时间戳，单位为ms。
endTime	Integer	一句的结束时间戳，单位为ms。

参数名	参数类型	说明
isFinal	Boolen	true表示是最终结果， false表示为中间临时结果。
result	RASRResult	调用成功表示识别结果，调用失败时无此字段。 请参考表8-9。

表 8-9 RASRResult

参数名	参数类型	说明
text	String	识别结果。
score	Float	识别结果的置信度，取值范围：0~1。此值仅会在最终结果时被赋值，在中间结果时统一置为“0.0”。 **说明** 目前置信度作用不是太大，请勿过多依赖此值。

示例代码

如下示例仅供参考，最新代码请前往SDK（websocket）章节获取并运行。

```
import SwiftUI
import AVFAudio
import SIS

struct Config {
    static let region = "cn-north-4"
    static let projectId = ""
}

class WebSocketDelegate: NSObject, WebSocketConnectionDelegate, ObservableObject {

    @Published var result = ""
    func onConnected(connection: WebSocketConnection) {
        print("connected")
    }

    func onDisconnected(connection: WebSocketConnection) {
        print("disconnected")
    }

    func onError(connection: WebSocketConnection, error: Error) {
        print(error.localizedDescription)
    }

    func onMessage(connection: WebSocketConnection, response: RASRResponse) {
        if response.respType == "RESULT" {
            self.result = response.segments![0].result.text
        }else if (response.respType == "ERROR"){
            self.result = response.errorMsg!
        }
    }
}
```

```
enum STATUS {
    case IDLE
    case TRANSCRIBING
}

struct RASRView: View {
    @ObservedObject var delegate = WebSocketDelegate()
    @State var client: RASRClient?
    @State var status = STATUS.IDLE

    var body: some View {
        VStack {
            Button("开始录音") {
                do {
                    try AVAudioSession.sharedInstance().setCategory(.record)
                    try AVAudioSession.sharedInstance().setActive(true)
                } catch {
                    self.delegate.result = "初始化录音失败"
                    return
                }

                //认证用的AK和SK硬编码在代码中或明文存储都有很大安全风险，建议在配置文件或环境变量中密文存放，使用时解密，确保安全。
                //本示例以AK和SK保存在环境变量中来实现身份验证为例，运行本示例请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
                let ak = ProcessInfo.processInfo.environment["HUAWEICLOUD_SDK_AK"]!
                let sk = ProcessInfo.processInfo.environment["HUAWEICLOUD_SDK_SK"]!
                let authInfo = AuthInfo(ak: ak, sk: sk, region: Config.region, projectId: Config.projectId)
                var config = RASRConfig()
                config.addPunc = "yes"
                config.digitNorm = "no"
                config.interimResults = "yes"
                self.client = RASRClient(auth: authInfo, config: config)
                self.delegate.result = ""
                self.client?.delegate = self.delegate
                self.client?.start()
                self.status = .TRANSCRIBING
            }
            .buttonStyle(.borderedProminent)
            .disabled(self.status == .TRANSCRIBING)

            Button("停止录音") {
                self.client?.stop()
                self.status = .IDLE
            }
            .buttonStyle(.borderedProminent)
            .disabled(self.status == .IDLE)
            Text(self.delegate.result)
        }
        .padding()
    }
}

struct RealTimeView_Previews: PreviewProvider {
    static var previews: some View {
        RASRView()
    }
}
```

9 Android SDK

9.1 一句话识别(http 版)

前提条件

- 确保已经按照配置好Android开发环境。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化SisClient,详细信息如下。

1. 配置客户端连接参数。
   - 默认配置
     ```
     // 使用默认配置
     HttpConfig config = HttpConfig.getDefaultHttpConfig();
     ```
   - 网络代理（可选）
     ```
     // 根据需要配置网络代理，网络代理默认的协议为 `http` 协议
     config.withProxyHost("proxy.huaweicloud.com")
         .withProxyPort(8080)
         .withProxyUsername("test")
         .withProxyPassword("test");
     ```
   - 超时配置（可选）
     ```
     // 默认连接超时时间为60秒，可根据需要调整
     config.withTimeout(60);
     ```
   - SSL配置（可选）
     ```
     // 根据需要配置是否跳过SSL证书验证
     config.withIgnoreSSLVerification(true);
     ```
2. 配置认证信息。

   配置AK、SK、project_id信息。华为云通过AK识别用户的身份，通过SK对请求数据进行签名验证，用于确保请求的机密性、完整性和请求者身份的正确性。
   - 使用永久AK和SK
     ```
     BasicCredentials basicCredentials = new BasicCredentials()
         .withAk(ak)
         .withSk(sk)
         .withProjectId(projectId);
     ```
   - 使用临时AK和SK


```
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk)
    .withSecurityToken(securityToken)
    .withProjectId(projectId)
```

认证参数说明：

- ak、sk：访问密钥信息，获取方法请参考**AK/SK认证**。
- projectId：华为云项目ID，获取方法请参考**获取项目ID**。
- securityToken：采用临时AK、SK 认证场景下的安全票据，可以**通过token获取**或者**通过委托授权获取**。

3. 初始化客户端（region和指定云服务endpoint二选一即可）。
   - 指定region方式（强烈推荐推荐）

```
// 初始化客户端认证信息，使用当前客户端初始化方式可不填 projectId/domainId，以初始化 BasicCredentials 为例
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

// 初始化指定云服务的客户端 {Service}Client ，以初始化 Region 级服务Sis的 SisClient 为例
SisClient client = SisClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(bhttps://developer.huaweicloud.com/endpoint?SISasicCredentials)
    .withRegion(SisRegion.valueOf("cn-north-4"))
    .build();
```

   - 指定云服务endpoint方式（可选）

```
// 指定终端节点，以Sis服务北京四的 endpoint 为例
String endpoint = "https://sis-ext.cn-north-4.myhuaweicloud.com";

// 初始化客户端认证信息，需要填写相应 projectId/domainId，以初始化 BasicCredentials 为例
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk)
    .withProjectId(projectId);

// 初始化指定云服务的客户端 {Service}Client ，以初始化 Region 级服务SiS的 SisClient 为例
SisClient client = SisClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(basicCredentials)
    .withEndpoint(endpoint)
    .build();
```

     endpoint是华为云各服务应用区域和各服务的终端节点，详情请查看 **地区和终端节点**。

请求参数

1. 请求类为RecognizeShortAudioRequest，该类的body参数为PostShortAudioReq。
2. PostShortAudioReq的包含data,和config两个参数，其中data为识别音频的base64格式的字符串。Config参数详见**表9-1**。

表 9-1 Config

参数	是否必选	参数类型	描述
audioFormat	是	String	支持语音的格式，请参考**表audio_format取值范围**。

参数	是否必选	参数类型	描述
property	是	String	所使用的模型特征串，通常是 “语种_采样率_领域”的形式，采样率需要与音频采样率保持一致，取值范围请参考**表 property取值范围**。
addPunc	否	String	表示是否在识别结果中添加标点，取值为“yes”和“no”，默认为“no”。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为“yes” 和 “no”，默认为“yes”。
vocabularyId	否	String	热词表id，不使用则不填写。 创建热词表信息请参考**创建热词表**。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

3. 伪代码

```
com.huaweicloud.sdk.sis.v1.model.Config configbody = new
com.huaweicloud.sdk.sis.v1.model.Config();configbody.setAudioFormat(com.huaweicloud.sdk.sis.v1.model.Config.AudioFormatEnum.fromValue("pcm16k16bit"));
configbody.setProperty(com.huaweicloud.sdk.sis.v1.model.Config.PropertyEnum.fromValue("chinese_16k_general"));
configbody.setAddPunc(com.huaweicloud.sdk.sis.v1.model.Config.AddPuncEnum.YES);
RecognizeShortAudioRequest request = new RecognizeShortAudioRequest();
PostShortAudioReq body = new PostShortAudioReq();body.withData(encoded);body.withConfig(configbody);
request.withBody(body);
```

发送请求

```
RecognizeShortAudioResponse response = client.recognizeShortAudio(request);
```

返回体RecognizeShortAudioResponse的参数如下表所示。

状态码： 200

表 9-2 响应 Body 参数

参数	是否必选	参数类型	描述
trace_id	是	String	服务内部的令牌，可用于在日志中追溯具体流程，调用失败无此字段。 在某些错误情况下可能没有此令牌字符串。
result	是	**Result** object	调用成功表示识别结果，调用失败时无此字段。

表 9-3 Result

参数	是否必选	参数类型	描述
text	是	String	调用成功表示识别出的内容。
score	是	Float	调用成功表示识别出的置信度，取值范围：0~1。
word_info	否	Array of **WordInfo** objects	分词信息列表。

表 9-4 WordInfo

参数	是否必选	参数类型	描述
start_time	否	Integer	起始时间。
end_time	否	Integer	结束时间。
word	否	String	分词。

状态码： 400

表 9-5 响应 Body 参数

参数	参数类型	描述
error_code	String	调用失败时的错误码。 调用成功时无此字段。
error_msg	String	调用失败时的错误信息。 调用成功时无此字段。

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2022-2022. All rights reserved.
 */
package com.huaweicloud.sis.android.demo.asr;

import java.io.IOException;

import java.util.Base64;

import android.app.Activity;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
```



```
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;

import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.sis.v1.SisClient;
import com.huaweicloud.sdk.sis.v1.model.PostShortAudioReq;
import com.huaweicloud.sdk.sis.v1.model.RecognizeShortAudioRequest;
import com.huaweicloud.sdk.sis.v1.model.RecognizeShortAudioResponse;
import com.huaweicloud.sdk.sis.v1.region.SisRegion;
import com.huaweicloud.sis.android.demo.Config;
import com.huaweicloud.sis.android.demo.R;
import com.huaweicloud.sis.android.demo.service.AudioRecordService;

/**
 * 功能描述
 * 一句话识别http
 *
 * @since 2022-07-11
 */
public class SasrHttpActivity extends Activity {
    private TextView result;
    private TextView title;
    private Button startButton;
    private Handler sasrHandler;

    private HttpConfig config;
    private SisClient client;

    private AudioRecordService audioRecordService;

