

# Android\_CPP\_SDK\_实时语音识别

提示:

- 在使用SDK之前，请先确保已阅读了 [接口说明文档](#)。

## 前提条件

- 使用SDK前，首先阅读接口说明，详情请参见 [接口说明](#)。
- 已获取项目Appkey，详情请参见 [创建项目](#)。
- 已获取Access Token，详情请参见 [获取Token概述](#)。

## 下载安装

1. [请到Android SDK页面下载SDK](#)，包含头文件和链接库文件

**重要** 请下载后在样例初始化代码中替换您的阿里云账号信息、Appkey和Token才可运行。

类别	兼容范围
系统	支持Android 4.0 以上版本，API LEVEL 14
架构	armeabi-v7a, arm64-v8a, x86, x86_64

此SDK还包含如下功能，若未支持您想要的功能，请去对应文档获取SDK。

功能	是否支持
一句话识别	是
实时语音识别	是
语音合成	是
实时长文本语音合成	是
流式文本语音合成	是
离线语音合成	否
录音文件识别极速版	是
唤醒及命令词	否
听悟实时推流	是

2. 解压ZIP包，在`android_libs`目录下获得SDK的动态库，在`android_include`目录下获得SDK的头文件。

## SDK关键接口

- `nui_initialize`：初始化SDK。

复制代码

```
/**
 * 初始化SDK，SDK为单例，请先释放后再次进行初始化。请勿在UI线程调用，意外下可能引起阻塞。
 * @param parameters: 初始化参数，参考接口说明：https://help.aliyun.com/zh/isi/developer-reference/api-reference-1
 * @param listener: 事件监听回调，参考下文具体回调
 * @param async_listener: 异步回调，设置nullptr采用同步方式调用
 * @param level: log打印级别，值越小打印越多
 * @param save_log: 是否保存log为文件，存储目录为parameter中的debug_path字段值。注意，log文件无上限，
  请注意持续存储导致磁盘存满。
 * @return 参考错误码：https://help.aliyun.com/zh/isi/developer-reference/overview-3
 */
NuiResultCode nui_initialize(const char *parameters,
                             const NuiSdkListener *listener,
                             const NuiAsyncCallback *async_listener = nullptr,
                             NuiSdkLogLevel level = LOG_LEVEL_VERBOSE,
                             bool save_log = false);
```

- `NuiSdkListener`类型

名称	类型	说明
<code>event_callback</code>	<code>FuncDialogListenerOnEvent</code>	NUI事件回调，参考下文
<code>user_data_callback</code>	<code>FuncDialogUserProvideData</code>	NUI麦克风数据请求回调，参考下文
<code>audio_state_changed_callback</code>	<code>FuncDialogAudioStateChange</code>	NUI麦克风状态回调，参考下文
<code>audio_extra_event_callback</code>	<code>FuncDialogAudioExtraEvent</code>	NUI特殊事件回调，暂不使用
<code>user_data</code>	<code>void *</code>	用户数据，上述回调中第一个参数

- `FuncDialogAudioStateChange`：根据音频状态进行录音功能的开关。

复制代码

```
/**
 * 当start/stop/cancel等接口调用时，SDK通过此回调通知App进行录音的开关操作
 * @param user_data: 暂不使用
 * @param state: 录音需要的状态（打开/关闭）
 */
typedef void (*FuncDialogAudioStateChange)(void *user_data,
```

```
NuiAudioState state);
```

- FuncDialogUserProvideData：在回调中提供音频数据。

▼ 复制代码

```
/**
 * 当开始识别时，此回调被连续调用，App需要在回调中进行语音数据填充
 * @param user_data: 暂不使用
 * @param buffer: 用于给用户填充语音的存储区
 * @param len: 需要填充语音的字节数
 * @return 实际填充的字节数
 */
typedef int (*FuncDialogUserProvideData)(void *user_data, char *buffer,
                                         int len);
```

