

BRTC实时交互智能体接入指引文档V1.4

目录

- 1. 申请智能体AppID
- 2. 智能体服务端API 接入指引
 - 2.1 LLM 设置
 - 2.2 TTS 设置
- 3. 客户端接入指引
 - 3.1. 微信小程序接入指引
 - 3.2. 安卓手机接入指引
 - 3.3. 苹果手机接入指引
 - 3.4. H5网页接入指引
 - 3.5. rtos sdk接入指引
 - 3.6. websocket API接入指引
- 4. 接入曦灵数字人的说明

1. 申请智能体AppID

需要开通百度云账号，在百度云控制台上申请智能体AppID

目前在实时音视频RTC 产品中申请AppID

地址：<https://cloud.baidu.com/doc/RTC/index.html>

获得AppID 后开始进入第二步，BRTC 采用预付费方式，需要对BRTC进行充值后使用。

2. 智能体服务端API 接入指引

接入文档：**AI实时互动服务管理对外API.pdf**

[https://brtc-](https://brtc-sdk.cdn.bcebos.com/ai/agent/api/AI%E5%AE%9E%E6%97%B6%E4%BA%92%E5%8A%A8%E6%9C%8D%E5%8A%A1%E7%AE%A1%E7%90%86%E5%AF%B9%E5%A4%96API.pdf)

[sdk.cdn.bcebos.com/ai/agent/api/AI%E5%AE%9E%E6%97%B6%E4%BA%92%E5%8A%A8%E6%9C%8D%E5%8A%A1%E7%AE%A1%E7%90%86%E5%AF%B9%E5%A4%96API.pdf](https://brtc-sdk.cdn.bcebos.com/ai/agent/api/AI%E5%AE%9E%E6%97%B6%E4%BA%92%E5%8A%A8%E6%9C%8D%E5%8A%A1%E7%AE%A1%E7%90%86%E5%AF%B9%E5%A4%96API.pdf)

服务器地址：<https://rtc-aiagent.baidubce.com>

API config中的模型选择： config: {llm, llm_url},

API 简要介绍：

- **1. 创建智能体：/api/v1/aiagent/generateAIAgentCall**
 - 参数格式：{app_id: AppID, quick_start: true, config: "{}"};
- **2. 停止智能体：/api/v1/aiagent/stopAIAgentInstance**
 - 参数格式：{ app_id: AppID, ai_agent_instance_id: AgentID};

- **3. 打断功能:** `/api/v1/aiagent/interrupt`
 - 参数格式: `{ app_id: AppID, ai_agent_instance_id: AgentID};`

2.1 LLM 设置

LLM 支持内置和第三方大模型，支持百度千帆智能体、阿里百炼应用、腾讯混元智能体、扣子等。

默认设置

llm、llm_url不传则使用服务端默认LLM配置

接入第三方LLM服务配置：

</> LLM 和智能体配置例子

[Go](#)

```
1 一. 使用兼容OPENAI 协议的接入:
2 {"llm": "OPENAI", "llm_url": "", "llm_cfg": "$MODEL", "llm_token": "$API-KEY"}
3
4 llm_url填入模型服务地址;llm_cfg填入模型名称, llm_token填入API-KEY
5 1.) 百度千帆模型服务地址: https://qianfan.baidubce.com/v2/chat/completions
6 2.) 阿里百炼模型服务地址: https://dashscope.aliyuncs.com/compatible-mode/v1/chat/completions
7 3.) 腾讯混元模型服务地址: https://api.hunyuan.cloud.tencent.com/v1/chat/completions
8 4.) 豆包模型服务地址: https://ark.cn-beijing.volces.com/api/v3/chat/completions
9 5.) 豆包自定义应用地址: https://ark.cn-beijing.volces.com/api/v3/bots/chat/completions
10 6.) DeepSeek模型服务地址: https://api.deepseek.com/chat/completions
11
12 二. 百度AppBuilder接入:
13 {"llm": "LLMAppBuilder", "llm_cfg": "app_id[,conversation_id]", "llm_token": "$API-KEY"}
14
15 三. 阿里百炼应用接入:
16 {"llm": "LLMDashScopeApp", "llm_url": "https://dashscope.aliyuncs.com/api/v1/apps/$YOUR_APPID/c
17   ompletion"
18   , "llm_cfg": "[session_id]", "llm_token": "$API-KEY"}
19
20 四. 腾讯元宝智能体接入:
21 {"llm": "LLMYuanbao", "llm_cfg": "assistant_id[,user_id]", "llm_token": "$API-KEY"}
22
23 五. 扣子接入:
24 {"llm": "LLMCoze", "llm_cfg": "bot_id[,user_id][,conversation_id]", "llm_token": "$API-KEY"}
25
26 六. Dify接入:
27 {"llm": "LLMDify", "llm_url": "https://api.dify.ai/v1/chat-messages", "llm_cfg": "[user],
28   [conversation_id]", "llm_token": "$API-KEY"}
```