    @Override
    protected void onStart() {
        super.onStart();
        initView();
        initResources();
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.sasr_http);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        audioRecordService.releaseAudioRecord();
    }


    /**
     * 绑定界面数据
     */
    private void initView() {
        title = findViewById(R.id.title);
        title.setText("一句话识别(http版)");
        result = findViewById(R.id.result);
        startButton = findViewById(R.id.start);
        startButton.setText("按住说话");
        // 按钮触及事件
        startButton.setOnTouchListener(new View.OnTouchListener() {
            @RequiresApi(api = Build.VERSION_CODES.O)
```

```
        @Override
        public boolean onTouch(View view, MotionEvent event) {
            switch (event.getAction()) {
                case MotionEvent.ACTION_UP: {
                    // 按钮松开之后触发事件
                    audioRecordService.stopAudioRecord();
                    recognitionAfterRecording();
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            Toast.makeText(getApplicationContext(), "录音结束识别中...",
Toast.LENGTH_SHORT).show();
                        }
                    });
                    break;
                }
                case MotionEvent.ACTION_DOWN: {
                    // 开始录音
                    audioRecordService.startRecording();
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            Toast.makeText(getApplicationContext(), "正在进行录音中...",
Toast.LENGTH_SHORT).show();
                        }
                    });
                    break;
                }
                default:
                    break;
            }
            return true;
        }
    });
}

/**
 * 初始化请求资源
 */
private void initResources() {
    BasicCredentials auth = new BasicCredentials()
            .withAk(this.getString(R.string.HUAWEICLOUD_SDK_AK))
            .withSk(this.getString(R.string.HUAWEICLOUD_SDK_SK))
            .withProjectId(Config.PROJECT_ID);
    config = HttpConfig.getDefaultHttpConfig();
    config.withIgnoreSSLVerification(true);
    client = SisClient.newBuilder()
            .withHttpConfig(config)
            .withCredential(auth)
            .withRegion(SisRegion.valueOf(Config.REGION))
            .build();
    audioRecordService = new AudioRecordService(16000);
    sasrHandler = new Handler(getMainLooper()) {
        @Override
        public void handleMessage(@NonNull Message message) {
            super.handleMessage(message);
            switch (message.what) {
                case 0:
                    Bundle bundle = message.getData();
                    result.setText(bundle.getString("result"));
                    break;
                default:
                    Log.e("Unexpected value: ", String.valueOf(message.what));
            }
        }
    };
}

/**
```

```
    * 初始化请求参数
    */
   private RecognizeShortAudioRequest getRecognizeShortAudioRequest() {
       String encoded = null;
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
           encoded =
Base64.getEncoder().encodeToString(audioRecordService.getByteArrayOutputStream().toByteArray());
       } else {
           encoded =
android.util.Base64.encodeToString(audioRecordService.getByteArrayOutputStream().toByteArray(), 2);
       }
       audioRecordService.closeByteArrayOutStream();
       com.huaweicloud.sdk.sis.v1.model.Config configbody = new com.huaweicloud.sdk.sis.v1.model.Config();

configbody.setAudioFormat(com.huaweicloud.sdk.sis.v1.model.Config.AudioFormatEnum.fromValue("pcm16
k16bit"));

configbody.setProperty(com.huaweicloud.sdk.sis.v1.model.Config.PropertyEnum.fromValue("chinese_16k_gen
eral"));
       configbody.setAddPunc(com.huaweicloud.sdk.sis.v1.model.Config.AddPuncEnum.YES);
       RecognizeShortAudioRequest request = new RecognizeShortAudioRequest();
       PostShortAudioReq body = new PostShortAudioReq();
       body.withData(encoded);
       body.withConfig(configbody);
       request.withBody(body);
       return request;
   }


   /**
    * 录音完成开始识别
    *
    * @throws IOException 输入异常
    */
   private void recognitionAfterRecording() {
       new Thread(() -> {
           RecognizeShortAudioRequest request = getRecognizeShortAudioRequest();
           String sasrRequestText = getSasrResponse(request);
           Message message = new Message();
           Bundle mBundle = new Bundle();
           mBundle.putString("result", sasrRequestText);
           message.setData(mBundle);
           sasrHandler.sendMessage(message);
       }).start();
   }


   /**
    * 一句话识别请求。
    *
    * @return 返回识别的文字
    */
   private String getSasrResponse(RecognizeShortAudioRequest request) {
       String resultStr = "";
       try {
           RecognizeShortAudioResponse response = client.recognizeShortAudio(request);
           resultStr = response.getResult().getText();
           Log.i("info:", response.getResult().getText());
       } catch (ConnectionException | RequestTimeoutException | ServiceResponseException e) {
           resultStr = e.toString();
           Log.i("error:", e.toString());
       }
       return resultStr;
   }
}
```

9.2 一句话识别(websocket 版)

前提条件

- 确保已经按照配置好Android开发环境。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化SasrWsClient，其中参数包含AuthInfo，SisHttpConfig，SasrWsResponseListener，SasrWsConnProcessListener。

表 9-6 SasrWsClient

参数	是否必选	参数类型	描述
AuthInfo	是	Object	鉴权信息类。
SisHttpConfig	是	Object	连接时网络的配置类。
SasrWsResponseListener	是	Object	webSocket回调过程中，业务逻辑的Listener。
SasrWsConnProcessListener	否	Object	webSocket生命周期的Listener。

其中AuthInfo和SisHttpConfig的参数如下表所示

表 9-7 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
serviceRegion	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
serviceEndPoint	否	String	终端节点，参考**地区和终端节点**。

表 9-8 SisHttpConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。
websocketWaitTimeout	否	Integer	webSocket返回数据时等待时间，默认20000，单位毫秒。
ProxyHostInfo	否	ProxyHostInfo	代理类。

表 9-9 ProxyHostInfo

参数名称	是否必选	参数类型	描述
userName	否	String	代理用户名（例：test）。
passWord	否	String	代理密码（例：test）。
hostName	否	String	代理地址（例：“proxy.huaweicloud.com”）。
port	否	int	代理端口号（例：8080）。

表 9-10 SasrWsResponseListener

函数	描述
void onExceededAudio();	识别时长超过一分钟时，响应，后续录入音频不再识别。
void onResponseError(AsrResponse response);	识别过程中出现异常，调用。
void onResponseEnd(AsrResponse response);	识别结束时回调。
void onResponseBegin(AsrResponse response);	识别开始时回调。
void onResponseMessage(AsrResponse message);	返回识别的结果。

表 9-11 SasrWsConnProcessListener

函数	描述
void onTranscriptionConnect()	webSocket连接建立后回调。
void onTranscriptionClose();	webSocket连接关闭后回调
void onTranscriptionFail(AsrResponse var1);	webSocket长连接连接失败时回调。

请求参数

请求类为SasrWsRequest，其中参数详见下表

表 9-12 SasrWsRequest

参数名称	是否必选	参数类型	描述
command	是	String	需设置为START，表示开始识别请求;发送END，表示识别结束请求。
config	是	Object	配置信息，详见下表。

表 9-13 Config

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，支持pcm，alaw，ulaw等，如pcm8k16bit，具体规格请参见《API参考》中开始识别章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_8k_common。
addPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
intermediateResult	否	String	是否显示中间结果，yes 或 no，默认no。
vocabularyId	否	String	热词表id，若没有则不填。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

建立连接

```
sasrWsClient.connect();
```

发送开始识别指令和配置信息

```
sasrWsClient.sendStart(getStartRequest());
```

发送识别数据

```
// data：发送byte数组
// byteSend :数组大小
// sleepTime : 休眠时间
sasrWsClient.sendByte(byte[] data, int byteSend, int sleepTime);
```

发送结束指令