- FuncDialogListenerOnEvent：SDK事件回调。

▼ 复制代码

```
/**
 * SDK主要事件回调
 * @param user_data: 暂不使用
 * @param event: 回调事件，参见如下事件列表
 * @param dialog: 会话编号，暂不使用
 * @param wuw: 语音唤醒功能使用，暂不使用
 * @param asr_result: 语音识别结果
 * @param finish: 本轮识别是否结束标志
 * @param code: 参考错误码，在EVENT_ASR_ERROR时有效
 * @param all_response: 此事件完整信息
 */
typedef void (*FuncDialogListenerOnEvent)(void *user_data,
                                         NuiCallbackEvent event, long dialog,
                                         const char *wuw,
                                         const char *asr_result, bool finish,
                                         int code, const char *all_response);
```

- 事件列表：

名称	说明
EVENT_VAD_START	检测到人声起点
EVENT_VAD_END	检测到人声尾点
EVENT_ASR_PARTIAL_RESULT	语音识别中间结果
EVENT_ASR_RESULT	语音识别最终结果
EVENT_ASR_ERROR	根据错误码信息判断出错原因
EVENT_MIC_ERROR	录音错误

EVENT_SENTENCE_START	实时语音识别事件，表示检测到一句话开始。
EVENT_SENTENCE_END	实时语音识别事件，表示检测到一句话结束，返回一句的完整结果。
EVENT_SENTENCE_SEMANTICS	暂不使用。
EVENT_TRANSCRIBER_COMPLETE	停止语音识别后最终事件。

- `nui_set_params`：以json格式设置SDK参数。参数参考接口说明文档。

复制代码

```

/**
 * 以json格式设置参数.
 * @param params: 参考接口说明:https://help.aliyun.com/zh/isi/developer-reference/api-reference
 * @param async_listener: 异步回调，设置nullptr采用同步方式调用
 * @return 参考错误码: https://help.aliyun.com/zh/isi/developer-reference/overview-4
 */
NuiResultCode nui_set_params(const char *params,
                             const NuiAsyncCallback *listener = nullptr);

```

- `nui_dialog_start`：开始识别。

复制代码

```

/**
 * 开始识别
 * @param vad_mode: 多种模式，对于识别场景，使用MODE_P2T即可
 * @param dialog_params: 设置识别参数，可不设置。参考接口说明:https://help.aliyun.com/zh/isi/developer-reference/api-reference
 * @param async_listener: 异步回调，设置nullptr采用同步方式调用
 * @return 参考错误码: https://help.aliyun.com/zh/isi/developer-reference/overview-4
 */
NuiResultCode nui_dialog_start(NuiVadMode vad_mode, const char *dialog_params,
                                const NuiAsyncCallback *listener = nullptr);

```

- `nui_dialog_cancel`：结束识别。

复制代码

```

/**
 * 结束识别，调用该接口后，服务端将返回最终识别结果并结束任务
 * @param force: 是否强制结束忽略最终结果，false表示停止但是等待完整结果返回
 * @param async_listener: 异步回调，设置nullptr采用同步方式调用
 * @return 参考错误码: https://help.aliyun.com/zh/isi/developer-reference/overview-4
 */
NuiResultCode nui_dialog_cancel(bool force,
                                const NuiAsyncCallback *listener = nullptr);

```

- `nui_release`：释放SDK。

复制代码

```

/**
 * 释放SDK资源
 * @param async_listener: 异步回调，设置nullptr采用同步方式调用
 * @return 参考错误码
 */
NuiResultCode nui_release(const NuiAsyncCallback *async_listener = nullptr);

```

- nui\_get\_version: 获得当前SDK版本信息。

 复制代码

```

/**
 * 获得当前SDK版本信息
 * @return 字符串形式的SDK版本信息
 */
const char *nui_get_version(const char *module = nullptr);


```

## 调用顺序

1. 初始化SDK，初始化录音实例。
2. 根据业务需求配置参数。
3. 调用nui\_dialog\_start开始识别。
4. 根据音频状态回调audio\_state\_changed\_callback打开录音机。
5. 在user\_data\_callback回调中提供录音数据。
6. 在EVENT\_SENTENCE\_START事件回调中表示当前开始识别一个句子，在EVENT\_ASR\_PARTIAL\_RESULT事件回调中获取识别中间结果，在EVENT\_SENTENCE\_END事件回调中获得这句话完整的识别结果和各相关信息。
7. 调用nui\_dialog\_cancel可以结束识别并从EVENT\_TRANSCRIBER\_COMPLETE事件回调确认已停止识别。
8. 结束调用使用nui\_release接口释放SDK资源。

