

Realtime

Communicate with a multimodal model in real time over low latency interfaces like WebRTC, WebSocket, and SIP. Natively supports speech-to-speech as well as text, image, and audio inputs and outputs.

[Learn more about the Realtime API.](#)

Session tokens

REST API endpoint to generate ephemeral session tokens for use in client-side applications.

Create realtime session

POST `https://api.openai.com/v1/realtime/client_secret`

Create a Realtime session and client secret for either realtime or transcription.

Request body

Example request

curl 

```
1 curl -X POST https://api.openai.com/v1/realtime
2   -H "Authorization: Bearer $OPENAI_API_KEY"
3   -H "Content-Type: application/json" \
4   -d '{
5     "expires_after": { "anchor": "created_at",
6     "session": {
7       "type": "realtime",
```

expires_after object Optional

Configuration for the ephemeral token expiration.

▼ Show properties

session object Optional

Session configuration to use for the client secret. Choose either a realtime session or a transcription session.

▼ Show possible types

Returns

The created client secret and the effective session object

```
8     "model": "gpt-4o-realtime",
9     "instructions": "You are a friendly
10  }
11  }'
```

Response



```
1  {
2    "value": "ek_68af296e8e408191a1120ab6383
3    "expires_at": 1756310470,
4    "session": {
5      "type": "realtime",
6      "object": "realtime.session",
7      "id": "sess_C9CiUVUzUzYIss3ELY1d",
8      "model": "gpt-4o-realtime",
9      "output_modalities": [
10       "audio"
11     ],
12     "instructions": "You are a friendly as
13     "tools": [],
14     "tool_choice": "auto",
15     "max_output_tokens": "inf",
16     "tracing": null,
17     "truncation": "auto",
18     "prompt": null,
19     "expires_at": 0,
20     "audio": {
```

Session response object

Response from creating a session and client secret for the Realtime API.

expires_at integer

Expiration timestamp for the client secret, in seconds since epoch.

session object

The session configuration for either a realtime or transcription session.

▼ Show possible types

value string

The generated client secret value.

OBJECT Session response object



```
1  {
2    "value": "ek_68af296e8e408191a1120ab6383
3    "expires_at": 1756310470,
4    "session": {
5      "type": "realtime",
6      "object": "realtime.session",
7      "id": "sess_C9CiUVUzUzYIssh3ELY1d",
8      "model": "gpt-4o-realtime-preview",
9      "output_modalities": [
10       "audio"
11     ],
12     "instructions": "You are a friendly as
13     "tools": [],
14     "tool_choice": "auto",
15     "max_output_tokens": "inf",
16     "tracing": null,
17     "truncation": "auto",
18     "prompt": null,
19     "expires_at": 0,
20     "audio": {
21       "input": {
22         "format": {
23           "type": "audio/pcm",
24           "rate": 24000
25         },
26         "transcription": null,
```

```
27     "noise_reduction": null,  
28     "turn_detection": {  
29         "type": "server_vad",  
30         "threshold": 0.5,  
31         "prefix_padding_ms": 300,  
32         "silence_duration_ms": 200,  
33         "idle_timeout_ms": null,  
34         "create_response": true
```

Client events

These are events that the OpenAI Realtime WebSocket server will accept from the client.

session.update

Send this event to update the session's default configuration. The client may send this event at any time to update any field, except for `voice`. However, note that once a session has been initialized with a particular `model`, it

OBJECT session.update



```
1  {  
2    "type": "session.update",  
3    "session": {
```

can't be changed to another model using `session.update` .

When the server receives a `session.update` , it will respond with a `session.updated` event showing the full, effective configuration. Only the fields that are present are updated. To clear a field like `instructions` , pass an empty string.

event_id string

Optional client-generated ID used to identify this event.

session object

Realtime session object configuration.

▼ Show properties

type string

The event type, must be `session.update` .

```

4  "type": "realtime",
5  "tools": [
6    {
7      "type": "function",
8      "name": "display_color_palette",
9      "description": "\nCall this functi
10     "parameters": {
11       "type": "object",
12       "strict": true,
13       "properties": {
14         "theme": {
15           "type": "string",
16           "description": "Description
17         },
18         "colors": {
19           "type": "array",
20           "description": "Array of fiv
21           "items": {
22             "type": "string",
23             "description": "Hex color
24           }
25         }
26       },
27       "required": [
28         "theme",
29         "colors"
30       ]
31     }
32   ]
33 },
34 "tool_choice": "auto"

```

input_audio_buffer.append

Send this event to append audio bytes to the input audio buffer. The audio buffer is temporary storage you can write to and later commit. In Server VAD mode, the audio buffer is used to detect speech and the server will decide when to commit. When Server VAD is disabled, you must commit the audio buffer manually.

The client may choose how much audio to place in each event up to a maximum of 15 MiB, for example streaming smaller chunks from the client may allow the VAD to be more responsive. Unlike made other client events, the server will not send a confirmation response to this event.

audio string

Base64-encoded audio bytes. This must be in the format specified by the `input_audio_format` field in the session configuration.

event_id string

Optional client-generated ID used to identify this event.

type string

The event type, must be `input_audio_buffer.append`.

OBJECT `input_audio_buffer.append`



```
1 {  
2   "event_id": "event_456",  
3   "type": "input_audio_buffer.append",  
4   "audio": "Base64EncodedAudioData"  
5 }
```

input_audio_buffer.commit

Send this event to commit the user input audio buffer, which will create a new user message item in the conversation.

This event will produce an error if the input audio buffer is empty. When in Server VAD mode, the client does not need to send this event, the server will commit the audio buffer automatically.

