AKOOL

Streaming Avatar

Streaming Avatar API Overview

Comprehensive guide to the Streaming Avatar API

Both the avatar_id and voice_id can be easily obtained by copying them directly from the web interface. You can also create and manage your streaming avatars using our intuitive web platform.

Create and manage your avatars at: **https://akool.com/apps/upload/avatar?from=%2Fapps%2Fstreaming-avatar%2Fedit**

The resources (image, video, voice) generated by our API are valid for 7 days. Please save the relevant resources as soon as possible to prevent expiration.

To experience our live avatar streaming feature in action, explore our demo built on the Agora streaming service: **AKool Streaming Avatar React Demo**.

Knowledge Base Integration: You can enhance your streaming avatar with contextual AI responses by integrating a **Knowledge Base**. When creating a session, provide a `knowledge_id` parameter to enable the AI to use documents and URLs from your knowledge base for more accurate and relevant responses.

API Endpoints

Avatar Management

Upload Streaming Avatar - Create a new streaming avatar from a video URL

Get Avatar List - Retrieve a list of all streaming avatars

Get Avatar Detail - Get detailed information about a specific avatar

Session Management

Create Session - Create a new streaming avatar session

Get Session Detail - Retrieve detailed information about a specific session

Close Session - Close an active streaming avatar session

Get Session List - Retrieve a list of all streaming avatar sessions

Live Avatar Stream Message

```
IAgoraRTCClient.on(event: "stream-message", listener: (uid: UID, pld: Ui

IAgoraRTCClient.sendStreamMessage(msg: Uint8Array | string, flag: boolea
```

Send Data

Chat Type Parameters

Parameter	Type	Required	Value	Description
v	Number	Yes	2	Version of the message
type	String	Yes	chat	Message type chat interactio
mid	String	Yes		Unique messa identifier for

Parameter	Type	Required	Value	Description
				conversation tracking
idx	Number	Yes		Sequential inde of the message start from 0
fin	Boolean	Yes		Indicates if this the final part o the message
pld	Object	Yes		Container for message payl
pld.text	String	Yes		Text content to send to avatar (e.g. “Hello”)

Command Type Parameters

Parameter	Type	Required	Value	Description
v	Number	Yes	2	Protocol versio number
type	String	Yes	command	Specifies this i system comma message
mid	String	Yes		Unique ID to tr correlate comm messages
pld	Object	Yes		Contains the command deta parameters
pld.cmd	String	Yes		Command acti execute. Valid **“set-params”** avatar settings

Parameter	Type	Required	Value	Description
				“int..rupt ,st current avatar response)
pld.data	Object	No		Parameters for command (req for **“set-param**
pld.data.vid	String	No		**Deprecated. U pld.data.vpara instead.** Voice change avatar' Only used with **params”**. Get v from **Voice Lis**
pld.data.vurl	String	No		**Deprecated. U pld.data.vpara instead.** Custo model URL. Or with **“set-para** Get valid URLs **Voice List API**
pld.data.lang	String	No		Language cod avatar respons “en”, “es”). Onl with **“set-para** Get valid code **Language List**
pld.data.mode	Number	No		Avatar interact style. Only use **“set-params”**. Retelling (avata repeats conter = Dialogue (av engages in conversation)
pld.data.bgurl	String	No		URL of backgr image/video fc

Parameter	Type	Required	Value	Description
				scen. On.y us **“set-params”**
pld.data.vparams	Object	No		Voice paramet use for the ses

Voice Parameters

Parameter	Type	Required	Value	Description
vid	String	No		Voice ID to cha avatar’s voice. used with **“set params”**. Get IDs from **Voice API**
vurl	String	No		Custom voice URL. Only use **“set-params”**. valid URLs fro **Voice List API**
speed	double	No	1	Controls the sp of the generate speech. Value range from 0.8 1.2, with 1.0 be the default spe
pron_map	Object	No		Pronunciation mapping for cu words. Exampl “pron_map”: { “akool” : “ai ku
stt_type	String	No		Speech-to-tex `"openai_real` = OpenAI Real

Parameter	Type	Required	Value	Description
turn_detection	Object	No		Turn detection configuration.

