Live API - WebSockets API reference

Preview: The Live API is in preview.

The Live API is a stateful API that uses WebSockets (https://en.wikipedia.org/wiki/WebSocket). In this section, you'll find additional details regarding the WebSockets API.

Sessions

A WebSocket connection establishes a session between the client and the Gemini server. After a client initiates a new connection the session can exchange messages with the server to:

- Send text, audio, or video to the Gemini server.
- Receive audio, text, or function call requests from the Gemini server.

WebSocket connection

To start a session, connect to this websocket endpoint:

```
wss://generativelanguage.googleapis.com/ws/google.ai.generativelanguage.v1beta.G
```

Note: The URL is for version `v1beta`.

Session configuration

The initial message after connection sets the session configuration, which includes the model, generation parameters, system instructions, and tools.

You can change the configuration parameters except the model during the session.

See the following example configuration. Note that the name casing in SDKs may vary. You can look up the Python SDK configuration options here (https://github.com/googleapis/python-genai/blob/main/google/genai/types.py).

```
{
  "model": string,
  "generationConfig": {
    "candidateCount": integer,
    "maxOutputTokens": integer,
    "temperature": number,
    "topP": number,
    "topK": integer,
    "presencePenalty": number,
    "frequencyPenalty": number,
    "responseModalities": [string],
    "speechConfig": object,
    "mediaResolution": object
  },
  "systemInstruction": string,
  "tools": [object]
}
```

For more information on the API field, see generationConfig (/api/generate-content#v1beta.GenerationConfig).

Send messages

To exchange messages over the WebSocket connection, the client must send a JSON object over an open WebSocket connection. The JSON object must have **exactly one** of the fields from the following object set:

```
{
  "setup": BidiGenerateContentSetup,
  "clientContent": BidiGenerateContentClientContent,
  "realtimeInput": BidiGenerateContentRealtimeInput,
  "toolResponse": BidiGenerateContentToolResponse
}
```

Supported client messages

See the supported client messages in the following table:

Message	Description
`BidiGenerateContentSetup`	Session configuration to be sent in the first message
`BidiGenerateContentClientContent`	Incremental content update of the current conversation delivered from the client
`BidiGenerateContentRealtimeInput`	Real time audio, video, or text input
`BidiGenerateContentToolResponse`	Response to a `ToolCallMessage` received from the server

Receive messages

To receive messages from Gemini, listen for the WebSocket 'message' event, and then parse the result according to the definition of the supported server messages.

See the following:

```
async with client.aio.live.connect(model='...', config=config) as session:
    await session.send(input='Hello world!', end_of_turn=True)
    async for message in session.receive():
        print(message)
```

Server messages may have a `usageMetadata` (#UsageMetadata) field but will otherwise include **exactly one** of the other fields from the `BidiGenerateContentServerMessage` (#BidiGenerateContentServerMessage) message. (The `messageType` union is not expressed in JSON so the field will appear at the top-level of the message.)

Messages and events

ActivityEnd

This type has no fields.

Marks the end of user activity.

ActivityHandling

The different ways of handling user activity.

Enums	
`ACTIVITY_HANDLING_UNSPECIFIED`	If unspecified, the default behavior is `START_OF_ACTIVITY_INTERRUPTS`.
`START_OF_ACTIVITY_INTERRUPTS`	If true, start of activity will interrupt the model's response (also called "barge in"). The model's current response will be cut-off in the moment of the interruption. This is the default behavior.
`NO_INTERRUPTION`	The model's response will not be interrupted.

ActivityStart

This type has no fields.

Marks the start of user activity.

AudioTranscriptionConfig

This type has no fields.

The audio transcription configuration.

AutomaticActivityDetection

Configures automatic detection of activity.

