

版本: (master) [https://gitee.com/Jieli-Tech/fw-AC79\\_AIoT\\_SDK](https://gitee.com/Jieli-Tech/fw-AC79_AIoT_SDK)

板级: AC7911 开发板

环境: window+windsurf(vscode)

vscode 终端

- 1. cd tools
- 2. .\make\_prompt.bat
- 3. make

火山大模型权限申请流程

主要是获取 RTC 的 ID、KEY, ASR、TTS 的 ID 和大模型 ID、和账号的 AK 和 SK, 并创建策略, 将 LLM 接入 RTC 中。

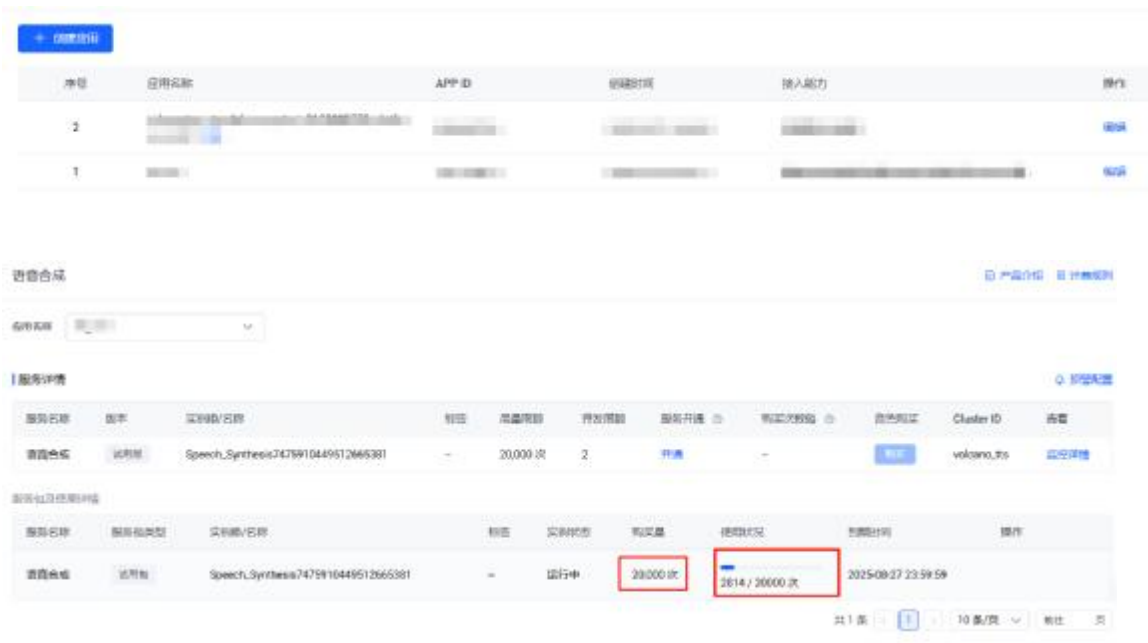
1. [创建火山引擎账号并申请音视频服务](#)



| 应用名称           | AppID                  | AppKey ID | 状态 | 创建时间                | 操作                      |
|----------------|------------------------|-----------|----|---------------------|-------------------------|
| defaultAppName | 62b71810c0a9f1e0a0a57f | *****     | 正常 | 2025-02-11 11:35:33 | 编辑Token Token设置 应用策略 更多 |

创建应用后拿到 RTC\_APP\_ID、RTC\_APP\_KEY 用于生成 TOKEN。

2. [开通语音识别（ASR）和语音合成（TTS）服务](#)



| 序号 | 应用名称               | APP ID                                | 创建时间 | 接入能力 | 操作 |
|----|--------------------|---------------------------------------|------|------|----|
| 2  | Speech_Synthesis   | Speech_Synthesis7475910449512665381   |      |      | 编辑 |
| 1  | Speech_Recognition | Speech_Recognition7475910449512665381 |      |      | 编辑 |

| 服务名称 | 速率  | 实例名称                                | 状态 | 最高限额     | 开发限额 | 服务开通 | 购买次数 | 购买策略 | Cluster ID  | 查看   |
|------|-----|-------------------------------------|----|----------|------|------|------|------|-------------|------|
| 语音合成 | 试用包 | Speech_Synthesis7475910449512665381 | -  | 20,000 次 | 2    | 开通   | -    | 购买   | volcano_tts | 查看详情 |