2.2 TTS 设置

tts支持内置和第三方，主要包含火山大模型tts、讯飞基础tts、百度大模型tts等。此处可以设置使用内置账号资源和用户自有账号资源，通过tts类型区分，同时tts_url参数需要根据不同的tts类型进行设置

默认设置

tts、tts_url不传则使用服务端默认tts配置

tts类型设置

</> tts类型Java

```
1      "DEFAULT", "VOLC", "XUNFEI", "BAIDU"
```

tts_url设置

tts_url中支持对tts常用参数的设置，如发音人、音量、音高、语速、情感（如果有）、语言等，具体格式如下：

</> 服务默认tts下修改音色JSON

```
1  {"tts_url": "DEFAULT{\"vcn\": \"111\"}"}
```

tts常用参数配置说明如下：

1	参数名	参数类型	参数描述
2	vcn	String	发音人
3	vol	Double	音量
4	spd	Double	语速
5	pit	Double	音高（如果有）
6	emotion	String	情感（如果有）
7	lang	String	语言
8	apid	String	tts应用id（使用第三方tts）
9	apikey	String	tts鉴权key、token等（使用第三方tts）
10	apisk	String	tts鉴权密钥（使用第三方tts）

配置示例

</> config配置TTS 示例JSON

```
1 使用内置tts账号配置：
2 {"tts": "DEFAULT", "tts_url": "DEFAULT{\"vcn\": \"111\", \"vol\": 2.0, \"spd\": 1.0}"}
```

```
3
4 使用客户自有百度大模型tts账号配置：
```

```
5 {"tts":"BAIDU", "tts_url": "BAIDU{\"vcn\": \"111\", \"vol\": 2.0, \"spd\":  
1.0, \"apikey\": \"token\"}"}
```

6

7 使用客户自有火山大模型tts账号配置:

```
8 {"tts":"VOLC", "tts_url": "VOLC{\"vcn\": \"111\", \"vol\": 2.0, \"spd\":  
1.0, \"apid\": \"xxx\", \"apikey\": \"token\"}"}
```

9

10 使用客户自有讯飞tts账号配置:

```
11 {"tts":"XUNFEI", "tts_url": "XUNFEI{\"vcn\": \"111\", \"vol\": 2.0, \"spd\":  
1.0, \"apid\": \"xxx\", \"apikey\": \"xxx\", \"apisk\": \"xxx\"}"}
```

12

13 接口中config完整配置例子:

```
14 {  
15     "config" : "{\"llm\" : \"LLMRacing\", \"llm_token\" : \"no\", \"lang\" : \"zh\",  
    \"tts\": \"DEFAULT\", \"tts_url\": \"DEFAULT{\"vcn\": \"nvsheng\", \"vol\":  
    0.5, \"spd\": 1.1, \"apid\": \"\", \"apikey\": \"\"}\"}"
```

16 }

3. 客户端接入指引

3.1. 微信小程序接入指引

微信小程序接入SDK: <https://brtc->

[sdk.cdn.bcebos.com/ai/agent/miniapp/BRTC.Agent.MiniApp.SDK.V1.0.5.zip](https://brtc-sdk.cdn.bcebos.com/ai/agent/miniapp/BRTC.Agent.MiniApp.SDK.V1.0.5.zip)

文档详见SDK压缩包中。

</> 微信小程序API简要介绍:

JavaScript

```
1 1. 初始化  
2 const brtc_agent = require('./agent.js');  
3 var mAgent = brtc_agent.Agent;  
4  
5 2. 开始通话  
6 // 启动Agent  
7 var cfg = {  
8     llm: 'LLMxxx',  
9     llm_url: ''  
10 };  
11 mAgent.Start({  
12     appid: 'AppIdxxx',  
13     cfg: cfg,  
14     success: function (pushurl, agentId) {
```

```
15     },
16     error: function (error) {
17     },
18     onmessage: function (msg) {
19         console.log('onmessage id: ' + msg.id + ' data: ' + msg.data);
20     },
21     remotevideocoming: function (id, display, attribute, pullurl) {
22     },
23 });
24
25 3. 发送文本消息
26 mAgent.sendMessageToUser('[T]:你好');
27
28 4. 结束通话
29 mAgent.Stop();
```