Committing the input audio buffer will trigger input audio transcription (if enabled in session configuration), but it will not create a response from the model. The server will respond with an `input_audio_buffer.committed` event.

event_id string

Optional client-generated ID used to identify this event.

type string

The event type, must be `input_audio_buffer.commit`.

OBJECT `input_audio_buffer.commit`



```
1 {  
2   "event_id": "event_789",  
3   "type": "input_audio_buffer.commit"  
4 }
```

input_audio_buffer.clear

Send this event to clear the audio bytes in the buffer. The server will respond with an `input_audio_buffer.cleared` event.

event_id string

Optional client-generated ID used to identify this event.

type string

The event type, must be `input_audio_buffer.clear` .

OBJECT `input_audio_buffer.clear`



```
1 {  
2   "event_id": "event_012",  
3   "type": "input_audio_buffer.clear"  
4 }
```

conversation.item.create

Add a new Item to the Conversation's context, including messages, function calls, and function call responses. This event can be used both to populate a "history" of the conversation and to add new items mid-stream, but has the current limitation that it cannot populate assistant audio messages.

OBJECT `conversation.item.create`



```
1 {  
2   "type": "conversation.item.create",  
3   "item": {  
4     "type": "message",  
5     "role": "user",
```

If successful, the server will respond with a

`conversation.item.created` event, otherwise an `error` event will be sent.

event_id string

Optional client-generated ID used to identify this event.

item object

A single item within a Realtime conversation.

▼ Show possible types

previous_item_id string

The ID of the preceding item after which the new item will be inserted. If not set, the new item will be appended to the end of the conversation. If set to `root`, the new item will be added to the beginning of the conversation. If set to an existing ID, it allows an item to be inserted mid-conversation. If the ID cannot be found, an error will be returned and the item will not be added.

type string

The event type, must be `conversation.item.create`.

```
6     "content": [  
7         {  
8             "type": "input_text",  
9             "text": "hi"  
10        }  
11    ]  
12 },  
13 "event_id": "b904fba0-0ec4-40af-8bbb-f90  
14 "timestamp": "2:30:35 PM"  
15 }
```

conversation.item.retrieve

Send this event when you want to retrieve the server's representation of a specific item in the conversation history. This is useful, for example, to inspect user audio after noise cancellation and VAD. The server will respond with a `conversation.item.retrieved` event, unless the item does not exist in the conversation history, in which case the server will respond with an error.

event_id string

Optional client-generated ID used to identify this event.

item_id string

The ID of the item to retrieve.

type string

The event type, must be `conversation.item.retrieve`.

OBJECT `conversation.item.retrieve`



```
1 {  
2   "event_id": "event_901",  
3   "type": "conversation.item.retrieve",  
4   "item_id": "msg_003"  
5 }
```

conversation.item.truncate

Send this event to truncate a previous assistant message's

OBJECT `conversation.item.truncate`



audio. The server will produce audio faster than realtime, so this event is useful when the user interrupts to truncate audio that has already been sent to the client but not yet played. This will synchronize the server's understanding of the audio with the client's playback.

Truncating audio will delete the server-side text transcript to ensure there is not text in the context that hasn't been heard by the user.

If successful, the server will respond with a

`conversation.item.truncated` event.

audio_end_ms integer

Inclusive duration up to which audio is truncated, in milliseconds. If the `audio_end_ms` is greater than the actual audio duration, the server will respond with an error.

content_index integer

The index of the content part to truncate. Set this to 0.

event_id string

Optional client-generated ID used to identify this event.

item_id string

The ID of the assistant message item to truncate. Only assistant

```
1 {  
2   "event_id": "event_678",  
3   "type": "conversation.item.truncate",  
4   "item_id": "msg_002",  
5   "content_index": 0,  
6   "audio_end_ms": 1500  
7 }
```

message items can be truncated.

type string

The event type, must be `conversation.item.truncate` .

conversation.item.delete

Send this event when you want to remove any item from the conversation history. The server will respond with a

`conversation.item.deleted` event, unless the item does not exist in the conversation history, in which case the server will respond with an error.

event_id string

Optional client-generated ID used to identify this event.

item_id string

The ID of the item to delete.

type string

The event type, must be `conversation.item.delete` .

OBJECT `conversation.item.delete`



```
1 {  
2   "event_id": "event_901",  
3   "type": "conversation.item.delete",  
4   "item_id": "msg_003"  
5 }
```

response.create

This event instructs the server to create a Response, which means triggering model inference. When in Server VAD mode, the server will create Responses automatically.

A Response will include at least one Item, and may have two, in which case the second will be a function call. These Items will be appended to the conversation history.

The server will respond with a `response.created` event, events for Items and content created, and finally a `response.done` event to indicate the Response is complete.

The `response.create` event includes inference configuration like `instructions`, and `temperature`. These fields will override the Session's configuration for this Response only.

event_id string

Optional client-generated ID used to identify this event.

OBJECT `response.create`



```
1 {  
2   "type": "response.create",  
3   "event_id": "xxx",  
4   "timestamp": "2:30:35 PM"  
5 }
```

response object

Create a new Realtime response with these parameters

▼ Show properties

type string

The event type, must be `response.create` .

response.cancel

Send this event to cancel an in-progress response. The server will respond with a `response.done` event with a status of `response.status=cancelled`. If there is no response to cancel, the server will respond with an error.

event_id string

Optional client-generated ID used to identify this event.

response_id string

A specific response ID to cancel - if not provided, will cancel an in-progress response in the default conversation.

type string

The event type, must be `response.cancel`.

OBJECT `response.cancel`



```
1 {  
2   "event_id": "event_567",  
3   "type": "response.cancel"  
4 }
```

transcription_session.update

Send this event to update a transcription session.

event_id string

Optional client-generated ID used to identify this event.

session object

Realtime transcription session object configuration.

