linux-sdk 使用说明

接口说明

sdk语音相关的接口

```
sdk 初始化函数，主要包括sdk参数初始化和授权两个过程，这个函数是阻塞的，
授权的超时时间为 15 秒。

struct dds_client *dds_client_init (const char *config_json);

参数说明:

@ config_json: 配置选项，json 格式的字符串；

返回值:

错误情况下返回NULL，否则返回 struct dds_client 实例指针；
```

```
运行 sdk 函数，调用此函数之后就可以语音交互了。

int dds_client_start(struct dds_client *ds, ddsLintener cb,
void *user);

参数说明:

@ ds: 由 dds_client_init 返回的 struct dds_client 实例指针；
@ cb: 监听 sdk 事件的回调函数；
@ user: 用户参数；

返回值:

出错返回 -1
```

```
释放 sdk 实例的函数:

void dds_client_release(struct dds_client *ds);

参数说明:

@ ds: 由 dds_client_init 返回的 struct dds_client 实例指针；
```

下面的接口必须在 `dds_client_init` 和 `dds_client_start` 正确返回之后才能正确执行。

```
向 sdk 内部发送消息:

int dds_client_publish(struct dds_client *ds, int ev,
const char *data);

参数说明:
@ ds: 由 dds_client_init 返回的 struct dds_client 实例指针；
@ ev: 发送的消息事件；
@ data: 此消息附带的数据，json 格式；


返回值:

只有当 sdk 完成初始化并且授权成功之后才能正确返回，否则返回 -1
```

```
返回 nativeAPI 查询结果的接口

int dds_client_resp_nativeapi(struct dds_client *ds,
const char *native_api, const char *native_api_data_json);

参数说明:

ds: sdk 实例指针；

native_api: 这个 nativeAPI topic 名；
```

native_api_data_json: 查询数据结果，json string，格式如下，
"duiWidget" 字段必须包含， 且目前取值为 "text"。 用户自定义的数据必须放在
extra 字段中。

```
{
    "duiWidget": "text",
    "extra": {
        "xx": "11"
    }
}
```

返回值：如果 sdk 没有初始化完成或者授权成功则返回-1

内部合成的接口：

```
int dds_client_speak(struct dds_client *ds, const char *text);
```

参数说明：

ds：sdk 实例指针

text：需要合成的文本

返回值：如果 sdk 没有初始化完成或者授权成功则返回-1

外部 feed 音频接口：

```
int dds_client_feed_audio(struct dds_client *ds, char *data, int len);
```

参数说明：

ds：sdk 实例指针

data：录音机数据

len：数据长度

返回值： 出错返回 -1 ，此接口只有在 recorder 配置为外部方式才会生效。

停止当前对话，包括停止合成，取消识别等。

```
int dds_client_stop_dialog(struct dds_client *ds);
```

参数说明：

ds：sdk 实例指针

text：需要合成的文本

返回值：如果 sdk 没有初始化完成或者授权成功则返回-1

关闭唤醒，如果在语音对话过程中调用此接口，会在这条对话自然结束之后才会禁止唤醒。

```
int dds_client_disable_wakeup(struct dds_client *ds);
```

参数说明：

ds：sdk 实例指针

返回值：如果 sdk 没有初始化完成或者授权成功则返回-1

打开唤醒

```
int dds_client_enable_wakeup(struct dds_client *ds);
```

参数说明：

ds：sdk 实例指针

返回值：如果 sdk 没有初始化完成或者授权成功则返回-1

设置用户唤醒词

```
int dds_client_update_customword(struct dds_client *ds,
const char *word);

参数说明:
ds: sdk 实例指针
word: 唤醒词配置，格式是 json string，说明如下:

    {
            "greetingFile":"path:../res/tts/help.mp3", 可选
            "greeting": "我在，有什么可以帮你", 可选
            "pinyin": "ni hao xiao chi", 必选
            "name": "你好小驰", 必选
            "threshold": 0.127 必选
        }

此函数成功返回后，唤醒词的相关配置会更新到 config.json 文件。
对于客户端异常断电可能导致 config.json 文件破坏的话， 需要开发者自己来避免，
比如采用备份文件的机制。
```