## 代码示例

### NUI SDK 初始化

 复制代码

```

void nuiDialogListenerOnEvent(void *user_data, NuiCallbackEvent event, long dialog,
                             const char *wuw, const char *asr_result, bool finish,
                             int code, const char *response) {}

void nuiDialogAudioStateChange(void *user_data, NuiAudioState state) {}

int nuiDialogUserProvideData(void *user_data, char *buffer, int len) {
    return 0;
}

// 此初始化参数可参考《Android SDK》中genInitParams和《接口说明》

```

```
// https://help.aliyun.com/zh/isi/developer-reference/nui-sdk-for-android
// https://help.aliyun.com/zh/isi/developer-reference/overview-3
const char* init_params = "\"app_key\", \"<您申请创建的app_key>\",
                          \"token\", \"<服务器生成的具有时效性的临时凭证>\",
                          \"url\", \"wss://nls-gateway.cn-shanghai.aliyuncs.com:443/ws/v1\",
                          \"device_id\", \"empty_device_id\",
                          \"workspace\", \"<资源目录路径>\",
                          \"service_mode\", \"1\"";

//nui listener
NuiSdkListener nuiListener;
nuiListener.event_callback = nuiDialogListenerOnEvent;
nuiListener.audio_state_changed_callback = nuiDialogAudioStateChange;
nuiListener.audio_extra_event_callback = nullptr;
nuiListener.user_data = nullptr;
nuiListener.user_data_callback = nuiDialogUserProvideData;

bool save_log = false;
nui_initialize(init_params, &nuiListener, nullptr, LOG_LEVEL_VERBOSE, save_log);
```

其中init\_params生成为String json串，包含资源目录和用户信息，具体含义请参考[接口说明文档](#)。

## 参数设置

以json字符串进行设置，具体参见[《接口说明》](#)和[《Android SDK》](#)中的genParams。参数设置需要在nui\_dialog\_start前调用。

▼
复制代码

```
const char* init_params = "\"service_type\": \"4\",
                          \"nls_config\": \"{\\\"enable_intermediate_result\\\":true}\"";
nui_set_params(init_params, nullptr);
```

## 开启对话

通过nui\_dialog\_start接口开启监听。

▼
复制代码

```
const char* param_string = "";
nui_dialog_start(MODE_P2T, param_string, nullptr);
```

## 回调处理

### FuncDialogAudioStateChange

录音状态回调，SDK内部维护录音状态，调用时根据该状态的回调进行录音机的开关操作。

▼
复制代码

```
void onNuiAudioStateChanged(void *user_data, nuiSdk::NuiAudioState state) {
```

```
TLog("onNuiAudioStateChanged state=%u", state);
if (state == STATE_CLOSE || state == STATE_PAUSE) {
    //stop recorder
} else if (state == STATE_OPEN){
    //start recorder
}
}
```

## FuncDialogUserProvideData

录音数据回调，在该回调中填充录音数据。

▼ [复制代码](#)

```
int onNuiNeedAudioData(void *user_data, char *audio_buffer, int len) {
    //copy data into audio_buffer and return the copied length.
}
```

## FuncDialogListenerOnEvent

NuiSdk事件回调，请勿在事件回调中调用SDK的接口，可能引起死锁。

▼ [复制代码](#)

```
void onNuiEventCallback(void *user_data,
                        nuisdk::NuiCallbackEvent nuiEvent, long dialog,
                        const char *wuw,
                        const char *asr_result, bool finish,
                        int code, const char *all_response) {
    printf("onNuiEventCallback event %d finish %d", nuiEvent);
    if (nuiEvent == nuisdk::EVENT_ASR_PARTIAL_RESULT || nuiEvent == nuisdk::EVENT_SENTENCE_END) {
        printf("ASR RESULT %s finish %d", asr_result, finish);
    } else if (nuiEvent == nuisdk::EVENT_ASR_ERROR) {
        printf("EVENT_ASR_ERROR error[%d]", code);
    } else if (nuiEvent == nuisdk::EVENT_MIC_ERROR) {
        printf("MIC ERROR");
    }

    if (finish) {
        //dialog finished
    }

    return;
}
```

## 结束识别

▼ [复制代码](#)

```
nui_dialog_cancel(false, nullptr);
```