▼ Show properties

type string

The event type, must be `transcription_session.update`.

OBJECT `transcription_session.update`



```
1  {
2    "type": "transcription_session.update",
3    "session": {
4      "input_audio_format": "pcm16",
5      "input_audio_transcription": {
6        "model": "gpt-4o-transcribe",
7        "prompt": "",
8        "language": ""
9      },
10     "turn_detection": {
11       "type": "server_vad",
12       "threshold": 0.5,
13       "prefix_padding_ms": 300,
14       "silence_duration_ms": 500,
15       "create_response": true,
16     },
17     "input_audio_noise_reduction": {
18       "type": "near_field"
19     },
20     "include": [
21       "item.input_audio_transcription.logp
22     ]
23   }
24 }
```

output_audio_buffer.clear

WebRTC Only: Emit to cut off the current audio response.

This will trigger the server to stop generating audio and emit a `output_audio_buffer.cleared` event. This event should be preceded by a `response.cancel` client event to stop the generation of the current response. [Learn more](#).

event_id string

The unique ID of the client event used for error handling.

type string

The event type, must be `output_audio_buffer.clear`.

OBJECT `output_audio_buffer.clear`



```
1 {  
2   "event_id": "optional_client_event_id",  
3   "type": "output_audio_buffer.clear"  
4 }
```

Server events

These are events emitted from the OpenAI Realtime WebSocket server to the client.

error

Returned when an error occurs, which could be a client problem or a server problem. Most errors are recoverable and the session will stay open, we recommend to implementors to monitor and log error messages by default.

error object

Details of the error.

⌵ Show properties

event_id string

The unique ID of the server event.

type string

The event type, must be `error`.

OBJECT error



```
1  {
2      "event_id": "event_890",
3      "type": "error",
4      "error": {
5          "type": "invalid_request_error",
6          "code": "invalid_event",
7          "message": "The 'type' field is mi
8          "param": null,
9          "event_id": "event_567"
10     }
11 }
```

session.created

Returned when a Session is created. Emitted automatically when a new connection is established as the first server event. This event will contain the default Session configuration.

event_id string

The unique ID of the server event.

session object

Realtime session object.

▼ Show properties

type string

The event type, must be `session.created`.

OBJECT session.created



```

1  {
2    "type": "session.created",
3    "event_id": "event_C9G5RJeJ2gF77mV7f2B1j",
4    "session": {
5      "type": "realtime",
6      "object": "realtime.session",
7      "id": "sess_C9G5QPteg4UIbotdKLoYQ",
8      "model": "gpt-4o-realtime-preview-2025",
9      "output_modalities": [
10       "audio"
11     ],
12     "instructions": "Your knowledge cutoff",
13     "tools": [],
14     "tool_choice": "auto",
15     "max_output_tokens": "inf",
16     "tracing": null,
17     "prompt": null,
18     "expires_at": 1756324625,
19     "audio": {
20       "input": {
21         "format": {
22           "type": "audio/pcm",
23           "rate": 24000
24         },
25         "transcription": null,
26         "noise_reduction": null,
27         "turn_detection": {
28           "type": "server_vad",
29           "threshold": 0.5,

```

```
30         "prefix_padding_ms": 300,  
31         "silence_duration_ms": 200,  
32         "idle_timeout_ms": null,  
33         "create_response": true,  
34         "interrupt_response": true
```

session.updated

Returned when a session is updated with a `session.update` event, unless there is an error.

event_id string

The unique ID of the server event.

session object

Realtime session object.

▼ Show properties

type string

The event type, must be `session.updated`.

OBJECT session.updated



```
1  {  
2    "type": "session.updated",  
3    "event_id": "event_C9G8mqI3IucaojlVKE8Cs",  
4    "session": {  
5      "type": "realtime",  
6      "object": "realtime.session",  
7      "id": "sess_C9G8l3zp50uFv4qgxfJ8o",  
8      "model": "gpt-4o-realtime-preview-2025",  
9      "output_modalities": [  
10       "audio"  
11     ],  
12     "instructions": "Your knowledge cutoff",  
13     "tools": [  
14       {  
15         "type": "function",  
16         "name": "display_color_palette",  
17         "description": "\nCall this functi
```

```
18     "parameters": {
19       "type": "object",
20       "strict": true,
21       "properties": {
22         "theme": {
23           "type": "string",
24           "description": "Description
25         },
26         "colors": {
27           "type": "array",
28           "description": "Array of five
29           "items": {
30             "type": "string",
31             "description": "Hex color
32           }
33         }
34       }
```

transcription_session.created

Returned when a transcription session is created.

event_id string

The unique ID of the server event.

session object

A Realtime transcription session configuration object.

▼ Show properties

type string

The event type, must be `transcription_session.created`.

OBJECT `transcription_session.created`



```
1  {
2    "event_id": "event_5566",
3    "type": "transcription_session.created",
4    "session": {
5      "id": "sess_001",
6      "object": "realtime.transcription_session",
7      "input_audio_format": "pcm16",
8      "input_audio_transcription": {
9        "model": "gpt-4o-transcribe",
10       "prompt": "",
11       "language": ""
12     },
13     "turn_detection": {
14       "type": "server_vad",
15       "threshold": 0.5,
16       "prefix_padding_ms": 300,
17       "silence_duration_ms": 500
18     },
19     "input_audio_noise_reduction": {
20       "type": "near_field"
21     },
22     "include": []
23   }
24 }
```

conversation.item.created

Returned when a conversation item is created. There are several scenarios that produce this event:

The server is generating a Response, which if successful will produce either one or two Items, which will be of type `message` (role `assistant`) or type `function_call`.