3.2. 安卓手机接入指引

安卓SDK: <https://brtc-sdk.cdn.bcebos.com/ai/agent/android/Android.ChatAgent.v1.0.25.zip>

文档详见SDK压缩包中。

</> 安卓SDK函数简介

Java

```
1 1. 初始化初始化AIAgentEngine
2     public static AIAgentEnginImpl init(Context context, AIAgentEngineParams params)
3 2. 开启对话
4     public abstract void call(String token, long instanceId);
5 3. 结束对话
6     public abstract void hangup();
7 4. 发送文本
8     public abstract void setTextToAIAgent(String text);
```

3.3. 苹果手机接入指引

iOS SDK: https://brtc-sdk.cdn.bcebos.com/ai/agent/ios/BaiduChatAgent.IOS_V1.2.2.zip

文档详见SDK压缩包中。

</> iOS SDK 函数简介

Object-C

```
1 1. 初始化BaiduChatAgent
2     self.baiduChatAgent = [[BaiduChatAgent alloc] initWithParams:self.baiduChatAgentParms
3     delegate:self];
3 2. 开启对话
4     - (void)call:(NSString *)token instanceId:(NSInteger)instanceId;
```

```
5 3. 结束对话
6     - (void)hangup;
7 4. 设置回调
8     - (id)initWithParams:(AgentEngineParams*) params delegate:
      (id<BaiduChatAgentDelegate>)delegate;
9 5. 发送文本
10    - (void)sendTextToAIAgent:(NSString *)text;
```

3.4. H5网页接入指引

H5 SDK: <https://brtc-sdk.cdn.bcebos.com/ai/agent/h5/BRTC.Agent.H5.SDK.V1.0.4.zip>

H5在线demo: https://brtc-sdk.cdn.bcebos.com/ai/agent/h5/brtc_agent.html

文档详见SDK压缩包中。

</> H5 SDK 函数简介

JavaScript

```
1 1. 初始化
2 var Agent = new BaiduRtcAgentClient();
3
4 2. 开始通话
5 var cfg = {
6     llm: 'LLMxxx', // 私有LLM 可以填入'LLMOpenAPI', llm_url传入私有化地址
7     llm_url: ''
8 };
9 Agent.Start({
10     appid: 'AppIdxxx',
11     cfg: cfg,
12     remotevideoviewid: 'therevideo',
13     localvideoviewid: 'herevideo',
14     success: function () {
15     },
16     error: function (error) {
17     },
18     onmessage: function (msg) {
19         console.log('onmessage id: ' + msg.id + ' data: ' + msg.data);
20     }
21
22 3. 发送文本消息
23 Agent.sendMessageToUser('[T]:你好');
24
25 4. 结束通话
26 Agent.Stop();
```

3.5. rtos sdk接入指引

1	芯片平台	SDK下载地址
2	支持乐鑫、杰里、BK、泰鑫、移远等平台	https://cloud.baidu.com/doc/RTC/s/wm8y592lc

文档详见SDK压缩包中。

</> rtos sdk函数简介

Java

```
1 1. 初始化初始化AIEngine
2     BaiduChatAgentEngine* baidu_create_chat_agent_engine(const BaiduChatAgentEvent* events);
3     int baidu_chat_agent_engine_init(BaiduChatAgentEngine* engine, const AgentEngineParams*
    param);
4 2. 开启对话
5     void baidu_chat_agent_engine_call(BaiduChatAgentEngine* engine);
6 3. 发送文本
7     void baidu_chat_agent_engine_send_text(BaiduChatAgentEngine* engine, const char* text);
8 4. 结束对话
9     void baidu_chat_agent_engine_destroy(BaiduChatAgentEngine* engine);
```