```
sasrWsClient.sendEnd();
```

发送关闭连接请求

```
sasrWsClient.close();
```

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2022-2022. All rights reserved.
 */

package com.huaweicloud.sis.android.demo.asr;

import java.util.concurrent.atomic.AtomicBoolean;

import android.os.Bundle;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sis.android.demo.R;
import com.huaweicloud.sis.android.demo.service.AudioRecordService;
import com.huaweicloud.sis.android.demo.Config;

import sis.android.sdk.SasrWsClient;
import sis.android.sdk.bean.AuthInfo;
import sis.android.sdk.bean.SisHttpConfig;
import sis.android.sdk.bean.request.SasrWsRequest;
import sis.android.sdk.bean.response.AsrResponse;
import sis.android.sdk.exception.SisException;
import sis.android.sdk.listeners.SasrWsResponseListener;
import sis.android.sdk.listeners.process.SasrWsConnProcessListener;

/**
 * 功能描述
 * 一句话识别websocket
 *
 * @since 2022-07-11
 */
public class SasrWsActivity extends AppCompatActivity {
```

```
    private TextView result;
    private Button startButton;
    private AudioRecordService audioRecordService;
    private AuthInfo authInfo;
    private SasrWsClient sasrWsClient;

    // 实时显示识别的结果
    private StringBuffer realTimeResult;

    // 是否需要发送end请求
    private AtomicBoolean sendEndFlag = new AtomicBoolean(false);

    private SasrWsResponseListener sasrWsResponseListener = new SasrWsResponseListener() {
        @Override
        public void onExceededAudio() {
            sendEndFlag.set(true);
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(getApplicationContext(), "超过1分钟，识别结束...",
Toast.LENGTH_SHORT).show();
                }
            });
        }

        // 一句话识别 回调结果更新到界面UI中
        @Override
        public void onResponseMessage(AsrResponse asrResponse) {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    for (int i = 0; i < asrResponse.getSegments().size(); i++) {
                        AsrResponse.Segment segment = asrResponse.getSegments().get(i);
                        result.setText(realTimeResult.toString() + segment.getResult().getText());
                        if (segment.getIsFinal()) {
                            realTimeResult.append(segment.getResult().getText());
                        }
                    }
                }
            });
        }

        /**
         * 开始时响应
         *
         * @param response 返回体
         */
        @Override
        public void onResponseBegin(AsrResponse response) {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(getApplicationContext(), "连接成功，请录入音频...",
Toast.LENGTH_SHORT).show();
                }
            });
        }

        @Override
        public void onResponseEnd(AsrResponse response) {}

        /**
         * 传输过程中的错误时响应
         *
         * @param response 返回体
         */
        @Override
        public void onResponseError(AsrResponse response) {
            runOnUiThread(new Runnable() {
```

```
            @Override
            public void run() {
                // 调用失败给用户提示
                Toast.makeText(getApplicationContext(), JsonUtils.toJSON(response),
Toast.LENGTH_SHORT).show();
            }
        });
    }
};

private SasrWsConnProcessListener sasrWsConnProcessListener = new SasrWsConnProcessListener() {
    /**
     * 连接建立后回调
     */
    @Override
    public void onTranscriptionConnect() {
        Log.i("info", "长连接开始");
    }

    /**
     * 连接关闭后回调
     */
    @Override
    public void onTranscriptionClose() {
        Log.i("info", "长连接关闭");
    }

    /**
     * 长连接连接失败时回调
     *
     * @param asrResponse 返回体
     */
    @Override
    public void onTranscriptionFail(AsrResponse asrResponse) {
        Log.i("info", "长连接异常,等待中");
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                // 调用失败给用户提示
                Toast.makeText(getApplicationContext(), JsonUtils.toJSON(asrResponse),
Toast.LENGTH_SHORT).show();
            }
        });
    }
};


@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.sasr_websocket);
}

@Override
protected void onDestroy() {
    super.onDestroy();
    if (audioRecordService != null) {
        audioRecordService.releaseAudioRecord();
    }
}

@Override
protected void onStart() {
    super.onStart();
    initView();
    initResources();
}

/**
```

```
    * 初始化UI界面
    */
    private void initView() {
        result = findViewById(R.id.result);
        startButton = findViewById(R.id.start);
        startButton.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View view, MotionEvent event) {
                switch (event.getAction()) {
                    case MotionEvent.ACTION_UP:
                        audioRecordService.stopAudioRecord();
                        if (!sendEndFlag.get()) {
                            try {
                                sasrWsClient.sendEnd();
                            } catch (SisException e) {
                                Log.e("error", e.getErrorCode() + e.getErrorMsg());
                            }
                            sendEndFlag.set(false);
                        }
                        sasrWsClient.close();
                        break;
                    case MotionEvent.ACTION_DOWN:
                        try {
                            actionDown();
                        } catch (SisException e) {
                            Log.e("error", e.getErrorCode() + e.getErrorMsg());
                        }
                        break;
                    default:
                        break;
                }
                return true;
            }
        });
    }

    /**
     * 初始化请求资源
     */
    private void initResources() {
        authInfo = new AuthInfo(this.getString(R.string.HUAWEICLOUD_SDK_AK),
this.getString(R.string.HUAWEICLOUD_SDK_SK),
Config.REGION, Config.PROJECT_ID);
        audioRecordService = new AudioRecordService(16000);
        realTimeResult = new StringBuffer();
    }

    /**
     * 开始请求
     */
    private SasrWsRequest getStartRequest() {
        SasrWsRequest sasrWsRequest = new SasrWsRequest();
        sasrWsRequest.setCommand("START");
        SasrWsRequest.Config config = new SasrWsRequest.Config();
        config.setAudioFormat("pcm16k16bit");
        config.setProperty("chinese_16k_common");
        config.setAddPunc("yes");
        config.setInterimResults("yes");
        sasrWsRequest.setConfig(config);
        return sasrWsRequest;
    }

    /**
     * 按钮按下之后事件
     *
     * @throws SisException
     */
    private void actionDown() throws SisException {
        realTimeResult = realTimeResult.delete(0, realTimeResult.length());
```

```
        try {
            sasrWsClient = new SasrWsClient(authInfo, sasrWsResponseListener, sasrWsConnProcessListener,
new SisHttpConfig());
            audioRecordService.getAudioRecord().startRecording();
            sasrWsClient.connect();
            sasrWsClient.sendStart(getStartRequest());
            audioRecordService.startSendRecordingData(sasrWsClient);
        } catch (SisException e) {
            Log.e("error", e.getErrorCode() + e.getErrorMsg());
        }
    }
}
```

9.3 实时语音识别连续模式

前提条件

- 确保已经按照配置好Android开发环境。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RasrClient，其中参数包含
AuthInfo,SisHttpCnfig,RasrResponseListener,RasrConnProcessListener

表 9-14 RasrClient

参数名称	是否必选	参数类型	描述
AuthInfo	是	Object	鉴权信息类。
SisHttpCnfig	是	Object	连接时网络的配置类。
RasrResponseListener	是	Object	webSocket回调过程中，业务逻辑的Listener。
RasrConnProcessListener	否	Object	webSocket生命周期的Listener。

表 9-15 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
serviceRegion	是	String	区域，如cn-north-4，参考**终端节点**。

参数名称	是否必选	参数类型	描述
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
serviceEndPoint	否	String	终端节点，参考**地区和终端节点**。

表 9-16 SisHttpConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。
websocketWaitTimeout	否	Integer	webSocket返回数据时等待时间，默认20000，单位毫秒。
ProxyHostInfo	否	ProxyHostInfo	代理类。

表 9-17 ProxyHostInfo

参数名称	是否必选	参数类型	描述
userName	否	String	代理用户名（例：test）。
passWord	否	String	代理密码（例：test）。
hostName	否	String	代理地址（例：“proxy.huaweicloud.com”）。
port	否	int	代理端口号（例：8080）。

表 9-18 RasrResponseListener

函数	描述
void onResponseBegin(AsrResponse response);	识别开始时回调。
void onResponseEnd(AsrResponse response);	识别结束时回调。
void onResponseError(AsrResponse response);	识别过程中出现异常，调用。
void onResponseMessage(AsrResponse message);	返回识别的结果。
void onVoiceStart();	单句模式下，响应VOICE_START事件，表示检测到语音，此时IVR可以做打断（连续模式可忽略）。
void onVoiceEnd();	单句模式下，响应VOICE_END事件，表示一句话结束，后续的音频将被忽略，不会再进行识别（连续模式可忽略）。
void onExcceededSilence();	单句模式下，响应EXCEEDED_SILENCE事件，表示超过vad_head没有检测到声音，通常表示用户一直没有说话。此时后续的音频将被忽略，不会再进行识别（连续模式可忽略）。

表 9-19 RasrConnProcessListener

函数	描述
void onTranscriptionConnect()	webSocket连接建立后回调。
void onTranscriptionClose();	webSocket连接关闭后回调。
void onTranscriptionFail(AsrResponse var1);	webSocket长连接连接失败时回调。

请求参数

请求类为RasrRequest，其中参数详见下表

表 9-20 RasrRequest

参数名称	是否必选	参数类型	描述
command	是	String	需设置为START，表示开始识别请求;发送END，表示识别结束请求。
config	是	Object	配置信息，详见表9-21。

表 9-21 Config

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，支持pcm，alaw，ulaw等，如pcm8k16bit，参见《API参考》中开始识别章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_16k_general，参见《API参考》中开始识别章节。
addPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
digitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
vadHead	否	Integer	头部最大静音时间，[0, 60000]，默认10000ms。
vadTail	否	Integer	尾部最大静音时间，[0, 3000]，默认500ms。
maxSeconds	否	Integer	音频最长持续时间， [1, 60]，默认30s。
intermediateResult	否	String	是否显示中间结果，yes 或 no，默认no。例如分3次发送音频，选择no结果一次性返回，选择yes分三次返回。
vocabularyId	否	String	热词表id，若没有则不填。
needWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

设置当前 Client 为连续模式

```
rasrClient.rasrContinueStreamConnect();
```

建立连接

```
rasrClient.connect();
```

发送开始识别指令和配置信息

```
rasrClient.sendStart(getStartRequest());
```

发送识别数据

```
// data：发送byte数组
// byteSend :数组大小
// sleepTime : 休眠时间
rasrClient.sendByte(byte[] data, int byteSend, int sleepTime);
```

发送结束指令

```
rasrClient.sendEnd();
```

发送关闭连接请求

```
rasrClient.close();
```

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2022-2022. All rights reserved.
 */

package com.huaweicloud.sis.android.demo.asr;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.huaweicloud.sis.android.demo.Config;
import com.huaweicloud.sis.android.demo.service.AudioRecordService;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sis.android.demo.R;

import sis.android.sdk.RasrClient;
import sis.android.sdk.bean.AuthInfo;
import sis.android.sdk.bean.SisHttpConfig;
import sis.android.sdk.exception.SisException;
import sis.android.sdk.bean.request.RasrRequest;
import sis.android.sdk.bean.response.AsrResponse;
import sis.android.sdk.listeners.RasrResponseListener;

import sis.android.sdk.listeners.process.RasrConnProcessListener;

/**
 * 功能描述
 * 实时语音识别连续模式
 *
 * @since 2022-07-11
 */
public class RasrCsActivity extends AppCompatActivity {
    private TextView title;
    private TextView result;
```

```
    private Button startButton;
    private Button endButton;
    private RasrClient rasrClient;
    // 实时显示识别的结果
    private StringBuffer realTimeResult;

    private AudioRecordService audioRecordService;
    private AuthInfo authInfo;

    private RasrConnProcessListener rasrConnProcessListener = new RasrConnProcessListener() {
        /**
         * 连接关闭后回调
         */
        @Override
        public void onTranscriptionClose() {
            Log.i("info", "长连接关闭");
        }

        /**
         * 连接建立后回调
         */
        @Override
        public void onTranscriptionConnect() {
            Log.i("info", "长连接开始");
        }