The input audio buffer has been committed, either by the client or the server (in `server_vad` mode). The server will take the content of the input audio buffer and add it to a new user message Item.

The client has sent a `conversation.item.create` event to add a new Item to the Conversation.

event_id string

The unique ID of the server event.

item object

A single item within a Realtime conversation.

OBJECT `conversation.item.created`



```
1  {
2    "event_id": "event_1920",
3    "type": "conversation.item.created",
4    "previous_item_id": "msg_002",
5    "item": {
6      "id": "msg_003",
7      "object": "realtime.item",
8      "type": "message",
9      "status": "completed",
10     "role": "user",
11     "content": []
12   }
13 }
```


▽ Show possible types

previous_item_id string or null

The ID of the preceding item in the Conversation context, allows the client to understand the order of the conversation. Can be `null` if the item has no predecessor.

type string

The event type, must be `conversation.item.created` .

conversation.item.added

Returned when a conversation item is added.

event_id string

The unique ID of the server event.

item object

A single item within a Realtime conversation.

▼ Show possible types

previous_item_id string or null

The ID of the item that precedes this one, if any. This is used to maintain ordering when items are inserted.

type string

The event type, must be `conversation.item.added`.

OBJECT `conversation.item.added`



```
1  {
2    "type": "conversation.item.added",
3    "event_id": "event_C9G8pjSJCfRNEhMEnYAVy",
4    "previous_item_id": null,
5    "item": {
6      "id": "item_C9G8pGVKYnaZu8PH5YQ90",
7      "type": "message",
8      "status": "completed",
9      "role": "user",
10     "content": [
11       {
12         "type": "input_text",
13         "text": "hi"
14       }
15     ]
16   },
17   "timestamp": "2:30:35 PM"
18 }
```

conversation.item.done

Returned when a conversation item is finalized.

event_id string

The unique ID of the server event.

item object

A single item within a Realtime conversation.

▼ Show possible types

previous_item_id string or null

The ID of the item that precedes this one, if any. This is used to maintain ordering when items are inserted.

type string

The event type, must be `conversation.item.done`.

OBJECT `conversation.item.done`



```
1  {
2    "type": "conversation.item.done",
3    "event_id": "event_C9G8ps2i70P5Wd60A0ftc",
4    "previous_item_id": null,
5    "item": {
6      "id": "item_C9G8pGVKYnaZu8PH5YQ90",
7      "type": "message",
8      "status": "completed",
9      "role": "user",
10     "content": [
11       {
12         "type": "input_text",
13         "text": "hi"
14       }
15     ]
16   },
17   "timestamp": "2:30:35 PM"
18 }
```

conversation.item.retrieved

Returned when a conversation item is retrieved with

`conversation.item.retrieve` .

event_id string

The unique ID of the server event.

item object

A single item within a Realtime conversation.

▼ Show possible types

type string

The event type, must be `conversation.item.retrieved` .

OBJECT `conversation.item.retrieved`



```
1  {
2      "event_id": "event_1920",
3      "type": "conversation.item.created",
4      "previous_item_id": "msg_002",
5      "item": {
6          "id": "msg_003",
7          "object": "realtime.item",
8          "type": "message",
9          "status": "completed",
10         "role": "user",
11         "content": [
12             {
13                 "type": "input_audio",
14                 "transcript": "hello how a",
15                 "audio": "base64encodedaud",
16             }
17         ]
18     }
19 }
```

conversation.item.input_audio_transcription.completed

This event is the output of audio transcription for user audio written to the user audio buffer. Transcription begins when the input audio buffer is committed by the client or server (in `server_vad` mode). Transcription runs asynchronously with Response creation, so this event may come before or after the Response events.

Realtime API models accept audio natively, and thus input transcription is a separate process run on a separate ASR (Automatic Speech Recognition) model. The transcript may diverge somewhat from the model's interpretation, and should be treated as a rough guide.

content_index integer

The index of the content part containing the audio.

event_id string

The unique ID of the server event.

item_id string

The ID of the user message item containing the audio.

logprobs array or null

The log probabilities of the transcription.

✎ Show properties

OBJECT conversation.item.input_audio_transc...



```

1  {
2      "event_id": "event_2122",
3      "type": "conversation.item.input_audio_transcription.completed",
4      "item_id": "msg_003",
5      "content_index": 0,
6      "transcript": "Hello, how are you?",
7      "usage": {
8          "type": "tokens",
9          "total_tokens": 48,
10         "input_tokens": 38,
11         "input_token_details": {
12             "text_tokens": 10,
13             "audio_tokens": 28,
14         },
15         "output_tokens": 10,
16     }
17 }
```

transcript string

The transcribed text.

type string

The event type, must be

`conversation.item.input_audio_transcription.completed`

.

usage object

Usage statistics for the transcription.

▼ Show possible types

conversation.item.input_audio_transcription.delta

Returned when the text value of an input audio transcription content part is updated.

content_index integer

The index of the content part in the item's content array.

delta string

The text delta.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

logprobs array or null

The log probabilities of the transcription.

▼ Show properties

type string

The event type, must be

`conversation.item.input_audio_transcription.delta` .

OBJECT `conversation.item.input_audio_transc...`



```
1 {  
2   "type": "conversation.item.input_audio_tr  
3   "event_id": "event_001",  
4   "item_id": "item_001",  
5   "content_index": 0,  
6   "delta": "Hello"  
7 }
```

conversation.item.input_audio_transcription.segment

Returned when an input audio transcription segment is identified for an item.

content_index integer

The index of the input audio content part within the item.

end number

End time of the segment in seconds.

event_id string

The unique ID of the server event.

id string

The segment identifier.

item_id string

The ID of the item containing the input audio content.

speaker string

The detected speaker label for this segment.

start number

Start time of the segment in seconds.