获取当前的唤醒词

```
char* dds_client_get_wakeupwords(struct dds_client *ds);

参数说明:
ds: sdk 实例指针

此函数返回字符串指针， 开发者需要主动释放内存。 返回字符串为json格式，如下:

{
    "majorword": [{
        "greetingFile": "path:../res/tts/help.mp3",
        "greeting": "我在，有什么可以帮你",
        "pinyin": "ni hao xiao le",
        "name": "你好小乐",
        "threshold": 0.144000
    }],
    "minorword": [{
        "greetingFile": "path:../res/tts/help.mp3",
        "greeting": "我在，有什么可以帮你",
        "pinyin": "ni hao xiao chi",
        "name": "你好小驰",
        "threshold": 0.127000
    }],
    "cmdword": [{
        "pinyin": "jiang di yin liang",
        "threshold": 0.100000,
        "action": "decrease.volume",
        "name": "降低音量"
    }],
    "customword": [{
        "pinyin": "ni hao tiam mao",
        "name": "你好天猫",
        "threshold": 0.200000
    }]
}

majorword 为主唤醒词，minorword 为副唤醒词， cmdword 为命令词，
customword 为用户定义唤醒词。 其实就是 config.json 文件里面的配置。
```

```
// 获取当前的 tts 发音人，出错返回 NULL
char *dds_client_get_speaker(struct dds_client *ds);

// 获取当前的 tts 播报速度，为 float 型， 在 0 ~ 1 之间，越大表示速度越慢。
float dds_clent_get_speed(struct dds_client *ds);

// 获取当前的 tts 的播报音量大小， 为 int 型， 在 0 ~ 100 之间。
int dds_client_get_volume(struct dds_client *ds);

// 设置当前的 tts 的播报音色人，出错返回 -1
int dds_client_set_speaker(struct dds_client *ds, char *speaker);
```

```
// 设置当前的 tts 的播报速度大小，出错返回 -1
int dds_client_set_speed(struct dds_client *ds, float speed);

// 设置当前的 tts 的播报音量大小，出错返回 -1
int dds_client_set_volume(struct dds_client *ds, int vol);
```

声纹的相关接口

```
// 获取当前的声纹详细信息

char *dds_client_vprint_get_detail(struct dds_client *ds);
```

返回的是一个 json 格式字符串，格式为：

```
{
    "vNum": 2,
    "detail": [
    {
        "name":"test1"
    },
    {
        "name":"test2"
    }
    ]
}
```

```
// 开始进入声纹注册的接口

int dds_client_vprint_regist(struct dds_client *ds, char *name);
```

name 参数表示声纹注册人的姓名标识。

出错返回-1，通常是格式错误。而声纹注册过程中的详细错误将会通过回调抛出。

```
// 删除注册人信息

int dds_client_vprint_unregist(struct dds_client *ds, char *name);
```

出错返回-1，通常是格式错误。而声纹删除过程中的详细错误将会通过回调抛出。

能量接口

```
int dds_client_energy_estimate(struct dds_client* ds, int second);
```

second 参数表示接下来所要计算的音频时间长度，单位为秒。计算结果通过回调给出。

出错返回 -1

客户端信息上传的相关接口

```
int dds_client_upload_city(struct dds_client *dc, char *city);
```

city 为当前设备所在的城市名，比如苏州，上海等

出错返回 -1

sdk回调消息接口

回调函数	消息	含义	参数
ddsLintener	local_wakeup.result	唤醒事件	json string，形如 {"type":"major","greeting":"好的", "word":"你好小驰"}
ddsLintener	doa.result	doa事件	json string, 形如 {"dao": 100}
ddsLintener	sys.vad.begin	vad开始的事件	无
ddsLintener	sys.vad.end	vad结束的事件	无