</> 服务端创建智能体请求

Plain Text

```
1 #####注意事项1#####
2 #ifdef BRTC_ENABLE_G722
3 #define JSON_CONFIG_TEMPLATE "{\"app_id\": \"%s\", \"config\": \"{\\\"llm\\\" : \\\"%s\\\",
    \\\"llm_token\\\" : \\\"no\\\", \\\"rtc_ac\\\": \\\"g722\\\", \\\"lang\\\" : \\\"%s\\\"}\"\",
    \"quick_start\": true}"
4 #else
5 #define JSON_CONFIG_TEMPLATE "{\"app_id\": \"%s\", \"config\": \"{\\\"llm\\\" : \\\"%s\\\",
    \\\"llm_token\\\" : \\\"no\\\", \\\"rtc_ac\\\": \\\"pcmu\\\", \\\"lang\\\" : \\\"%s\\\"}\"\",
    \"quick_start\": true}"
6 #endif
7 //目前rtos 支持pcmu和g722格式，需要按照上面格式将音频格式进行http请求发送。
8 #####
9
10
11 #####注意事项2#####
12 const char* json_str =
13 "{"
14     "\"ai_agent_instance_id\": 2230595646193664,\"
15     "\"context\": {"
```

```

16         "\"cid\": 1,"
17         "\"token\":
    \"00415f2bd1c3fe5dbc02cc49af726b18d6c6bfc56dc174238426178be806a1742470661\"
18     }"
19 "}; //这是服务端返回的参数，需要将参数给客户端
20
21 #####注意事项3#####
22 //客户在集成rtos sdk时需要传入设备鉴权的params->license_key和params->userId
23 //如果使用了芯片平台的自带http发送请求，可以按照下面的setUserParameters方式来实现，例如：杰里
    芯片平台可参考下面实现
24 void setUserParameters(AgentEngineParams *params) {
25     snprintf(params->agent_platform_url, sizeof(params->agent_platform_url), "%s",
    SERVER_HOST_ONLINE); //SERVER_HOST_ONLINE客户搭建服务的IP或者域名
26     snprintf(params->appid, sizeof(params->appid), "%s", BDCloudDefaultRTCApplID); //需要和服
    务端使用同一个applID
27     snprintf(params->userId, sizeof(params->userId), "%s", "12345678"); //终端用户唯一的id
    号，例如手机号
28     snprintf(params->cer, sizeof(params->cer), "%s", "./a.cer");
29     snprintf(params->workflow, sizeof(params->workflow), "%s", "VoiceChat");
30     snprintf(params->license_key, sizeof(params->license_key), "%s", "xxxx"); //"xxxx"为
    license_key字符串，需要购买获得
31     snprintf(params->remote_params, sizeof(params->remote_params), "%s", json_str);
32
33     params->verbose = false;
34     params->enable_local_agent = false;
35     params->enable_voice_interrupt = true;
36     params->level_voice_interrupt = 80;
37     strncpy(params->llm, "LLMRacing", sizeof(params->llm) - 1);
38     strncpy(params->lang, "zh", sizeof(params->lang) - 1);
39     params->AudioInChannel = 1;
40     params->AudioInFrequency = 16000;
41 }

```

</> 客户端自己创建智能体

Plain Text

```

1 //如果使用了rtos sdk内部http发送请求 可以按照下面的setUserParameters方式来实现，例如：乐鑫芯
    片平台可以参考下面方式实现
2 void setUserParameters(AgentEngineParams *params) {
3     snprintf(params->agent_platform_url, sizeof(params->agent_platform_url), "%s",
    SERVER_HOST_ONLINE); //SERVER_HOST_ONLINE客户搭建服务的IP或者域名
4     snprintf(params->appid, sizeof(params->appid), "%s", BDCloudDefaultRTCApplID); //需要和服
    务端使用同一个applID

```

```
5    snprintf(params->userId, sizeof(params->userId), "%s", "12345678"); //终端用户唯一的id
   号，例如手机号
6    snprintf(params->cer, sizeof(params->cer), "%s", "./a.cer");
7    snprintf(params->workflow, sizeof(params->workflow), "%s", "VoiceChat");
8    snprintf(params->license_key, sizeof(params->license_key), "%s", "xxxx"); // "xxxx"为
   license_key字符串，需要购买获得
9
10   params->instance_id = 10373;
11   params->verbose = false;
12   params->enable_local_agent = true;
13   params->enable_voice_interrupt = true;
14   params->level_voice_interrupt = 80;
15   strncpy(params->llm, "LLMRacing", sizeof(params->llm) - 1);
16   strncpy(params->lang, "zh", sizeof(params->lang) - 1);
17   params->AudioInChannel = 1;
18   params->AudioInFrequency = 16000;
19 }
```

</> 客户搭建自己的服务

Plain Text

```
1 https://cloud.baidu.com/doc/RTC/s/wm8y5921c 下载服务端Demo示例代码userserver-and-web-demo.zip
2 服务端示例代码可以运行在本地或者服务器上（windows/mac/linux），只需要安装jdk17即可
3 linux/mac环境下执行步骤：
4 BCE_APP_AK=xxxx\
5 BCE_APP_SK=xxxx\
6 java -jar rtc-aiagent-userserver-1.0-SNAPSHOT.jar\
7
8 windows环境下执行步骤：
9 set BCE_APP_AK=xxxx
10 set BCE_APP_SK=xxxx
11 java -jar rtc-aiagent-userserver-1.0-SNAPSHOT.jar
12
13 #####注意事项1#####
14 服务端默认服务地址: "https://rtc-aiagent.baidubce.com/api/v1/aiagent"
15 海外服务端服务地址: "https://rtc-aiagent-out.baidubce.com/api/v1/aiagent"
```