OBJECT conversation.item.input_audio_transc...



```
1  {
2      "event_id": "event_6501",
3      "type": "conversation.item.input_audio",
4      "item_id": "msg_011",
5      "content_index": 0,
6      "text": "hello",
7      "id": "seg_0001",
8      "speaker": "spk_1",
9      "start": 0.0,
10     "end": 0.4
11 }
```


text string

The text for this segment.

type string

The event type, must be

`conversation.item.input_audio_transcription.segment` .

conversation.item.input_audio_transcription.failed

Returned when input audio transcription is configured, and a transcription request for a user message failed. These events are separate from other `error` events so that the client can identify the related Item.

content_index integer

The index of the content part containing the audio.

error object

Details of the transcription error.

▼ Show properties

event_id string

The unique ID of the server event.

item_id string

The ID of the user message item.

type string

The event type, must be

`conversation.item.input_audio_transcription.failed` .

OBJECT `conversation.item.input_audio_transc...`



```
1  {
2    "event_id": "event_2324",
3    "type": "conversation.item.input_audio
4    "item_id": "msg_003",
5    "content_index": 0,
6    "error": {
7      "type": "transcription_error",
8      "code": "audio_unintelligible",
9      "message": "The audio could not be
10     "param": null
11   }
12 }
```

conversation.item.truncated

Returned when an earlier assistant audio message item is truncated by the client with a `conversation.item.truncate` event. This event is used to synchronize the server's understanding of the audio with the client's playback.

This action will truncate the audio and remove the server-side text transcript to ensure there is no text in the context that hasn't been heard by the user.

audio_end_ms integer

The duration up to which the audio was truncated, in milliseconds.

content_index integer

The index of the content part that was truncated.

event_id string

The unique ID of the server event.

item_id string

The ID of the assistant message item that was truncated.

type string

The event type, must be `conversation.item.truncated`.

OBJECT `conversation.item.truncated`



```
1 {  
2   "event_id": "event_2526",  
3   "type": "conversation.item.truncated",  
4   "item_id": "msg_004",  
5   "content_index": 0,  
6   "audio_end_ms": 1500  
7 }
```

conversation.item.deleted

Returned when an item in the conversation is deleted by the client with a `conversation.item.delete` event. This event is used to synchronize the server's understanding of the conversation history with the client's view.

event_id string

The unique ID of the server event.

item_id string

The ID of the item that was deleted.

type string

The event type, must be `conversation.item.deleted`.

OBJECT `conversation.item.deleted`



```
1 {  
2   "event_id": "event_2728",  
3   "type": "conversation.item.deleted",  
4   "item_id": "msg_005"  
5 }
```

input_audio_buffer.committed

Returned when an input audio buffer is committed, either by the client or automatically in server VAD mode. The

`item_id` property is the ID of the user message item that will be created, thus a `conversation.item.created` event will also be sent to the client.

event_id string

The unique ID of the server event.

item_id string

The ID of the user message item that will be created.

previous_item_id string or null

The ID of the preceding item after which the new item will be inserted. Can be `null` if the item has no predecessor.

type string

The event type, must be `input_audio_buffer.committed`.

OBJECT `input_audio_buffer.committed`



```
1 {  
2   "event_id": "event_1121",  
3   "type": "input_audio_buffer.committed",  
4   "previous_item_id": "msg_001",  
5   "item_id": "msg_002"  
6 }
```

input_audio_buffer.cleared

Returned when the input audio buffer is cleared by the client with a `input_audio_buffer.clear` event.

event_id string

The unique ID of the server event.

type string

The event type, must be `input_audio_buffer.cleared`.

OBJECT `input_audio_buffer.cleared`



```
1 {  
2   "event_id": "event_1314",  
3   "type": "input_audio_buffer.cleared"  
4 }
```

input_audio_buffer.speech_started

Sent by the server when in `server_vad` mode to indicate that speech has been detected in the audio buffer. This can happen any time audio is added to the buffer (unless speech is already detected). The client may want to use this event to interrupt audio playback or provide visual feedback to the user.

The client should expect to receive a

`input_audio_buffer.speech_stopped` event when speech

OBJECT `input_audio_buffer.speech_started`



```
1 {  
2   "event_id": "event_1516",  
3   "type": "input_audio_buffer.speech_star  
4   "audio_start_ms": 1000,  
5   "item_id": "msg_003"  
6 }
```

stops. The `item_id` property is the ID of the user message item that will be created when speech stops and will also be included in the `input_audio_buffer.speech_stopped` event (unless the client manually commits the audio buffer during VAD activation).

audio_start_ms integer

Milliseconds from the start of all audio written to the buffer during the session when speech was first detected. This will correspond to the beginning of audio sent to the model, and thus includes the `prefix_padding_ms` configured in the Session.

event_id string

The unique ID of the server event.

item_id string

The ID of the user message item that will be created when speech stops.

type string

The event type, must be `input_audio_buffer.speech_started`.

input_audio_buffer.speech_stopped

Returned in `server_vad` mode when the server detects the end of speech in the audio buffer. The server will also send an `conversation.item.created` event with the user message item that is created from the audio buffer.

audio_end_ms integer

Milliseconds since the session started when speech stopped. This will correspond to the end of audio sent to the model, and thus includes the `min_silence_duration_ms` configured in the Session.

event_id string

The unique ID of the server event.

item_id string

The ID of the user message item that will be created.

type string

The event type, must be `input_audio_buffer.speech_stopped`.