        /**
         * 长连接连接失败时回调
         *
         * @param asrResponse 返回体
         */
        @Override
        public void onTranscriptionFail(AsrResponse asrResponse) {
            Log.i("info", "长连接异常");
            // 调用失败给用户提示
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(getApplicationContext(), "实时语音单句模式长连接失败" +
JsonUtils.toJSON(asrResponse), Toast.LENGTH_SHORT).show();
                }
            });
            rasrClient.close();
        }
    };

    private RasrResponseListener rasrResponseListener = new RasrResponseListener() {
        /**
         * 检测到句子开始事件
         */
        @Override
        public void onVoiceStart() {
        }

        /**
         * 检测到句子结束事件
         */
        @Override
        public void onVoiceEnd() {
        }

        /**
         * 返回识别的信息
         * @param message
         */
        @Override
        public void onResponseMessage(AsrResponse message) {
            runOnUiThread(new Runnable() {
                @Override
```

```
            public void run() {
                for (int i = 0; i < message.getSegments().size(); i++) {
                    AsrResponse.Segment segment = message.getSegments().get(i);
                    // 实时语音识别连续模式 回调结果更新到界面UI中
                    result.setText(realTimeResult.toString() + segment.getResult().getText());
                    if (segment.getIsFinal()) {
                        realTimeResult.append(segment.getResult().getText());
                    }
                }
            }
        });
    }

    /**
     *
     * 静音超长，也即没有检测到声音。响应事件
     */
    @Override
    public void onExcceededSilence() {
    }

    /**
     * 返回识别的信息
     * @param response
     */
    @Override
    public void onResponseBegin(AsrResponse response) {
    }

    @Override
    public void onResponseEnd(AsrResponse response) {
    }

    @Override
    public void onResponseError(AsrResponse response) {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                Toast.makeText(getApplicationContext(), "实时语音识别连续模式，错误响应:" +
JsonUtils.toJSON(response), Toast.LENGTH_SHORT).show();
            }
        });
        rasrClient.close();
    }
};


@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.rasr);
}

@Override
protected void onStart() {
    super.onStart();
    initResources();
    initView();
}

@Override
protected void onDestroy() {
    super.onDestroy();
    if (rasrClient != null) {
        rasrClient.close();
    }
    if (audioRecordService != null && audioRecordService.getIsRecording().get()) {
        audioRecordService.stopAudioRecord();
        audioRecordService.releaseAudioRecord();
```

```
    }
  }

  /**
   * 初始化界面
   */
  private void initView() {
      result = findViewById(R.id.result);
      title = findViewById(R.id.title);
      title.setText("实时语音识别连续模式");
      startButton = findViewById(R.id.start);
      endButton = findViewById(R.id.end);
      // 定义开始按钮点击事件
      startButton.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View view) {
              updateButtonState(startButton, false);
              updateButtonState(endButton, true);
              realTimeResult = realTimeResult.delete(0, realTimeResult.length());
              new Thread(new Runnable() {
                  @Override
                  public void run() {
                      try {
                          rasrClient = new RasrClient(authInfo, rasrResponseListener, rasrConnProcessListener, new
SisHttpConfig());
                          rasrClient.rasrContinueStreamConnect();
                          // 建立连接
                          rasrClient.connect();
                          rasrClient.sendStart(getStartRequest());
                          audioRecordService.startSendRecordingData(rasrClient);
                      } catch (SisException e) {
                          Log.e("error", e.getErrorCode() + e.getErrorMsg());
                      }
                  }
              }).start();
              Toast.makeText(getApplicationContext(), "正在进行录音中...", Toast.LENGTH_SHORT).show();
          }
      });
      // 结束按钮识别事件
      endButton.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View view) {
              updateButtonState(startButton, true);
              updateButtonState(endButton, false);
              if (audioRecordService.getIsRecording().get()) {
                  audioRecordService.stopAudioRecord();
                  Toast.makeText(getApplicationContext(), "识别结束...", Toast.LENGTH_SHORT).show();
              }
              try {
                  rasrClient.sendEnd();
              } catch (SisException e) {
                  Log.e("error", e.getErrorCode() + e.getErrorMsg());
              }
              rasrClient.close();
          }
      });
      updateButtonState(startButton, true);
      updateButtonState(endButton, false);
  }

  /**
   * 初始化设置资源
   */
  private void initResources() {
      authInfo = new AuthInfo(this.getString(R.string.HUAWEICLOUD_SDK_AK),
this.getString(R.string.HUAWEICLOUD_SDK_SK),
Config.REGION, Config.PROJECT_ID);
      audioRecordService = new AudioRecordService(16000);
      realTimeResult = new StringBuffer();
```

```
    }

    /**
     * 开始请求
     *
     * @return 返回请求体内容
     */
    private RasrRequest getStartRequest() {
        RasrRequest rasrRequest = new RasrRequest();
        rasrRequest.setCommand("START");
        RasrRequest.Config config = new RasrRequest.Config();
        config.setAudioFormat("pcm16k16bit");
        config.setProperty("chinese_16k_general");
        config.setAddPunc("yes");
        config.setInterimResults("yes");
        rasrRequest.setConfig(config);
        return rasrRequest;
    }

    // 用于设置按钮的状态
    private void updateButtonState(final Button btn, final boolean state) {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                btn.setEnabled(state);
            }
        });
    }
}
```

9.4 语音合成(http 版)

前提条件

- 确保已经按照配置好Android开发环境。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化SisClient,详细信息如下。

1. 配置客户端连接参数。
   - 默认配置
     ```
     // 使用默认配置
     HttpConfig config = HttpConfig.getDefaultHttpConfig();
     ```
   - 网络代理（可选）
     ```
     // 根据需要配置网络代理，网络代理默认的协议为 `http` 协议
     config.withProxyHost("proxy.huaweicloud.com")
         .withProxyPort(8080)
         .withProxyUsername("test")
         .withProxyPassword("test");
     ```
   - 超时配置（可选）
     ```
     // 默认连接超时时间为60秒，可根据需要调整
     config.withTimeout(60);
     ```
   - SSL配置（可选）
     ```
     // 根据需要配置是否跳过SSL证书验证
     config.withIgnoreSSLVerification(true);
     ```
2. 配置认证信息。

   配置AK、SK、project_id信息。华为云通过AK识别用户的身份，通过SK对请求数据进行签名验证，用于确保请求的机密性、完整性和请求者身份的正确性。

- 使用永久AK和SK

```
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk)
    .withProjectId(projectId);
```

- 使用临时AK和SK

```
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk)
    .withSecurityToken(securityToken)
    .withProjectId(projectId)
```

认证参数说明：

- ak、sk：访问密钥信息，获取方法请参考**AK/SK认证**。
- projectId：华为云项目ID，获取方法请参考**获取项目ID**。
- securityToken：采用临时AK、SK 认证场景下的安全票据，可以**通过token获取**或者**通过委托授权获取**。

3. 初始化客户端（region和指定云服务endpoint二选一即可）。

- 指定region方式（强烈推荐）

```
// 初始化客户端认证信息，使用当前客户端初始化方式可不填 projectId/domainId，以初始化 BasicCredentials 为例
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

// 初始化指定云服务的客户端 {Service}Client ，以初始化 Region 级服务SIS的 SisClient 为例
SisClient client = SisClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(basicCredentials)
    .withRegion(SisRegion.valueOf("cn-north-4"))
    .build();
```

- 指定云服务endpoint方式（可选）

```
// 指定终端节点，以SIS服务北京四的 endpoint 为例
String endpoint = "https://sis-ext.cn-north-4.myhuaweicloud.com";

// 初始化客户端认证信息，需要填写相应 projectId/domainId，以初始化 BasicCredentials 为例
BasicCredentials basicCredentials = new BasicCredentials()
    .withAk(ak)
    .withSk(sk)
    .withProjectId(projectId);

// 初始化指定云服务的客户端 {Service}Client ，以初始化 Region 级服务SIS的 CbsClient 为例
SisClient client = SisClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(basicCredentials)
    .withEndpoint(endpoint)
    .build();
```

endpoint是华为云各服务应用区域和各服务的终端节点，详情请查看 **地区和终端节点**。

请求参数

1. 请求类为RunTtsRequest,其中包含参数类PostCustomTTSReq，该类包含两个参数text(待合成文本)和TtsConfig，详见TtsConfig。

表 9-22 TtsConfig

参数名称	是否必选	参数类型	描述
audio_format	否	String	待合成的音频格式，可选mp3，wav等，默认wav。具体信息请参见《API参考》中**语音合成**章节。
pitch	否	Integer	音高，[-500,500]，默认是0。
speed	否	Integer	语速，[-500,500]，默认是0。
volume	否	Integer	音量，[0,100]，默认是50。
sample_rate	否	String	采样率，支持“8000”、“16000”，默认“8000”。
property	否	String	特征字符串，{language}_{speaker}_{domain}，默认chinese_xiaoqi_common。具体信息请参见《API参考》中**语音合成**章节。

2. 伪代码