Fields	
`disabled`	`bool` Optional. If enabled (the default), detected voice and text input count as activity. If disabled, the client must send activity signals.
`startOfSpeech Sensitivity`	`StartSensitivity` (#RealtimeInputConfig.AutomaticActivityDetection.StartSensitivity) Optional. Determines how likely speech is to be detected.
`prefixPaddingMs`	`int32` Optional. The required duration of detected speech before start-of-speech is committed. The lower this value, the more sensitive the start-of-speech detection is and shorter speech can be recognized. However, this also increases the probability of false positives.
`endOfSpeech Sensitivity`	`EndSensitivity` (#RealtimeInputConfig.AutomaticActivityDetection.EndSensitivity) Optional. Determines how likely detected speech is ended.
`silenceDurationMs`	`int32` Optional. The required duration of detected non-speech (e.g. silence) before end-of-speech is committed. The larger this value, the longer speech gaps can be without interrupting the user's activity but this will increase the model's latency.

BidiGenerateContentClientContent

Incremental update of the current conversation delivered from the client. All of the content here is unconditionally appended to the conversation history and used as part of the prompt to the model to generate content.

A message here will interrupt any current model generation.

Fields	
`turns[]`	`Content` (#Content) Optional. The content appended to the current conversation with the model. For single-turn queries, this is a single instance. For multi-turn queries, this is a repeated field that contains conversation history and the latest request.
`turnComplete`	`bool` Optional. If true, indicates that the server content generation should start with the currently accumulated prompt. Otherwise, the server awaits additional messages before starting generation.

BidiGenerateContentRealtimeInput

User input that is sent in real time.

The different modalities (audio, video and text) are handled as concurrent streams. The ordering across these streams is not guaranteed.

This is different from `BidiGenerateContentClientContent` (#BidiGenerateContentClientContent) in a few ways:

- Can be sent continuously without interruption to model generation.
- If there is a need to mix data interleaved across the `BidiGenerateContentClientContent` (#BidiGenerateContentClientContent) and the `BidiGenerateContentRealtimeInput` (#BidiGenerateContentRealtimeInput), the server attempts to optimize for best response, but there are no guarantees.
- End of turn is not explicitly specified, but is rather derived from user activity (for example, end of speech).
- Even before the end of turn, the data is processed incrementally to optimize for a fast start of the response from the model.

Fields	
`mediaChunks[]`	**`Blob`** (#Blob) Optional. Inlined bytes data for media input. Multiple **`mediaChunks`** are not supported, all but the first will be ignored. DEPRECATED: Use one of **`audio`**, **`video`**, or **`text`** instead.
`audio`	**`Blob`** (#Blob) Optional. These form the realtime audio input stream.
`video`	**`Blob`** (#Blob) Optional. These form the realtime video input stream.
`activityStart`	**`ActivityStart`** (#BidiGenerateContentRealtimeInput.ActivityStart) Optional. Marks the start of user activity. This can only be sent if automatic (i.e. server-side) activity detection is disabled.
`activityEnd`	**`ActivityEnd`** (#BidiGenerateContentRealtimeInput.ActivityEnd) Optional. Marks the end of user activity. This can only be sent if automatic (i.e. server-side) activity detection is disabled.
`audioStreamEnd`	`bool` Optional. Indicates that the audio stream has ended, e.g. because the microphone was turned off. This should only be sent when automatic activity detection is enabled (which is the default). The client can reopen the stream by sending an audio message.
`text`	`string` Optional. These form the realtime text input stream.

BidiGenerateContentServerContent

Incremental server update generated by the model in response to client messages.

Content is generated as quickly as possible, and not in real time. Clients may choose to buffer and play it out in real time.

Fields	
`generationComplete`	`bool` Output only. If true, indicates that the model is done generating. When model is interrupted while generating there will be no 'generation_complete' message in interrupted turn, it will go through 'interrupted > turn_complete'. When model assumes realtime playback there will be delay between generation_complete and turn_complete that is caused by model waiting for playback to finish.
`turnComplete`	`bool` Output only. If true, indicates that the model has completed its turn. Generation will only start in response to additional client messages.
`interrupted`	`bool` Output only. If true, indicates that a client message has interrupted current model generation. If the client is playing out the content in real time, this is a good signal to stop and empty the current playback queue.
`groundingMetadata`	`GroundingMetadata` (#GroundingMetadata) Output only. Grounding metadata for the generated content.
`inputTranscription`	`BidiGenerateContentTranscription` (#BidiGenerateContentTranscription) Output only. Input audio transcription. The transcription is sent independently of the other server messages and there is no guaranteed ordering.