OBJECT `input_audio_buffer.speech_stopped`



```
1 {  
2   "event_id": "event_1718",  
3   "type": "input_audio_buffer.speech_stop  
4   "audio_end_ms": 2000,  
5   "item_id": "msg_003"  
6 }
```

input_audio_buffer.timeout_triggered

Returned when the server VAD timeout is triggered for the input audio buffer.

audio_end_ms integer

Millisecond offset where speech ended within the buffered audio.

audio_start_ms integer

Millisecond offset where speech started within the buffered audio.

event_id string

The unique ID of the server event.

item_id string

The ID of the item associated with this segment.

type string

The event type, must be

`input_audio_buffer.timeout_triggered`.

OBJECT `input_audio_buffer.timeout_triggered`



```
1 {  
2   "event_id": "event_6401",  
3   "type": "input_audio_buffer.timeout_tri  
4   "audio_start_ms": 1200,  
5   "audio_end_ms": 2150,  
6   "item_id": "msg_010"  
7 }
```

response.created

Returned when a new Response is created. The first event of response creation, where the response is in an initial state of

`in_progress`.

event_id string

The unique ID of the server event.

response object

The response resource.

▼ Show properties

type string

The event type, must be `response.created`.

OBJECT `response.created`



```

1  {
2    "type": "response.created",
3    "event_id": "event_C9G8pqbTEddBSIxbBN60s",
4    "response": {
5      "object": "realtime.response",
6      "id": "resp_C9G8p7IH2WxLbkgPNouYL",
7      "status": "in_progress",
8      "status_details": null,
9      "output": [],
10     "conversation_id": "conv_C9G8mmBkLhQJw",
11     "output_modalities": [
12       "audio"
13     ],
14     "max_output_tokens": "inf",
15     "audio": {
16       "output": {
17         "format": {
18           "type": "audio/pcm",
19           "rate": 24000
20         },
21         "voice": "marin"
22       }
23     },
24     "usage": null,
25     "metadata": null
26   },
27   "timestamp": "2:30:35 PM"
28 }
```

response.done

Returned when a Response is done streaming. Always emitted, no matter the final state. The Response object included in the `response.done` event will include all output items in the Response but will omit the raw audio data.

event_id string

The unique ID of the server event.

response object

The response resource.

✓ Show properties

type string

The event type, must be `response.done`.

OBJECT `response.done`



```
1  {
2    "event_id": "event_3132",
3    "type": "response.done",
4    "response": {
5      "id": "resp_001",
6      "object": "realtime.response",
7      "status": "completed",
8      "status_details": null,
9      "output": [
10       {
11         "id": "msg_006",
12         "object": "realtime.item",
13         "type": "message",
14         "status": "completed",
15         "role": "assistant",
16         "content": [
17           {
18             "type": "text",
19             "text": "Sure, how
20           }
21         ]
22       }
23     ]
24   }
25 }
```

```
23     ],
24     "usage": {
25         "total_tokens": 275,
26         "input_tokens": 127,
27         "output_tokens": 148,
28         "input_token_details": {
29             "cached_tokens": 384,
30             "text_tokens": 119,
31             "audio_tokens": 8,
32             "cached_tokens_details": {
33                 "text_tokens": 128,
```

response.output_item.added

Returned when a new Item is created during Response generation.

event_id string

The unique ID of the server event.

item object

A single item within a Realtime conversation.

▼ Show possible types

output_index integer

The index of the output item in the Response.

response_id string

The ID of the Response to which the item belongs.

type string

The event type, must be `response.output_item.added`.

OBJECT `response.output_item.added`



```
1  {
2      "event_id": "event_3334",
3      "type": "response.output_item.added",
4      "response_id": "resp_001",
5      "output_index": 0,
6      "item": {
7          "id": "msg_007",
8          "object": "realtime.item",
9          "type": "message",
10         "status": "in_progress",
11         "role": "assistant",
12         "content": []
13     }
14 }
```

response.output_item.done

Returned when an Item is done streaming. Also emitted when a Response is interrupted, incomplete, or cancelled.

event_id string

The unique ID of the server event.

item object

A single item within a Realtime conversation.

▼ Show possible types

output_index integer

The index of the output item in the Response.

response_id string

The ID of the Response to which the item belongs.

type string

The event type, must be `response.output_item.done`.

OBJECT `response.output_item.done`



```
1  {
2      "event_id": "event_3536",
3      "type": "response.output_item.done",
4      "response_id": "resp_001",
5      "output_index": 0,
6      "item": {
7          "id": "msg_007",
8          "object": "realtime.item",
9          "type": "message",
10         "status": "completed",
11         "role": "assistant",
12         "content": [
13             {
14                 "type": "text",
15                 "text": "Sure, I can help"
16             }
17         ]
18     }
19 }
```

response.content_part.added

Returned when a new content part is added to an assistant message item during response generation.

content_index integer

The index of the content part in the item's content array.

event_id string

The unique ID of the server event.

item_id string

The ID of the item to which the content part was added.

output_index integer

The index of the output item in the response.

part object

The content part that was added.

▼ Show properties

response_id string

The ID of the response.

type string

The event type, must be `response.content_part.added`.

OBJECT `response.content_part.added`



```
1  {
2      "event_id": "event_3738",
3      "type": "response.content_part.added",
4      "response_id": "resp_001",
5      "item_id": "msg_007",
6      "output_index": 0,
7      "content_index": 0,
8      "part": {
9          "type": "text",
10         "text": ""
11     }
12 }
```

response.content_part.done

Returned when a content part is done streaming in an assistant message item. Also emitted when a Response is interrupted, incomplete, or cancelled.

content_index integer

The index of the content part in the item's content array.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

part object

The content part that is done.