3.6. websocket API接入指引

协议文档: <https://brtc-sdk.cdn.bcebos.com/ai/agent/ws/doc/WebsocketAPI.pdf>

线上demo: <https://brtc-sdk.bj.bcebos.com/ai/agent/ws/realtime.html>

ws SDK (javascript) : <https://brtc-sdk.cdn.bcebos.com/ai/agent/ws/h5/BRTC.Agent.WS.H5.SDK.V1.0.1.zip>

ws SDK (Go) : <https://brtc-sdk.cdn.bcebos.com/ai/agent/ws/go/BRTC.Agent.WS.Go.SDK.V1.0.0.zip>

</> Go demo简介

Go

```
1 # BRTC Agent WS Go client demo
2 需要先到百度智能云申请BRTC的AppID, https://cloud.baidu.com/doc/RTC/index.html
3
4 ## 1. 编译并测试
5 ```bash
6 # pcmu demo
7 go build client_pcmu.go
8 ./client_pcmu -a $APPID_GET_FROM_BAIDU
9 ```
10 ## 2. Go 使用WebSocket API 协议详解
11 ### 2.1 建立wss链接
12 ```go
13 var uri = "wss://rtc-aiotgw.exp.bcelive.com/v1/realtime?" + "a=" + appid + "&id=" + id +
14   "&t=" + token + "&ac=pcmu"
15 websocket.DefaultDialer.Dial(uri, nil)
16
17 ### 2.2 读取消息, 根据消息类型进行处理, Text消息进行显示, Binary消息就是pcmu数据, 可以解码播放
18 ```go
19 func func() {
20     defer close(done)
21     for {
22         mt, message, err := c.ReadMessage()
23         if err != nil {
24             log.Println("read:", err)
25             return
26         }
27         if mt == websocket.TextMessage {
28             log.Printf("rx: %s", message)
29         } else if mt == websocket.BinaryMessage {
30             my.OnAudioData(message)
31         }
32     }
33 }()
34 ```
35 ### 2.3 发送pcmu数据, 发送20ms的pcmu数据给互动智能体
36 ```go
```

```
37     ticker := time.NewTicker(time.Millisecond * 20)
38     for ; true; <-ticker.C {
39         c.WriteMessage(websocket.BinaryMessage, payload[Pos:Pos + 160])
40         Pos += 160;
41     }
42 ```
43 ### 2.4 发送Text文本消息直接进行播放，用于开场白等场景
44 ```go
45     c.WriteMessage(websocket.TextMessage, []byte(`[TTS]:你是谁?`))
46 ```
```

4. 接入曦灵数字人的说明

在曦灵管理平台创建数字人应用， 购买云渲染交互组件。

<https://xiling.cloud.baidu.com/open/widgetStore/list>

获得数字人的AppID 和AppKey后使用Token 生成工具生成Token（根据业务需要选择token有效期，建议为900000小时）：

<https://open.xiling.baidu.com/token-gen-tool.html>

把Token填入xiling_auth字段

</> demo 配置实例

Bash

```
1 https://brtc-sdk.bj.bcebos.com/ai/agent/h5/brtc_agent.html?
   a=XXXX&it=DigitalHumanH&tts=DHV2&xiling_auth=YY
```

</> 代码中配置曦灵数字人Token

JavaScript

```
1 var cfg = {
2     xiling_auth: 'xiling-token-XXX',
3     xiling_bgimg: '',
4     fid: 'A2A_V2-muqing', // 数字人形象， 可选形象
   https://cloud.baidu.com/doc/AI_DH/s/2lyzilgsg
5     tts_url: 's?per=5132', // 数字人音色， 可选音色
   https://cloud.baidu.com/doc/AI_DH/s/Slywt3fxy
6     tts: 'DHV2', // 固定值，指定数字人服务类别为DHV2
7 };
8 Agent.Start({
9     instance_type: 'DigitalHuman', // 固定值，指明是数字人智能体
```

```
10  appid: 'AppIdxxx',
11  cfg: cfg,
12  remotevideoviewid: 'therevideo',
13  localvideoviewid: 'herevideo',
14  success: function () {
15  },
16  error: function (error) {
17  },
18  onmessage: function (msg) {
19      console.log('onmessage id: ' + msg.id + ' data: ' + msg.data);
20  }
21
```