Fields	
`outputTranscription`	[`BidiGenerateContentTranscription`](#BidiGenerateContentTranscription) (#BidiGenerateContentTranscription) Output only. Output audio transcription. The transcription is sent independently of the other server messages and there is no guaranteed ordering, in particular not between `serverContent` and this `outputTranscription`.
`modelTurn`	[`Content`](#Content) (#Content) Output only. The content that the model has generated as part of the current conversation with the user.

## BidiGenerateContentServerMessage

Response message for the BidiGenerateContent call.

Fields	
`usageMetadata`	[`UsageMetadata`](#UsageMetadata) (#UsageMetadata) Output only. Usage metadata about the response(s).
Union field `messageType`. The type of the message. `messageType` can be only one of the following:	
`setupComplete`	[`BidiGenerateContentSetupComplete`](#BidiGenerateContentSetupComplete) (#BidiGenerateContentSetupComplete) Output only. Sent in response to a `BidiGenerateContentSetup` message from the client when setup is complete.
`serverContent`	[`BidiGenerateContentServerContent`](#BidiGenerateContentServerContent) (#BidiGenerateContentServerContent) Output only. Content generated by the model in response to client messages.
`toolCall`	[`BidiGenerateContentToolCall`](#BidiGenerateContentToolCall) (#BidiGenerateContentToolCall)

Fields	
	Output only. Request for the client to execute the `functionCalls` and return the responses with the matching `id`s.
`toolCall Cancellation`	`BidiGenerateContentToolCallCancellation` (#BidiGenerateContentToolCallCancellation) Output only. Notification for the client that a previously issued `ToolCallMessage` with the specified `id`s should be cancelled.
`goAway`	`GoAway` (#GoAway) Output only. A notice that the server will soon disconnect.
`sessionResumption Update`	`SessionResumptionUpdate` (#SessionResumptionUpdate) Output only. Update of the session resumption state.

## BidiGenerateContentSetup

Message to be sent in the first (and only in the first) `BidiGenerateContentClientMessage`. Contains configuration that will apply for the duration of the streaming RPC.

Clients should wait for a `BidiGenerateContentSetupComplete` message before sending any additional messages.

Fields	
`model`	`string` Required. The model's resource name. This serves as an ID for the Model to use. Format: `models/{model}`
`generationConfig`	`GenerationConfig` (#GenerationConfig) Optional. Generation config. The following fields are not supported:

Fields	
	• `responseLogprobs` • `responseMimeType` • `logprobs` • `responseSchema` • `stopSequence` • `routingConfig` • `audioTimestamp`
`systemInstruction`	**`Content`** (#Content) Optional. The user provided system instructions for the model. Note: Only text should be used in parts and content in each part will be in a separate paragraph.
`tools[]`	**`Tool`** (#Tool) Optional. A list of `Tools` the model may use to generate the next response. A `Tool` is a piece of code that enables the system to interact with external systems to perform an action, or set of actions, outside of knowledge and scope of the model.
`realtimeInputConfig`	**`RealtimeInputConfig`** (#RealtimeInputConfig) Optional. Configures the handling of realtime input.
`sessionResumption`	**`SessionResumptionConfig`** (#SessionResumptionConfig) Optional. Configures session resumption mechanism. If included, the server will send `SessionResumptionUpdate` messages.
`contextWindowCompression`	**`ContextWindowCompressionConfig`** (#ContextWindowCompressionConfig) Optional. Configures a context window compression mechanism.

Fields	
	If included, the server will automatically reduce the size of the context when it exceeds the configured length.
`inputAudio Transcription`	**`AudioTranscriptionConfig`** (#AudioTranscriptionConfig) Optional. If set, enables transcription of voice input. The transcription aligns with the input audio language, if configured.
`outputAudio Transcription`	**`AudioTranscriptionConfig`** (#AudioTranscriptionConfig) Optional. If set, enables transcription of the model's audio output. The transcription aligns with the language code specified for the output audio, if configured.