Turn Detection Configuration

Parameter	Type	Required	Value	Description
type	String	No	"server_vad"	Turn detectio type. `"server_vad` Server VAD, `"semantic_v` = Semantic V
threshold	Number	No	0.5	Activation threshold (0 t A higher thre will require lo audio to activ the model, ar thus might perform bette noisy environments Available whe type is `"server_vad`
prefix_padding_ms	Number	No	300	Amount of au (in millisecon include befor VAD detectec speech. Avail when type is `"server_vad`
silence_duration_ms	Number	No	500	Duration of si (in millisecon detect speec

Parameter	Type	Required	Value	Description
				stop. Wit.. sh values turns be detected r quickly. Avail when type is "server_vad

JSON Example

Chat Request | Set Avatar Params | Interrupt Response | Set Avatar Actions

```
{
    "v": 2,
    "type": "chat",
    "mid": "msg-1723629433573",
    "idx": 0,
    "fin": true,
    "pld": {
      "text": "Hello"
    },
}
```

Receive Data

Chat Type Parameters

Parameter	Type	Value	Description
v	Number	2	Version of the message
type	String	chat	Message type for chat interactions
mid	String		Unique message identifier for tracking conversation flow

Parameter	Type	Value	Description
idx	Number		Sequential index of the message part
fin	Boolean		Indicates if this is the final part of the response
pld	Object		Container for message payload
pld.from	String	”bot” or “user”	Source of the message - “bot” for avatar responses, “user” for speech recognition input
pld.text	String		Text content of the message

Command Type Parameters

Parameter	Type	Value	Description
v	Number	2	Version of the message
type	String	command	Message type for system commands
mid	String		Unique identifier for tracking related messages in a conversation
pld	Object		Container for command payload
pld.cmd	String	”set-params”, “interrupt”	Command to execute: **“set-params”** to update avatar settings, **“interrupt”** to stop current response
pld.code	Number	1000	Response code from the server, 1000 indicates success
pld.msg	String		Response message from the server

JSON Example

Chat Response Command Response


```
{
    "v": 2,
    "type": "chat",
    "mid": "msg-1723629433573",
    "idx": 0,
    "fin": true,
    "pld": {
      "from": "bot",
      "text": "Hello!  How can I assist you today? "
    }
}
```


Typescript Example

Create Client Send Message Set Avatar Params Interrupt Response

```
const client: IAgoraRTCClient = AgoraRTC.createClient({
  mode: 'rtc',
  codec: 'vp8',
});

client.join(agora_app_id, agora_channel, agora_token, agora_uid);

client.on('stream-message', (message: Uint8Array | string) => {
  console.log('received: %s', message);
});
```

Integrating Your Own LLM Service

Before dispatching a message to the WebSocket, consider executing an HTTP request to your LLM service.

TypeScript

```typescript
const client: IAgoraRTCClient = AgoraRTC.createClient({
  mode: 'rtc',
  codec: 'vp8',
});

client.join(agora_app_id, agora_channel, agora_token, agora_uid);

client.on('stream-message', (message: Uint8Array | string) => {
  console.log('received: %s', message);
});

let inputMessage = 'hello';

try {
  const response = await fetch('https://your-backend-host/api/llm/answe
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify({
      question: inputMessage,
    }),
  });

  if (response.ok) {
    const result = await response.json();
    inputMessage = result.answer;
  } else {
    console.error("Failed to fetch from backend", response.statusText);
  }
} catch (error) {
  console.error("Error during fetch operation", error);
}
```

```
const message = {
  v: 2,
  type: "chat",
  mid: "msg-1723629433573",
  idx: 0,
  fin: true,
  pld: {
      text: inputMessage,
  },
};
client.sendStreamMessage(JSON.stringify(message), false);
```

Response Code Description

> Please note that if the value of the response code is not equal to 1000, the request is failed or wrong

Parameter	Value	Description
code	1000	Success
code	1003	Parameter error or Parameter can not beempty
code	1008	The content you get does not exist
code	1009	Youdo not have permission to operate
code	1101	Invalid authorization or Therequest token has expired
code	1102	Authorization cannot be empty
code	1200	The account has been banned
code	1201	create audio error, pleasetry again later

Parameter	Value	Description
code	1202	The same video cannot be translated lipSync inthe same language more than 1 times
code	1203	video should be with audio
code	1204	Your video duration is exceed 60s!
code	1205	Create videoerror, please try again later
code	1207	The video you are using exceeds thesize limit allowed by the system by 300M
code	1209	Please upload a videoin another encoding format
code	1210	The video you are using exceeds thevalue allowed by the system by 60fp
code	1211	Create lipsync error, pleasetry again later

‹ ErrorCode

Upload Streaming Avatar ›