```
TtsConfig configbody = new TtsConfig();
configbody.setAudioFormat(TtsConfig.AudioFormatEnum.fromValue("wav"));
configbody.setSampleRate(TtsConfig.SampleRateEnum.fromValue("8000"));
configbody.setProperty(TtsConfig.PropertyEnum.fromValue("chinese_huaxiaomei_common"));
RunTtsRequest request = new RunTtsRequest();
PostCustomTTSReq body = new PostCustomTTSReq();
body.withConfig(configbody);
if (!StringUtils.isEmpty(text.getText().toString())) {
    body.withText(text.getText().toString());
} else {
    body.withText("请输入合成文本");
}
request.withBody(body);
return request;
```

响应参数

响应类为RunTtsResponse，详见下表。调用失败处理方法请参见**错误码**。

表 9-23 RunTtsResponse

参数名	是否必选	参数类型	说明
trace_id	是	String	服务内部的令牌，可用于在日志中追溯具体流程，调用失败无此字段。 在某些错误情况下可能没有此令牌字符串。
result	是	Object	调用成功时为合成语音内容，请参考**表9-24**。 调用失败时无此字段。

表 9-24 Result

参数名	是否必选	参数类型	说明
data	是	String	合成后生成的语音数据，以Base64编码格式返回。用户如需生成音频，需要将Base64编码解码成byte数组，再保存为wav音频。

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2022-2022. All rights reserved.
 */

package com.huaweicloud.sis.android.demo.tts;

import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.StringUtils;
import com.huaweicloud.sdk.sis.v1.SisClient;
import com.huaweicloud.sdk.sis.v1.model.PostCustomTTSReq;
import com.huaweicloud.sdk.sis.v1.model.RunTtsRequest;
import com.huaweicloud.sdk.sis.v1.model.RunTtsResponse;
import com.huaweicloud.sdk.sis.v1.model.TtsConfig;
import com.huaweicloud.sdk.sis.v1.region.SisRegion;
import com.huaweicloud.sis.android.demo.R;
import com.huaweicloud.sis.android.demo.service.MediaPlayerService;
import com.huaweicloud.sis.android.demo.Config;

/**
 * 功能描述
 * 语音合成 http
 *
 * @since 2022-07-18
 */
public class SttsActivity extends AppCompatActivity {
    private EditText text;
    private TextView outResult;
    private Button startSoundRecording;
    private Button startPlay;
    private Handler handler;

    // 保存合成的base64字符串
```

```
private String base64Data;

// 合成音频路径
private String createFilePath;

// 客户端请求
private SisClient client;

private MediaPlayerService mediaPlayerService;


@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.stts);
}

@Override
protected void onStart() {
    super.onStart();
    initView();
    initResoureces();
}

@Override
protected void onDestroy() {
    mediaPlayerService.stopMyPlayer(createFilePath);
    super.onDestroy();
}

// 初始化界面
private void initView() {
    text = findViewById(R.id.input_text);
    outResult = findViewById(R.id.out_result);
    startSoundRecording = findViewById(R.id.start);
    startPlay = findViewById(R.id.startplay);
    startSoundRecording.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Runnable runnable = new Runnable() {
                @Override
                public void run() {
                    String sttsRequestText = getSttsResponse();
                    Message message = new Message();
                    Bundle mBundle = new Bundle();
                    mBundle.putString("result", sttsRequestText);
                    message.setData(mBundle);
                    handler.sendMessage(message);
                }
            };
            new Thread(runnable).start();
        }
    });
    startPlay.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (!StringUtils.isEmpty(base64Data)) {
                createFilePath = mediaPlayerService.createAudioFile(getBaseContext(), "wav", base64Data);
                mediaPlayerService.startPlay(createFilePath);
            }
        }
    });
}

/**
 * 初始化资源
 *
 * @return
 */
```

```
private void initResoureces() {
    BasicCredentials auth = new BasicCredentials()
                .withAk(this.getString(R.string.HUAWEICLOUD_SDK_AK))
                .withSk(this.getString(R.string.HUAWEICLOUD_SDK_SK))
        .withProjectId(Config.PROJECT_ID);
    HttpConfig config = HttpConfig.getDefaultHttpConfig();
    config.withIgnoreSSLVerification(true);
    client = SisClient.newBuilder()
        .withHttpConfig(config)
        .withCredential(auth)
        .withRegion(SisRegion.valueOf(Config.REGION))
        .build();
    handler = new Handler(getMainLooper()) {
        @Override
        public void handleMessage(@NonNull Message message) {
            super.handleMessage(message);
            switch (message.what) {
                case 0:
                    Bundle bundle = message.getData();
                    String rstr = bundle.getString("result");
                    outResult.setText(rstr);
                    break;
                default:
                    Log.e("Unexpected value: ", String.valueOf(message.what));
            }
        }
    };
    mediaPlayerService = new MediaPlayerService();
}


// 设置请求体
private RunTtsRequest getRunTtsRequest() {
    TtsConfig configbody = new TtsConfig();
    configbody.setAudioFormat(TtsConfig.AudioFormatEnum.fromValue("wav"));
    configbody.setSampleRate(TtsConfig.SampleRateEnum.fromValue("8000"));
    configbody.setProperty(TtsConfig.PropertyEnum.fromValue("chinese_huaxiaomei_common"));
    RunTtsRequest request = new RunTtsRequest();
    PostCustomTTSReq body = new PostCustomTTSReq();
    body.withConfig(configbody);
    if (!StringUtils.isEmpty(text.getText().toString())) {
        body.withText(text.getText().toString());
    } else {
        body.withText("请输入合成文本");
    }
    request.withBody(body);
    return request;
}

// 发送请求
private String getSttsResponse() {
    RunTtsRequest request = getRunTtsRequest();
    String ttsString = "";
    try {
        RunTtsResponse response = client.runTts(request);
        if (response.getResult().getData() != null) {
            base64Data = response.getResult().getData();
            ttsString = "合成成功";
        } else {
            ttsString = "合成失败";
        }
        Log.i("info", ttsString);
    } catch (ConnectionException | RequestTimeoutException | ServiceResponseException e) {
        Log.e("error", e.toString());
    } catch (Exception e) {
        Log.e("error", e.toString());
    }
    return ttsString;
}
```

```
    @Override
    protected void onPause() {
        mediaPlayerService.stopMyPlayer(createFilePath);
        super.onPause();
    }
}
```

9.5 语音合成(webSocket 版)

前提条件

- 确保已经按照配置好Android开发环境。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RttsClient，其中参数包含AuthInfo和SisHttpConfig和RttsResponseListener。其中AuthInfo和SisHttpConfig的参数如**表9-25**所示。

表 9-25 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
serviceRegion	是	String	区域，如cn-north-4，参考**终端节点**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
serviceEndPoint	否	String	终端节点，参考**地区和终端节点**。

表 9-26 SisHttpConfig

参数名称	是否必选	参数类型	描述
connectionTimeout	否	Integer	连接超时，默认10000，单位ms。
readTimeout	否	Integer	读取超时，默认10000，单位ms。
websocketWaitTimeout	否	Integer	webSocket返回数据时等待时间，默认20000，单位毫秒。

参数名称	是否必选	参数类型	描述
ProxyHostInfo	否	ProxyHostInfo	代理类。

表 9-27 ProxyHostInfo

参数名称	是否必选	参数类型	描述
userName	否	String	代理用户名（例：test）。
passWord	否	String	代理密码（例：test）。
hostName	否	String	代理地址（例：“proxy.huaweicloud.com”）。
port	否	int	代理端口号（例：8080）。

其中RttsResponseListener是用户自定义的，建立webSocket之后，接收服务端返回消息的Listener。

表 9-28 RttsResponseListener

函数名称	作用
void onTranscriptionConnect();	webSocket建立连接后后调。
void onTranscriptionClose();	webSocket连接关闭后回调。
void onTranscriptionFail(RttsResponse response);	长连接连接失败时回调。
void onTranscriptionBegin(RttsResponse response);	开始合成音频数据时回调。
void onTranscriptionEnd(RttsResponse response);	合成音频数据结束时回调。

函数名称	作用
void onTranscriptionError(RttsResponse response);	合成音频数据过程中失败时回调。
void onTranscriptionResponse(byte[] bytes);	返回合成的二进制数据。

请求参数

请求类为RttsRequest，其中参数包含text,command,Config,详见**表9-29**。

表 9-29 RttsRequest

参数名称	是否必选	参数类型	描述
text	是	String	待合成的文本，文本长度限制小于500字符。
command	是	String	需设置为START，表示开始识别请求。
Config	是	String	配置信息。可参照**表9-30**。

表 9-30 Config

参数名称	是否必选	参数类型	描述
audio_format	否	String	待合成的音频格式，可选mp3，wav等，默认wav。具体信息请参见《API参考》中**语音合成**章节。
pitch	否	Integer	音高，[-500,500]，默认是0。
speed	否	Integer	语速，[-500,500]，默认是0。
volume	否	Integer	音量，[0,100]，默认是50。
sample_rate	否	String	采样率，支持“8000”、“16000”，默认“8000”。
property	否	String	特征字符串，{language}_{speaker}_{domain}，默认chinese_xiaoqi_common。具体信息请参见《API参考》中**语音合成**章节。

建立连接，发送合成数据

```
rttsClient.connect();
rttsClient.sendData(rttsRequest);
```

发送关闭连接请求

```
rttsClient.close();
```

代码示例

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2022-2022. All rights reserved.
 */

package com.huaweicloud.sis.android.demo.tts;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import com.cloud.sdk.util.StringUtils;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sis.android.demo.R;
import com.huaweicloud.sis.android.demo.service.AudioTrackService;
import com.huaweicloud.sis.android.demo.Config;
import com.huaweicloud.sis.android.demo.service.AudioTrackServiceCallback;

import sis.android.sdk.RttsClient;
import sis.android.sdk.bean.AuthInfo;
import sis.android.sdk.bean.SisHttpConfig;
import sis.android.sdk.bean.request.RttsRequest;
import sis.android.sdk.bean.response.RttsResponse;
import sis.android.sdk.exception.SisException;
import sis.android.sdk.listeners.RttsResponseListener;

/**
 * 功能描述
 * 语音合成 websocket
 *
 * @since 2022-07-18
 */
public class RttsActivity extends AppCompatActivity {
    private EditText textView;
    private Button startPlay;

    private RttsClient rttsClient;

    private SisHttpConfig sisHttpConfig;

    private AudioTrackService audioTrackService;
    private AudioTrackServiceCallback audioPlayerCallback;

    private AuthInfo authInfo;

    private static int dataAcceptanceTime = 1000 * 60 * 5;

    private RttsResponseListener rttsResponseListener = new RttsResponseListener() {
        @Override
        public void onTranscriptionConnect() {
```

```
            Log.i("info", "建立连接后回调");
        }

        @Override
        public void onTranscriptionClose() {
            Log.i("info", "关闭连接后回调");
        }

        @Override
        public void onTranscriptionFail(RttsResponse response) {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(getApplicationContext(), "出现异常" + JsonUtils.toJSON(response),
Toast.LENGTH_SHORT).show();
                }
            });
            updateButtonState(startPlay, true);
            Log.i("info", "长连接失败后回调" + JsonUtils.toJSON(response));
        }

        @Override
        public void onTranscriptionBegin(RttsResponse response) {
            Log.i("info", "开始合成时的响应事件" + response.toString());
        }