✓ Show properties

response_id string

OBJECT response.content_part.done



```
1  {
2      "event_id": "event_3940",
3      "type": "response.content_part.done",
4      "response_id": "resp_001",
5      "item_id": "msg_007",
6      "output_index": 0,
7      "content_index": 0,
8      "part": {
9          "type": "text",
10         "text": "Sure, I can help with tha
11     }
12 }
```

The ID of the response.

type string

The event type, must be `response.content_part.done` .

response.output_text.delta

Returned when the text value of an "output_text" content part is updated.

content_index integer

The index of the content part in the item's content array.

delta string

The text delta.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be `response.output_text.delta`.

OBJECT `response.output_text.delta`



```
1 {  
2   "event_id": "event_4142",  
3   "type": "response.output_text.delta",  
4   "response_id": "resp_001",  
5   "item_id": "msg_007",  
6   "output_index": 0,  
7   "content_index": 0,  
8   "delta": "Sure, I can h"  
9 }
```

response.output_text.done

Returned when the text value of an "output_text" content part is done streaming. Also emitted when a Response is interrupted, incomplete, or cancelled.

content_index integer

The index of the content part in the item's content array.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

text string

The final text content.

type string

The event type, must be `response.output_text.done` .

OBJECT `response.output_text.done`



```
1 {  
2   "event_id": "event_4344",  
3   "type": "response.output_text.done",  
4   "response_id": "resp_001",  
5   "item_id": "msg_007",  
6   "output_index": 0,  
7   "content_index": 0,  
8   "text": "Sure, I can help with that."  
9 }
```

response.output_audio_transcript.delta

Returned when the model-generated transcription of audio output is updated.

content_index integer

The index of the content part in the item's content array.

delta string

The transcript delta.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be

`response.output_audio_transcript.delta` .

OBJECT `response.output_audio_transcript.del...`



```
1 {
2   "event_id": "event_4546",
3   "type": "response.output_audio_transcri
4   "response_id": "resp_001",
5   "item_id": "msg_008",
6   "output_index": 0,
7   "content_index": 0,
8   "delta": "Hello, how can I a"
9 }
```

response.output_audio_transcript.done

Returned when the model-generated transcription of audio output is done streaming. Also emitted when a Response is interrupted, incomplete, or cancelled.

content_index integer

The index of the content part in the item's content array.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

transcript string

The final transcript of the audio.

type string

The event type, must be

`response.output_audio_transcript.done` .

OBJECT `response.output_audio_transcript.done`



```
1 {  
2   "event_id": "event_4748",  
3   "type": "response.output_audio_transcri  
4   "response_id": "resp_001",  
5   "item_id": "msg_008",  
6   "output_index": 0,  
7   "content_index": 0,  
8   "transcript": "Hello, how can I assist  
9 }
```

response.output_audio.delta

Returned when the model-generated audio is updated.

content_index integer

The index of the content part in the item's content array.

delta string

Base64-encoded audio data delta.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be `response.output_audio.delta`.

OBJECT `response.output_audio.delta`



```
1 {
2   "event_id": "event_4950",
3   "type": "response.output_audio.delta",
4   "response_id": "resp_001",
5   "item_id": "msg_008",
6   "output_index": 0,
7   "content_index": 0,
8   "delta": "Base64EncodedAudioDelta"
9 }
```

response.output_audio.done

Returned when the model-generated audio is done. Also emitted when a Response is interrupted, incomplete, or cancelled.

content_index integer

The index of the content part in the item's content array.

event_id string

The unique ID of the server event.

item_id string

The ID of the item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be `response.output_audio.done`.

OBJECT `response.output_audio.done`



```
1 {  
2   "event_id": "event_5152",  
3   "type": "response.output_audio.done",  
4   "response_id": "resp_001",  
5   "item_id": "msg_008",  
6   "output_index": 0,  
7   "content_index": 0  
8 }
```

`response.function_call_arguments.delta`

Returned when the model-generated function call arguments are updated.

call_id string

The ID of the function call.

delta string

The arguments delta as a JSON string.

event_id string

The unique ID of the server event.

item_id string

The ID of the function call item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be

`response.function_call_arguments.delta` .

OBJECT `response.function_call_arguments.del...`



```
1 {
2   "event_id": "event_5354",
3   "type": "response.function_call_argumen
4   "response_id": "resp_002",
5   "item_id": "fc_001",
6   "output_index": 0,
7   "call_id": "call_001",
8   "delta": "{\"location\": \"San\""
9 }
```

`response.function_call_arguments.done`

Returned when the model-generated function call arguments are done streaming. Also emitted when a Response is interrupted, incomplete, or cancelled.

arguments string

The final arguments as a JSON string.

call_id string

The ID of the function call.

event_id string

The unique ID of the server event.

item_id string

The ID of the function call item.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be

`response.function_call_arguments.done` .

OBJECT `response.function_call_arguments.done`



```
1 {
2   "event_id": "event_5556",
3   "type": "response.function_call_argumen
4   "response_id": "resp_002",
5   "item_id": "fc_001",
6   "output_index": 0,
7   "call_id": "call_001",
8   "arguments": "{\\"location\\": \\"San Fran
9 }
```

response.mcp_call_arguments.delta

Returned when MCP tool call arguments are updated during response generation.

delta string

The JSON-encoded arguments delta.

event_id string

The unique ID of the server event.

item_id string

The ID of the MCP tool call item.

obfuscation string or null

If present, indicates the delta text was obfuscated.

output_index integer

The index of the output item in the response.

response_id string

The ID of the response.

type string

The event type, must be `response.mcp_call_arguments.delta`.