ddsLintener	sys.tts.begin	合成音开始的事件	无
ddsLintener	sys.tts.end	合成音结束的事件，播放结束	无
ddsLintener	sys.asr.begin	sdk内部开始做识别	无
ddsLintener	asr.speech.text	实时的语音识别结果反馈	json string
ddsLintener	asr.speech.result	最终的语音识别结果反馈	json string
ddsLintener	dm.output	对话的输出结果	json string
ddsLintener	sys.dm.end	表示结束对话	无
ddsLintener	device.mode.return	表示设置设备状态的回复消息	json string "{"result":"success"}"
ddsLintener	sys.client.error	表示客户端出现异常情况	json string "{"error":"ttsError"}" 目前 error 字段的取值一共有: ttsError, ddsNetworkError, vadSlienceTimeout
ddsLintener	command://xx	在dui平台上配置的command指令	json string
ddsLintener	native://xx	在dui平台上配置的native 指令	json string
ddsLintener	vprint.regist.result	声纹注册的消息回调	json string 形如, {"operation":"start"} 表示声纹注册接口调用成成功，开始注册声纹。 {"operation":"nameRepeat"} 表示注册人姓名重复，{"operation":"vNumLimit"} 表示超出声纹注册上限， {"operation":"unavailable"} 表示所注册音频信噪比不够， {"operation":"continue"} 表示可以继续注册声纹，{"operation":"success"}表示声纹注册成功
ddsLintener	vprint.unregist.result	声纹删除的消息回调	json string 形如 {"operation":"success"} 表示删除成功,{"operation":"noSpeaker"} 表示没有所有删除的注册人信息
ddsLintener	vprint.test.result	声纹计算的消息回调	json string 形如: {"register":"nothing"} 表示当前还没有声纹模型，{"score":23.286682,"word":"qi ke kong tiao","register":"test0","time":179.679932,"speech":0.880000,"RTF":0.204188} 表示收到了正常的计算结果，其中的 register 字段表示所计算的声纹标识。 如果 register 为 others表示没有匹配到具体的声纹
ddsLintener	energy.estimate.result	能量计算的消息回调	json string 形如, {"value":40}

配置选项

```
{
    "sdk": {
        "configPath":"./config.json"
    },
    "auth": {
        "productId": "278569448",
        "deviceProfile": ""
    },
    "front": {
        "aecBinPath": "",
        "wakeupBinPath": "",
```

```
        "beamformingBinPath": "",
        "rollBack": 0
    },
    "vad": {
        "resBinPath": "",
        "pauseTime": 500,
        "slienceTimeout": 5
    },
    "cloud": {
        "productId": "278569448",
        "aliasKey": "prod"
    },
    "recorder": {
         "mode": "internal"
        "samplerate":16000,
        "bits":16,
        "channels":1,
        "device": "default"
    },
    "player": {
        "device": "default"
    },
    "tts": {
        "type": "cloud",
        "zhilingf": {
            "resBinPath":"",
            "dictPath":""
        },
        "voice": "zhilingf",
        "volume": 50,
        "speed": 0.85
    },
    "oneShot": {
        "enable": false
    },
    "abnormal": {
        "netErrorHint":"path:../res/tts/net.mp3",
        "ttsErrorHint":"path:../res/tts/tts_error.mp3"
    },
    "debug": {
        "recAudioDumpFile":"",
        "bfAudioDumpFile":""
    }
}
```

参数	类型	含义	是否必须
sdk	json 对象	客户端的一些配置	必选
sdk.configPath	string	配置文件路径	必选
auth	json 对象	授权信息	必选
auth.productId	string	dui 上创建产品ID	必选
auth.deviceProfile	string	授权信息	必选
front	json 对象	前端信号处理的相关配置	必选
front.aecBinPath	string	aec 算法资源路径	可选
front.wakeupBinPath	string	唤醒算法资源路径	必选
front.beamformingBinPath	string	beamforming 算法资源路径	可选
vad	json 对象	vad 算法模块配置	必选
vad.resBinPath	string	vad算法资源路径	必选
vad.pauseTime	int	vad截止检测时长，单位为 ms，默认为 500ms	可选
vad.slienceTimeout	int	vad 的静音检测超时时长，单位为s，默认为 6s	可选
cloud	json 对象	云端产品的相关配置	必选