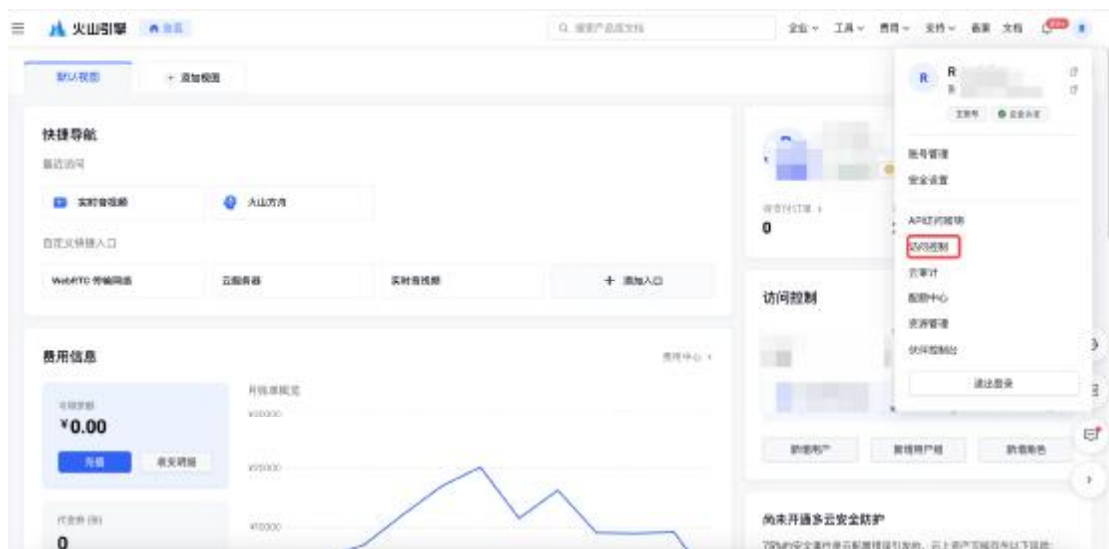
  

| 服务名称 | 服务包类型 | 实例名称                                | 状态 | 实例状态 | 购买量      | 使用状况           | 到期时间                | 操作 |
|------|-------|-------------------------------------|----|------|----------|----------------|---------------------|----|
| 语音合成 | 试用包   | Speech_Synthesis7475910449512665381 | -  | 运行中  | 20,000 次 | 2614 / 20000 次 | 2025-08-27 23:59:59 |    |