        @Override
        public void onTranscriptionEnd(RttsResponse response) {
            rttsClient.close();
            audioTrackService.setPlayState(AudioTrackService.PlayState.playEnd);
        }

        @Override
        public void onTranscriptionError(RttsResponse response) {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(getApplicationContext(), "出现异常" + JsonUtils.toJSON(response),
Toast.LENGTH_SHORT).show();
                }
            });
            updateButtonState(startPlay, true);
            Log.i("info", "合成时发生错误的响应事件" + response.toString());
        }

        @Override
        public void onTranscriptionResponse(byte[] bytes) {
            Log.i("info", "合成过程中返回的二进制流");
            audioTrackService.setAudioData(bytes);
        }
    };

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.rtts);
    }

    @Override
    protected void onStart() {
        super.onStart();
        initView();
        initResources();
    }


    @Override
    protected void onDestroy() {
        super.onDestroy();
```

```
        if (audioTrackService != null) {
            audioTrackService.stop();
            audioTrackService.releaseTrack();
        }
    }

    /**
     * 初始化UI
     */
    private void initView() {
        textView = findViewById(R.id.itext);
        startPlay = findViewById(R.id.startplay);
        startPlay.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                updateButtonState(startPlay, false);
                new Thread(new Runnable() {
                    @Override
                    public void run() {
                        try {
                            audioTrackService.play();
                            rttsClient = new RttsClient(authInfo, sisHttpConfig, rttsResponseListener);
                            RttsRequest rttsRequest = rttsRequets();
                            rttsClient.connect();
                            rttsClient.sendData(rttsRequest);
                        } catch (SisException e) {
                            Log.e("error", e.getErrorCode() + e.getErrorMsg());
                        }
                    }
                }).start();
            }
        });
    }

    // 用于设置按钮的状态
    private void updateButtonState(final Button btn, final boolean state) {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                btn.setEnabled(state);
            }
        });
    }

    /**
     * 初始化资源
     */
    private void initResources() {
        authInfo = new AuthInfo(this.getString(R.string.HUAWEICLOUD_SDK_AK),
this.getString(R.string.HUAWEICLOUD_SDK_SK),
Config.REGION, Config.PROJECT_ID);
        sisHttpConfig = new SisHttpConfig();
        // 设置等待事件为5分钟
        sisHttpConfig.setWebsocketWaitTimeout(dataAcceptanceTime);
        audioPlayerCallback = new AudioTrackServiceCallback() {
            @Override
            public void playStart() {

            }

            @Override
            public void playOver() {
                updateButtonState(startPlay, true);
            }
        };
        audioTrackService = new AudioTrackService(audioPlayerCallback);
    }
```

```
    /**
     * 功能描述
     * 语音合成websocket 版
     */
    private RttsRequest rttsRequets() {
        RttsRequest rttsRequest = new RttsRequest();
        RttsRequest.Config config = new RttsRequest.Config();
        config.setAudioFormat("pcm");
        config.setProperty("chinese_huaxiaomei_common");
        config.setSampleRate("16000");
        config.setPitch(0);
        config.setVolume(50);
        rttsRequest.setCommand("START");
        rttsRequest.setConfig(config);
        String text = textView.getText().toString().trim();
        if (!StringUtils.isNullOrEmpty(text)) {
            rttsRequest.setText(text);
        } else {
            rttsRequest.setText("请输入合成音频");
        }
        return rttsRequest;
    }
}
```

10 CPP SDK（Windows）

10.1 使用实时语音识别

前提条件

- 确保已按照**配置CPP环境（Windows）**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RasrClient，其参数包括AuthInfo。

表 10-1 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

请求参数

请求类为RasrRequest，详见**表 RasrRequest**。

表 10-2 RasrRequest

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，支持pcm等，如pcm8k16bit，参见《API参考》中**开始识别**章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_8k_common，参见《API参考》中**开始识别**章节。

通过set方法可以设置具体参数，详见**表 RasrRequest设置参数**

表 10-3 RasrRequest 设置参数

方法名称	是否必选	参数类型	描述
SetPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
SetDigitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
SetVadHead	否	Integer	头部最大静音时间，[0, 60000]，默认10000ms。
SetVadTail	否	Integer	尾部最大静音时间，[0, 3000]，默认500ms。
SetMaxSeconds	否	Integer	音频最长持续时间， [1, 60]，默认30s。
SetIntermediateResult	否	String	是否显示中间结果，yes 或 no，默认no。
SetVocabularyId	否	String	热词表id，若没有则不填。
SetNeedWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
#include <iostream>

#include "RasrClient.h"
#include "RasrRequest.h"
#include "IoUtil.h"
```

```
void OnConnect() {
    std::cout << "now rasr Connect success" << std::endl;
}

void OnStart(std::string text) {
    std::cout << "now rasr receive start response: " << text << std::endl;
}

void OnResp(std::string text) {
    // text encoded by utf-8 contains chinese character, which will cause error code. So we should convert to
ansi
    // cout << "rasr receive " << text << endl;
    std::cout << "now rasr receive " << Utf8ToAnsi(text) << std::endl;
}

void OnEnd(std::string text) {
    std::cout << "now rasr receive end response: " << text << std::endl;
}


void OnClose() {
    std::cout << "now rasr receive Close" << std::endl;
}

void OnError(string text) {
    std::cout << "now rasr receive error: " << text << std::endl;
}

void OnEvent(string text) {
    std::cout << "now rasr receive event: " << text << std::endl;
}

void RasrTest() {
    // 1. config parameter
    // 1.1 init authInfo
        // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密
文存放，使用时解密，确保安全;
        // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量
HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
    std::string ak = GetEnv("HUAWEICLOUD_SDK_AK");
    std::string sk = GetEnv("HUAWEICLOUD_SDK_SK");
    string region = "";
    string projectId = "";
    AuthInfo authInfo(ak, sk, region, projectId);

    // 1.2 config Connect parameter
    HttpConfig httpConfig;
    httpConfig.SetReadTimeout(20000);
    httpConfig.SetConnectTimeout(20000);

    // 1.3 config callback, callback function are optional, if not set, it will use function in RasrListener
    WebsocketService::ptr websocketServicePtr = websocketpp::lib::make_shared<WebsocketService>();
    websocketServicePtr->SetOnConnectFunc(OnConnect);       // Connect success callback
    websocketServicePtr->SetOnStartFunc(OnStart);           // receive start response callback
    websocketServicePtr->SetOnRespFunc(OnResp);             // receive transcribe result callback
    websocketServicePtr->SetOnEndFunc(OnEnd);               // receive end response callback
    websocketServicePtr->SetOnCloseFunc(OnClose);           // Close callback
    websocketServicePtr->SetOnEventFunc(OnEvent);           // receive event callback
    websocketServicePtr->SetOnErrorFunc(OnError);           // receive error callback

    // 1.4 config request parameter
    RasrRequest request("pcm16k16bit", "chinese_16k_general");
    request.SetIntermediateResult("no");


    // 2. init client
    RasrClient* rasrClient = new RasrClient(authInfo, websocketServicePtr, httpConfig);

    // 3. create connection :ContinueStreamConnect/ShortStreamConnect/SentenceStreamConnect
```

```
    rasrClient->ContinueStreamConnect();

    // 4. send start
    rasrClient->SendStart(request);

    // 5. send binary audio. (filePtr, fileLength, byteLen, SleepTime ). If the audio is generated by recording,
then it should set sleep time 0.
    int fileLength;
    std::string filePath = "../sisCppSdkDemo/123.wav";
    unsigned char* buff = ReadBinary(filePath, &fileLength);
    if (buff == nullptr) {
        cout << filePath << " read file failed";
        rasrClient->Close();
        delete rasrClient;
        return;
    }
    rasrClient->SendBinary(buff, fileLength, 3200, 50);

    // 6. send end
    rasrClient->SendEnd();

    // 7. close
    rasrClient->Close();

    delete[] buff;
    delete rasrClient;
}

int main() {
    RasrTest();
    return 0;
}
```

10.2 使用实时语音合成

前提条件

- 确保已按照**配置CPP环境（Windows）**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RttsClient，其参数包括AuthInfo

表 10-4 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

请求参数

请求类为RttsRequest，详见**表 RttsRequest**。

表 10-5 RttsRequest

参数名称	是否必选	参数类型	描述
text	是	String	待合成文本，不超过500字。

通过set方法可以设置具体参数，详见**表 RttsRequest设置参数**

表 10-6 RttsRequest 设置参数

方法名称	是否必选	参数类型	描述
SetAudioFormat	否	String	设置语音格式，默认pcm。
SetAudioProperty	否	String	设置语音合成特征字符串，{language}_{speaker}_{domain}，即“语种_人员标识_领域”。默认chinese_xiaoyan_common 。详见**API文档**。
SetSampleRate	否	String	设置采样率：8000、16000，默认8000。
SetPitch	否	Integer	设置音高，-500~500，默认0。
SetVolume	否	Integer	设置音量，0~100，默认50。
SetSpeed	否	Integer	设置语速，-500~500，默认0。

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
#include "IoUtil.h"
#include "RttsClient.h"
#include "RttsRequest.h"

void OnRttsConnect() {
    std::cout << "now rtts client Connect success" << std::endl;
}

void OnRttsStart(std::string text) {
    std::cout << "now rtts client receive start response: " << text << std::endl;
}

void OnRttsEnd(std::string text) {
    std::cout << "now rtts client  receive end response: " << text << std::endl;
}
```

```
void OnRttsClose() {
    std::cout << "now rtts client receive Close" << std::endl;
}

void OnRttsError(std::string text) {
    std::cout << "now  rtts client receive error: " << text << std::endl;
}


void OnRttsBinary(std::string binaryData) {
    // data content can be available by data() method, data length can be available by size() method
    // const char* data = binaryData.data();
    // int dataLength = binaryData.size();
    std::cout << "now rtts client receive binary data " << binaryData.size() << std::endl;
}

void RttsTest() {
    // 1. config parameter
    // 1.1 init authInfo
        // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
       // 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK
    std::string ak = GetEnv("HUAWEICLOUD_SDK_AK");
    std::string sk = GetEnv("HUAWEICLOUD_SDK_SK");
    std::string region = "";
    std::string projectId = "";
    AuthInfo authInfo(ak, sk, region, projectId);