cloud.productId	string	dui平台上创建的产品ID	必选
cloud.aliasKey	string	dui平台上创建的产品ID 发布支持， 取值为 "prod | test"	可选，默认为 prod 分支
recorder	json 对象	录音机的相关配置	必选
recorder.mode	string	录音方式，取值为 "internal | external" 分列表示内部录音和外部录音。	可选
recorder.samplerate	int	录音采样率	必选
recorder.bits	int	录音采样位数	必选
recorder.channels	int	录音采用通道数	必选
recorder.device	string	内部录音机的设备名，默认当前系统的 default 音频设备	可选
player	json 对象	内部播放器的设置	可选
player.device	string	内部播放器的设备名，默认为 default	可选
tts	json 对象	合成音的相关配置	必选
tts.type	string	合成音的类型，支持 "cloud" | "local" 分别表示云端合成和本地合成	必选
tts.voice	string	合成音的音色，如果为本地合成， 仅支持 "zhilingf"	必选
tts.zhilingf	json 对象	当 tts.type 为 "local"时，会根据 tts.voice 选择对应音色的合成资源路径。	可选
tts.zhilingf.resBinPath	string	本地合成 zhilingf 的资源路径	可选
tts.zhilingf.dictPath	string	本地合成 zhilingf 的词典路径	可选
tts.volume	int	合成音的音量	可选
tts.speed	int	合成音的速度	可选
oneShot	json 对象	oneshot 模块配置	必选
oneShot.enable	bool	是否启用oneshot，当前仅支持 false	可选
abnormal	json 对象	sdk异常情况下对话配置	可选
abnormal.netErrorHint	string	网络错误下的提示音，需要配置成本地文件，网络不好的情况下云端合成也用不了。	可选
abnormal.ttsErrorHint	string	云端tts合成播放错误情况下的的提示音，需要配置成本地文件。	可选
debug	json 对象	保存音频的配置选项	可选
debug.recAudioDumpFile	string	原始录音保存文件路径	可选
debug.bfAudioDumpFile	string	beamforming算法输出的音频文件路径	可选

唤醒词说明

```
{
           "greetingFile":"path:./res/tts/help.mp3",
            "greeting": "我在，有什么可以帮你",
            "pinyin": "ni hao xiao chi",
            "name": "你好小驰",
            "threshold": 0.127
        }


greetingFile: 唤醒之后播放的提示音，支持本地录音文件，传入文件路径。
greeting: 唤醒之后的提示文本， sdk内部合成。
pinyin: 唤醒词拼音。
name: 唤醒词的中文。
threshold: 唤醒词阈值。
```

```
唤醒提示音播放优先级：如果配置了 greetingFile 则播放 greetingFile ，
否则播放 greeting 。
```

命令唤醒词说明

```
{
            "pinyin": "jiang di yin liang",
            "threshold": 0.100,
            "action": "decrease.volume",
            "name": "降低音量"
        }

pinyin：唤醒词拼音。
name：唤醒词的中文。
threshold：唤醒词阈值。
action：该命令唤醒词对应的动作，比如这个例子中，sdk回调函数会抛出
command://decrease.volume 消息。
```

用户事件说明

用户可以通过dds_client_publish(struct dds_client *ds, int ev, const char* data); 接口给sdk发送事件。

`DDS_CLIENT_USER_EXTERNAL_WAKEUP` 表示外部唤醒事件，用户可以在收到声纹计算结果之后通过这个事件让sdk继续交互。

这个事件传入的数据格式为：

{
“nlg”:”xxxx”
}

可以通过nlg字段传入所有播报的内容，播放完成之后就可以做识别了。 如果传入NULL表示立刻开始识别。