共 1 页 1 / 10 页/页 默认 页



登录主账号 [RTC 控制台](#)。在右上角导航栏单击访问控制。



在左侧导航栏选择角色并新建角色。



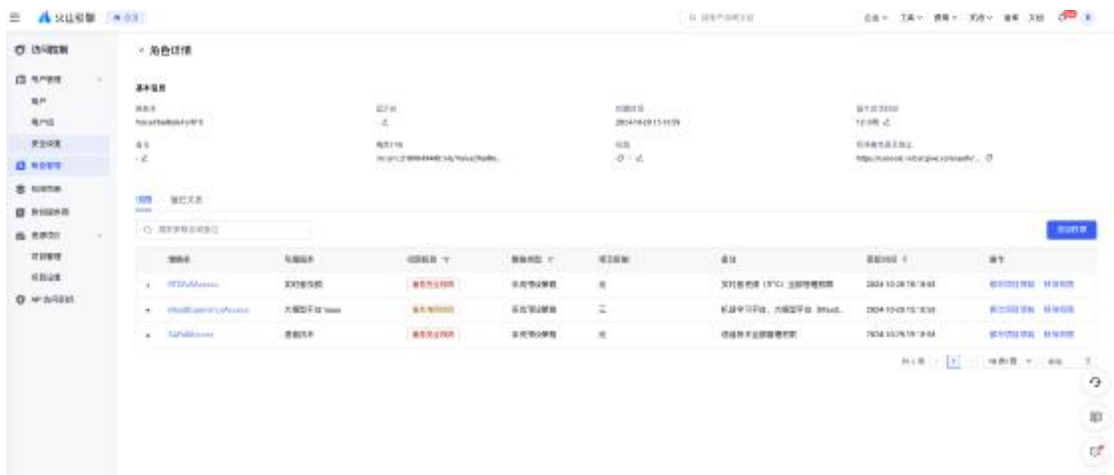
选择服务，并填入实时音视频。



填入角色名。此处角色名固定取值：VoiceChatRoleForRTC。



添加 SAFullAccess 、 MaaSExperienceAccess 和 RTCFullAccess 权限，且权限作用范围选择全局。



5. 获取 AK、SK，用于生成 HTTPS 请求时的签名



上述流程获取完整的各种权限。

具体使用

wifi\_story\_machine 工程中的 LLM/VOLC 文件夹为实现程序，其中的 net\_request.c 用于请求智能体开启停止和更新。access\_token.c 用于生成加入 RTC 房间所需的 token。

```
Makefile > ^app_main.c > ^VolcEngineRTCLiteDemo.c ^net_request.c ^access_token.c ...>
1 #include <string.h>
2 #include <time.h>
3 #include <mbedtls/md.h>
4 #include <mbedtls/base64.h>
5 #include <curl/curl.h>
6 #include <cJSON.h>
7
8 #define RTC_API_HOST "rtc.volcengineapi.com"
9 #define AK
10 #define SK
11 #define ASR_APP_ID
12 #define TTS_APP_ID
13 #define LLM_APP_ID
14 #define RTC_APP_ID
15 #define RTC_APP_KEY
16
```

将之前获取到的各个参数填入。

调用 volcengineRtcliteDemo.c 中的 VolcEngineRTCDemo 函数，初始化 RTC，成功会打印出 roomid、uid、token，并且播放初始欢迎音"你好"，这个 demo 不断循环获取音频数据并上传，开启对话。火山默认支持 opus 其他音频输入需要联系平台加白，并且房间号以音频格式开头。

后续 roomid 和 userid 可以自由设置。

```
int start_voice_chat(rtc_room_info_t *room_info)
{
    char *json_body = start_voice_chat_json(RTC_APP_ID, "G711A123456", "user123465");
    char query_string[50];
}
```

对于大模型的配置在这个 json 中。[参考官方文档进行配置](#)。

```
char *start_voice_chat_json(const char *app_id, const char *room_id, const char *user_id)
{
    cJSON *body = cJSON_CreateObject();
    cJSON_AddStringToObject(body, "AppId", app_id);
    cJSON_AddStringToObject(body, "RoomId", room_id);
    cJSON_AddStringToObject(body, "UserId", user_id);
    cJSON_AddStringToObject(body, "TaskId", "task1");

    cJSON *config = cJSON_CreateObject();
    cJSON_AddNumberToObject(config, "InterruptMode", 1);
    cJSON *asrConfig = cJSON_CreateObject();
    cJSON_AddStringToObject(asrConfig, "Provider", "volcano");
    cJSON *ProviderParams = cJSON_CreateObject();
    cJSON_AddStringToObject(ProviderParams, "Mode", "smallmodel");
    cJSON_AddStringToObject(ProviderParams, "AppId", ASR_APP_ID);
    cJSON_AddStringToObject(ProviderParams, "Cluster", "volcengine_streaming_common");
    cJSON_AddItemToObject(asrConfig, "ProviderParams", ProviderParams);
    cJSON *VADConfig = cJSON_CreateObject();
    cJSON_AddNumberToObject(VADConfig, "SilenceTime", 500);
    cJSON_AddItemToObject(asrConfig, "VADConfig", VADConfig);
}
```

在 app\_config.h 中打开大模型的宏后，配网成功后会在 wifi\_app\_task.c 中网络连接成功回到中，创建大模型线程。

```
7 // #define CONFIG_VOLC_LLM_ENABLE //火山大模型
8
```

```

#ifdef CONFIG_VOLC_LLM_ENABLE
    int VolcEngineRTCDemo();
    thread_fork("Volc_demo", 4, 3 * 1024, 0, 0, VolcEngineRTCDemo, NULL);
#endif

```

如有需求可以在 app\_config.h 中打开硬件回采进行回声消除。

```

#ifdef CONFIG_AEC_ENC_ENABLE
#define CONFIG_USB_AUDIO_AEC_ENABLE //usb mic使能回声消除功能
// #define CONFIG_AEC_LINEIN_CHANNEL_ENABLE //AEC回采使用硬件通道数据
// #define CONFIG_AEC_USE_PLAY_MUSIC_ENABLE //播歌时需要使用AEC
#endif

```