    // 1.2 config Connect parameter
    HttpConfig httpConfig;
    httpConfig.SetReadTimeout(20000);
    httpConfig.SetConnectTimeout(20000);

    // 1.3 config callback, callback function are optional, if not set, it will use function in RttsListner
    WebsocketService::ptr websocketServicePtr = websocketpp::lib::make_shared<WebsocketService>();
    websocketServicePtr->SetOnConnectFunc(OnRttsConnect);       // Connect success callback
    websocketServicePtr->SetOnStartFunc(OnRttsStart);           // receive start response callback
    websocketServicePtr->SetOnEndFunc(OnRttsEnd);             // receive end response callback
    websocketServicePtr->SetOnCloseFunc(OnRttsClose);          // Close callback
    websocketServicePtr->SetOnErrorFunc(OnRttsError);         // receive error callback
    websocketServicePtr->SetOnBinaryFunc(OnRttsBinary);     // receive binary callback

    // 1.3 option, use RttsListener, which can save file; You can edit RttsListener.h to finish your own business
    //WebsocketService::ptr websocketServicePtr = websocketpp::lib::make_shared<WebsocketService>();
    //RttsListener rttsListener;
    //rttsListener.SetSaved(true);
    //rttsListener.SetFilePath("d:/test5.pcm");
        //websocketServicePtr->SetRttsListener(rttsListener);


    // 1.4 config request parameter
    std::string text = AnsiToUtf8("华为致力于把数字世界带入每个人每个家庭每个组织，构建万物互联的智能世界。");
    RttsRequest request(text);
    request.SetAudioFormat("pcm");
    request.SetVolume(50);
    request.SetSpeed(0);
    request.SetPitch(0);
    request.SetSampleRate("8000");
    request.SetAudioProperty("chinese_xiaoyan_common");


    // 2. init client
    RttsClient* rttsClient = new RttsClient(authInfo, websocketServicePtr, httpConfig);

    // 3. send request
    rttsClient->Synthesis(request);
```

```
    // wait for save file, if setSaved false in rttsListener or don't use rttsListener, it can be removed.
    std::this_thread::sleep_for(std::chrono::milliseconds(2000));
    delete rttsClient;

}

int main(){
    RttsTest();
    return 0;
}
```

11 CPP SDK（Linux）

11.1 使用实时语音识别

前提条件

- 确保已按照**配置CPP环境（Linux）**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RasrClient，其参数包括AuthInfo

表 11-1 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

请求参数

请求类为RasrRequest，详见**表 RasrRequest**。

表 11-2 RasrRequest

参数名称	是否必选	参数类型	描述
audioFormat	是	String	音频格式，支持pcm等，如pcm8k16bit，参见《API参考》中**开始识别**章节。
property	是	String	属性字符串，language_sampleRate_domain， 如chinese_8k_common，参见《API参考》中**开始识别**章节。

通过set方法可以设置具体参数，详见**表 RasrRequest设置参数**

表 11-3 RasrRequest 设置参数

方法名称	是否必选	参数类型	描述
SetPunc	否	String	表示是否在识别结果中添加标点，取值为yes 、no，默认no。
SetDigitNorm	否	String	表示是否将语音中的数字识别为阿拉伯数字，取值为yes 、 no，默认为yes。
SetVadHead	否	Integer	头部最大静音时间，[0, 60000]，默认10000ms。
SetVadTail	否	Integer	尾部最大静音时间，[0, 3000]，默认500ms。
SetMaxSeconds	否	Integer	音频最长持续时间， [1, 60]，默认30s。
SetIntermediateResult	否	String	是否显示中间结果，yes 或 no，默认no。
SetVocabularyId	否	String	热词表id，若没有则不填。
SetNeedWordInfo	否	String	表示是否在识别结果中输出分词结果信息，取值为“yes”和“no”，默认为“no”。

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2020-2020. All rights reserved.
 */

#include "Utils.h"
#include "RasrClient.h"
#include "gflags/gflags.h"
```

```
// auth info
// refer to https://support.huaweicloud.com/api-sis/sis_03_0051.html
// 认证用的AK和SK硬编码在代码中或明文存储都有很大安全风险，建议在配置文件或环境变量中密文存放，使用时解密，确保安全。
DEFINE_string(ak, "", "access key");
DEFINE_string(sk, "", "secrect key");

// region, for example cn-east-3, cn-north-4
DEFINE_string(region, "cn-east-3", "project region, such as cn-east-3");
// projectId, refer to https://support.huaweicloud.com/api-sis/sis_03_0008.html
DEFINE_string(projectId, "", "project id");
// endpoint, relevant to region, sis-ext.${region}.myhuaweicloud.com
DEFINE_string(endpoint, "", "service endpoint");

DEFINE_string(audioFormat, "pcm16k16bit", "such pcm16k16bit alaw16k16bit etc.");
DEFINE_string(property, "chinese_16k_general", "");
DEFINE_string(audioPath, "xx.wav", "audio path");

DEFINE_int32(chunkSize, 3000, "bytes per send");
DEFINE_int32(sampleRate, 16000, "sample rate of audio");
DEFINE_int32(readTimeOut, 20000, "read time out, default 20s");
DEFINE_int32(connectTimeOut, 20000, "connecting time out, default 20s");
DEFINE_int32(bytesPerSecond, 32000, "32000 bytes per second");

void OnOpen()
{
    LOG(INFO) << "now rasr Connect success";
}

void OnStart(std::string text)
{
    LOG(INFO) << "now rasr receive start response: " << text;
}

void OnResp(std::string text)
{
    // text encoded by utf-8 contains chinese character, which will cause error code. So we should convert to ansi
    LOG(INFO) << "rasr receive " << text;
}

void OnEnd(std::string text)
{
    LOG(INFO) << "now rasr receive end response: " << text;
}

void OnClose()
{
    LOG(INFO) << "now rasr receive Close";
}

void OnError(std::string text)
{
    LOG(INFO) << "now rasr receive error: " << text;
}

void OnEvent(std::string text)
{
    LOG(INFO) << "now rasr receive event: " << text;
}

void RasrTest(const std::string filePath)
{
    const int sleepTime = FLAGS_bytesPerSecond / FLAGS_chunkSize;

    speech::huawei_asr::AuthInfo authInfo(FLAGS_ak, FLAGS_sk, FLAGS_region, FLAGS_projectId, FLAGS_endpoint);
    // config Connect parameter
```

```
    speech::huawei_asr::HttpConfig httpConfig;
    httpConfig.SetReadTimeout(FLAGS_readTimeOut);
    httpConfig.SetConnectTimeout(FLAGS_connectTimeOut);

    // config callback, callback function are optional, if not set, it will use function in RasrListener
    speech::huawei_asr::WebsocketService::ptr websocketServicePtr =
        websocketpp::lib::make_shared<speech::huawei_asr::WebsocketService>();
    websocketServicePtr->SetOnConnectFunc(OnOpen); // Connect success callback
    websocketServicePtr->SetOnStartFunc(OnStart);  // receive start response callback
    websocketServicePtr->SetOnRespFunc(OnResp);    // receive transcribe result callback
    websocketServicePtr->SetOnEndFunc(OnEnd);      // receive end response callback
    websocketServicePtr->SetOnCloseFunc(OnClose);  // Close callback
    websocketServicePtr->SetOnEventFunc(OnEvent);  // receive event callback
    websocketServicePtr->SetOnErrorFunc(OnError);  // receive error callback


    // step1 create client
    std::shared_ptr<speech::huawei_asr::RasrClient> rasrClient =
        std::make_shared<speech::huawei_asr::RasrClient>(authInfo, websocketServicePtr, httpConfig);

    // step2 connect, just select one mode, the following is continue stream connect.
    rasrClient->ContinueStreamConnect();
    // short stream connect
    // rasrClient->ShortStreamConnect();
    // sentence stream connect
    // rasrClient->SentenceStreamConnect();

    // step3 construct request params
    speech::huawei_asr::RasrRequest request(FLAGS_audioFormat, FLAGS_property);
    // set whether to add punctuation, yes or no, default no, optional operation.
    request.SetPunc("no");
    // set whether to transcribe number into arabic numerals, yes or no, default yes,optional operation.
    request.SetDigitNorm("yes");
    // set vad head, max silent head, [0, 60000], default 10000, optional operation.
    request.SetVadHead(10000);
    // set vad tail, max silent tail, [0, 3000], default 500, optional operation.
    request.SetVadTail(500);
    // set max seconds of one sentence, [1, 60], default 30, optional operation.
    request.SetMaxSeconds(30);
    // set whether to return intermediate result, yes or no, default no. optional operation.
    request.SetIntermediateResult("no");
    // set whether to return word_info, yes or no, default no. optional operation.
    request.SetNeedWordInfo("no");
    // set vocabulary_id, it should be filled only if it exists or it will report error
    // request.SetVocabularyId("");

    // step4 send start
    rasrClient->SendStart(request);

    // step5 send audio
    std::string audioContent;
    int ret = speech::huawei_asr::ReadBinary(filePath, audioContent);
    if (ret != 0) {
        LOG(ERROR) << "RasrDemo running failed";
        rasrClient->Close();
        return;
    }
    unsigned char *buf = (unsigned char *)(audioContent.c_str());
    rasrClient->SendBinary(buf, audioContent.size(), FLAGS_chunkSize, sleepTime);

    // step5 send end
    rasrClient->SendEnd();

    // step6 close
    rasrClient->Close();
}

int main(int argc, char *argv[])
{
```

```
    FLAGS_alsologtostderr = true;
    FLAGS_log_dir = "./logs";
    gflags::ParseCommandLineFlags(&argc, &argv, true);
    google::InitGoogleLogging(argv[0]);
    RasrTest(FLAGS_audioPath);
    return 0;
}
```