OBJECT `response.mcp_call_arguments.delta`



```
1 {
2   "event_id": "event_6201",
3   "type": "response.mcp_call_arguments.de
4   "response_id": "resp_001",
5   "item_id": "mcp_call_001",
6   "output_index": 0,
7   "delta": "{\"partial\":true}"
8 }
```

response.mcp_call_arguments.done

Returned when MCP tool call arguments are finalized during response generation.

arguments string
The final JSON-encoded arguments string.


event_id string
The unique ID of the server event.

item_id string
The ID of the MCP tool call item.

output_index integer
The index of the output item in the response.

response_id string
The ID of the response.

type string
The event type, must be `response.mcp_call_arguments.done`.

OBJECT `response.mcp_call_arguments.done` 

```
1 {
2   "event_id": "event_6202",
3   "type": "response.mcp_call_arguments.do
4   "response_id": "resp_001",
5   "item_id": "mcp_call_001",
6   "output_index": 0,
7   "arguments": "{\"q\":\"docs\"}"
8 }
```

mcp_list_tools.in_progress

Returned when listing MCP tools is in progress for an item.

event_id string
The unique ID of the server event.

item_id string
The ID of the MCP list tools item.

type string
The event type, must be `mcp_list_tools.in_progress` .

OBJECT `mcp_list_tools.in_progress`



```
1 {
2   "event_id": "event_6101",
3   "type": "mcp_list_tools.in_progress",
4   "item_id": "mcp_list_tools_001"
5 }
```

mcp_list_tools.completed

Returned when listing MCP tools has completed for an item.

event_id string
The unique ID of the server event.

item_id string
The ID of the MCP list tools item.

type string
The event type, must be `mcp_list_tools.completed` .

OBJECT `mcp_list_tools.completed`



```
1 {
2   "event_id": "event_6102",
3   "type": "mcp_list_tools.completed",
4   "item_id": "mcp_list_tools_001"
5 }
```

mcp_list_tools.failed

Returned when listing MCP tools has failed for an item.

event_id string
The unique ID of the server event.

item_id string
The ID of the MCP list tools item.

type string
The event type, must be `mcp_list_tools.failed` .

OBJECT `mcp_list_tools.failed`



```
1 {
2   "event_id": "event_6103",
3   "type": "mcp_list_tools.failed",
4   "item_id": "mcp_list_tools_001"
5 }
```

response.mcp_call.in_progress

Returned when an MCP tool call has started and is in progress.

event_id string
The unique ID of the server event.

item_id string
The ID of the MCP tool call item.

output_index integer
The index of the output item in the response.

type string
The event type, must be `response.mcp_call.in_progress`.

OBJECT `response.mcp_call.in_progress`



```
1 {
2   "event_id": "event_6301",
3   "type": "response.mcp_call.in_progress"
4   "output_index": 0,
5   "item_id": "mcp_call_001"
6 }
```

response.mcp_call.completed

Returned when an MCP tool call has completed successfully.

event_id string
The unique ID of the server event.

item_id string
The ID of the MCP tool call item.

output_index integer
The index of the output item in the response.

type string
The event type, must be `response.mcp_call.completed`.

OBJECT `response.mcp_call.completed`



```
1 {
2   "event_id": "event_6302",
3   "type": "response.mcp_call.completed",
4   "output_index": 0,
5   "item_id": "mcp_call_001"
6 }
```

response.mcp_call.failed

Returned when an MCP tool call has failed.

event_id string
The unique ID of the server event.

item_id string
The ID of the MCP tool call item.

output_index integer
The index of the output item in the response.

type string
The event type, must be `response.mcp_call.failed`.

OBJECT `response.mcp_call.failed`



```
1 {
2   "event_id": "event_6303",
3   "type": "response.mcp_call.failed",
4   "output_index": 0,
5   "item_id": "mcp_call_001"
6 }
```

transcription_session.updated

Returned when a transcription session is updated with a `transcription_session.update` event, unless there is an error.

event_id string

The unique ID of the server event.

session object

A Realtime transcription session configuration object.

✓ Show properties

type string

The event type, must be `transcription_session.updated`.

OBJECT `transcription_session.updated`



```
1  {
2    "event_id": "event_5678",
3    "type": "transcription_session.updated",
4    "session": {
5      "id": "sess_001",
6      "object": "realtime.transcription_sess",
7      "input_audio_format": "pcm16",
8      "input_audio_transcription": {
9        "model": "gpt-4o-transcribe",
10       "prompt": "",
11       "language": ""
12     },
13     "turn_detection": {
14       "type": "server_vad",
15       "threshold": 0.5,
16       "prefix_padding_ms": 300,
17       "silence_duration_ms": 500,
18       "create_response": true,
19       // "interrupt_response": false -- t
20     },
21     "input_audio_noise_reduction": {
22       "type": "near_field"
23     },
24     "include": [
25       "item.input_audio_transcription.avg_
26     ],
27   }
28 }
```

rate_limits.updated

Emitted at the beginning of a Response to indicate the updated rate limits. When a Response is created some tokens will be "reserved" for the output tokens, the rate limits shown here reflect that reservation, which is then adjusted accordingly once the Response is completed.

event_id string

The unique ID of the server event.

rate_limits array

List of rate limit information.

✓ Show properties

type string

The event type, must be `rate_limits.updated`.

OBJECT `rate_limits.updated`



```
1  {
2    "event_id": "event_5758",
3    "type": "rate_limits.updated",
4    "rate_limits": [
5      {
6        "name": "requests",
7        "limit": 1000,
8        "remaining": 999,
9        "reset_seconds": 60
10     },
11     {
12       "name": "tokens",
13       "limit": 50000,
14       "remaining": 49950,
15       "reset_seconds": 60
16     }
17   ]
18 }
```