编译脚本

以下编译脚本仅供参考，您可以根据实际业务需求，对RasrDemo.cpp进行定制修改。

```
cd ${project_dir}
mkdir build && cd build
mkdir logs
cmake ..
make -j
./RasrDemo --audioPath=yourAudioPath --ak=yourAk --sk=yourSk --region=yourRegion --
projectId=yourProjectId
```

11.2 使用实时语音合成

前提条件

- 确保已按照**配置CPP环境（Linux）**配置完毕。
- 请参考**SDK（websocket）**获取最新版本SDK包。

初始化 Client

初始化RttsClient，其参数包括AuthInfo

表 11-4 AuthInfo

参数名称	是否必选	参数类型	描述
ak	是	String	用户的ak，可参考**AK/SK认证**。
sk	是	String	用户的sk，可参考**AK/SK认证**。
projectId	是	String	项目ID，同region一一对应，参考**获取项目ID**。
region	是	String	区域，如cn-north-4，参考**终端节点**。
endpoint	否	String	终端节点，参考**地区和终端节点**。一般使用默认即可。

请求参数

请求类为RttsRequest，详见**表 RttsRequest**。

表 11-5 RttsRequest

参数名称	是否必选	参数类型	描述
text	是	String	待合成文本。

通过set方法可以设置具体参数，详见**表 RttsRequest设置参数**

表 11-6 RttsRequest 设置参数

方法名称	是否必选	参数类型	描述
SetAudioFormat	否	String	设置语音格式，默认pcm。
SetAudioProperty	否	String	设置语音合成特征字符串，{language}_{speaker}_{domain}，即“语种_人员标识_领域”。默认chinese_xiaoyan_common 。详见**API文档**。
SetSampleRate	否	String	设置采样率：8000、16000，默认8000。
SetPitch	否	Integer	设置音高，-500~500，默认0。
SetVolume	否	Integer	设置音量，0~100，默认50。
SetSpeed	否	Integer	设置语速，-500~500，默认0。
SetSubtitle	否	String	设置字幕，部分发音人支持字幕时间戳，详见**API文档**。

示例代码

如下示例仅供参考，最新代码请前往**SDK（websocket）**章节获取并运行。

```
#include "RttsClient.h"
#include "RttsRequest.h"
#include "gflags/gflags.h"
// refer to https://support.huaweicloud.com/api-sis/sis_03_0115.html
// auth info
// 认证用的AK和SK硬编码在代码中或明文存储都有很大安全风险，建议在配置文件或环境变量中密文存放，使用时解密，确保安全。
DEFINE_string(ak, "", "access key");
DEFINE_string(sk, "", "secrect key");
// region, for example cn-east-3, cn-north-4
DEFINE_string(region, "cn-north-4", "project region, such as cn-north-4");
// projectId, refer to https://support.huaweicloud.com/api-sis/sis_03_0008.html
DEFINE_string(projectId, "", "project id");
DEFINE_string(text, "华为致力于把数字世界带入每个人每个家庭每个组织，构建万物互联的智能世界。", "Text to be synthesized");
DEFINE_string(audioFormat, "pcm", "audio format, such pcm");
DEFINE_string(property, "chinese_xiaoyan_common", "");
DEFINE_string(audioPath, "test.pcm", "audio saved path");
```

```
DEFINE_string(sampleRate, "16000", "sample rate of audio");
DEFINE_string(subtitle, "", "subtitle info");
DEFINE_int32(volume, 50, "");
DEFINE_int32(speed, 0, "");
DEFINE_int32(pitch, 0, "");
DEFINE_int32(readTimeOut, 20000, "read time out, default 20s. Increase this value appropriately according
to the length of the text");
DEFINE_int32(connectTimeOut, 20000, "connecting time out, default 20s");
DEFINE_bool(isSaved, true, "save the audio as a local file");
class CallBack : public RttsListener {
public:
    void OnConnect() {
        LOG(INFO) << "rtts Connect success";
    }
    void OnStart(std::string text) {
        LOG(INFO) << "rtts receive start response " << text;
    }
    void OnResp(std::string binaryData) {
        if (isSaved) {
            dataContents.push_back(binaryData);
        }
        LOG(INFO) << "rtts receive data " << binaryData.size();
    }
    void onResponseSubtitle(std::string message) {
        LOG(INFO) << message;
    }
    void OnEnd(std::string text) {
        LOG(INFO) << "rtts receive end response " << text;
        if (isSaved) {
            std::ofstream fout(filePath,  std::ios::binary);
            if (!fout.is_open()) {
                LOG(INFO) << "filePath " << filePath << " is invalid";
                return;
            }
            for (int i = 0; i < dataContents.size(); i++) {
                fout.write(dataContents[i].data(), dataContents[i].size());
            }
            fout.close();
            LOG(INFO) << "success to save file in " << filePath;
        }
    }
    void OnClose() {
        LOG(INFO) << "rtts receive Close";
    }
    void OnError(std::string text) {
        LOG(INFO) << "rtts receive error" << text;
    }
    void SetFilePath(std::string fPath) {
        filePath = fPath;
    }
    void SetSaved(bool saved) {
        isSaved = saved;
    }
private:
    std::string filePath;
    bool isSaved = false;
    std::vector<std::string> dataContents;
};
void RttsTest() {
    // 1. config parameter
    // 1.1 init authInfo
    speech::huawei_asr::AuthInfo authInfo(FLAGS_ak, FLAGS_sk, FLAGS_region, FLAGS_projectId,
FLAGS_endpoint);
    // 1.2 config Connect parameter
    speech::huawei_asr::HttpConfig httpConfig;
    httpConfig.SetReadTimeout(FLAGS_readTimeOut);
    httpConfig.SetConnectTimeout(FLAGS_connectTimeOut);
    // 1.3 config callback, callback function are optional, if not set, it will use function in RttsListner
    speech::huawei_asr::WebsocketService::ptr websocketServicePtr =
```

```
websocketpp::lib::make_shared<speech::huawei_asr::WebsocketService>();
    CallBack callback;
    callback.SetSaved(FLAGS_isSaved);
    callback.SetFilePath(FLAGS_audioPath);
    websocketServicePtr->SetTtsCallBack(&callback);
    // 1.4 config request parameter
    speech::huawei_tts::RttsRequest request(FLAGS_text);
    request.SetAudioFormat(FLAGS_audioFormat);
    request.SetVolume(FLAGS_volume);
    request.SetSpeed(FLAGS_speed);
    request.SetPitch(FLAGS_pitch);
    request.SetSampleRate(FLAGS_sampleRate);
    request.SetAudioProperty(FLAGS_property);
    request.SetSubtitle(FLAGS_subtitle);
    // 2. init client
    speech::huawei_tts::RttsClient* rttsClient = new speech::huawei_tts::RttsClient(authInfo,
websocketServicePtr, httpConfig);
    // 3. send request
    rttsClient->Synthesis(request);
    // wait for save file, if setSaved false in rttsListener or don't use rttsListener, it can be removed.
    std::this_thread::sleep_for(std::chrono::milliseconds(2000));
    delete rttsClient;
}
int main(int argc, char *argv[]) {
    FLAGS_alsologtostderr = true;
    FLAGS_log_dir = "./logs";
    gflags::ParseCommandLineFlags(&argc, &argv, true);
    google::InitGoogleLogging(argv[0]);
    RttsTest();
    return 0;
}
```

编译脚本

以下编译脚本仅供参考，您可以根据实际业务需求，对RasrDemo.cpp进行定制修改。

```
cd ${project_dir}
mkdir build && cd build
mkdir logs
cmake ..
make -j
./RttsDemo --ak=yourAk --sk=yourSk --region=yourRegion --projectId=yourProjectId --isSaved=true  --
audioPath=test.pcm
```

12 附录

12.1 示例音频

测试音频如**表 示例音频**所示，音频文件标题表示采样率和位宽。如8k16bit.pcm表示音频采样率为8k，位宽为16bit。

表 12-1 示例音频

音频格式	下载链接
mp3	**https://sis-sample-audio.obs.cn-north-1.myhuaweicloud.com/16k16bit.mp3**
wav	**https://sis-sample-audio.obs.cn-north-1.myhuaweicloud.com/16k16bit.wav**
pcm	**https://sis-sample-audio.obs.cn-north-1.myhuaweicloud.com/16k16bit.pcm**
pcm	**https://sis-sample-audio.obs.cn-north-1.myhuaweicloud.com/8k16bit.pcm**

13 修订记录

发布日期	修订说明
2023-09-27	新增： iOS SDK新增一句话识别、实时语音识别连续模式
2023-03-06	新增： CPP SDK支持Linux版本
2022-08-02	新增： Java SDK新增实时语音合成 Python SDK新增实时语音合成 CPP SDK新增实时语音合成
2022-07-07	新增： 获取录音文件识别结果API支持返回提交音频的时长。 CPP SDK上线。
2022-06-08	新增： 实时语音合成支持温柔女声、朝气男声精品发音人。
2022-05-31	新增： 语音合成新增朝气男声发音人。
2022-02-15	新增： 录音文件识别输入参数兼容公网访问的url。
2021-08-28	SDK发布1.7.0版本，新增以下章节： 一句话识别Websocket接口
2021-06-16	SDK发布1.6.0版本，新增以下章节： 录音文件极速版

发布日期	修订说明
2020-08-20	SDK发布1.3.0版本，修改以下章节： ● Java SDK ● Python SDK
2020-07-17	新增“digit_norm”字段，修改以下章节： ● 一句话识别 ● 录音文件识别 ● 实时语音识别
2020-04-21	新增： 热词管理章节
2019-11-29	新增： Python SDK章节
2019-11-18	整改SDK手册
2019-09-25	新增： 语音合成章节
2019-08-06	新增： 语音识别章节 修改： Runtime Exception修改为Checked Exception
2019-07-30	新增： 实时语音识别章节
2019-07-02	下线了ASR SDK的长语音识别功能。
2019-06-10	重新封装SDK，进行了重构，加入重试机制。
2019-03-30	第一次正式发布。