BidiGenerateContentSetupComplete

This type has no fields.

Sent in response to a `BidiGenerateContentSetup` message from the client.

BidiGenerateContentToolCall

Request for the client to execute the `functionCalls` and return the responses with the matching `ids`.

Fields	
`functionCalls[]`	**`FunctionCall`** (#FunctionCall) Output only. The function call to be executed.

BidiGenerateContentToolCallCancellation

Notification for the client that a previously issued `ToolCallMessage` with the specified `ids` should not have been executed and should be cancelled. If there were side-effects to those tool calls, clients may attempt to undo the tool calls. This message occurs only in cases where the clients interrupt server turns.

Fields	
`ids[]`	`string` Output only. The ids of the tool calls to be cancelled.

BidiGenerateContentToolResponse

Client generated response to a `ToolCall` received from the server. Individual `FunctionResponse` objects are matched to the respective `FunctionCall` objects by the `id` field.

Note that in the unary and server-streaming GenerateContent APIs function calling happens by exchanging the `Content` parts, while in the bidi GenerateContent APIs function calling happens over these dedicated set of messages.

Fields	
`functionResponses[]`	`FunctionResponse` (#FunctionResponse) Optional. The response to the function calls.

BidiGenerateContentTranscription

Transcription of audio (input or output).

Fields	
`text`	`string` Transcription text.

ContextWindowCompressionConfig

Enables context window compression — a mechanism for managing the model's context window so that it does not exceed a given length.

Fields	
Union field **`compressionMechanism`**. The context window compression mechanism used. **`compressionMechanism`** can be only one of the following:	
`slidingWindow`	**`SlidingWindow`** (#ContextWindowCompressionConfig.SlidingWindow) A sliding-window mechanism.
`triggerTokens`	**`int64`** The number of tokens (before running a turn) required to trigger a context window compression. This can be used to balance quality against latency as shorter context windows may result in faster model responses. However, any compression operation will cause a temporary latency increase, so they should not be triggered frequently. If not set, the default is 80% of the model's context window limit. This leaves 20% for the next user request/model response.

EndSensitivity

Determines how end of speech is detected.

Enums	
`END_SENSITIVITY_UNSPECIFIED`	The default is END_SENSITIVITY_HIGH.
`END_SENSITIVITY_HIGH`	Automatic detection ends speech more often.

Enums	
`END_SENSITIVITY_LOW`	Automatic detection ends speech less often.

GoAway

A notice that the server will soon disconnect.

Fields	
`timeLeft`	**`Duration`** (https://protobuf.dev/reference/protobuf/google.protobuf/#duration) The remaining time before the connection will be terminated as ABORTED. This duration will never be less than a model-specific minimum, which will be specified together with the rate limits for the model.

## RealtimeInputConfig

Configures the realtime input behavior in `BidiGenerateContent`.

Fields	
`automaticActivityDetection`	**`AutomaticActivityDetection`** (#RealtimeInputConfig.AutomaticActivityDetection) Optional. If not set, automatic activity detection is enabled by default. If automatic voice detection is disabled, the client must send activity signals.
`activityHandling`	**`ActivityHandling`** (#RealtimeInputConfig.ActivityHandling) Optional. Defines what effect activity has.
`turnCoverage`	**`TurnCoverage`** (#RealtimeInputConfig.TurnCoverage) Optional. Defines which input is included in the user's turn.

SessionResumptionConfig

Session resumption configuration.

This message is included in the session configuration as `BidiGenerateContentSetup.sessionResumption`. If configured, the server will send `SessionResumptionUpdate` messages.

Fields	
`handle`	`string` The handle of a previous session. If not present then a new session is created. Session handles come from `SessionResumptionUpdate.token` values in previous connections.

SessionResumptionUpdate

Update of the session resumption state.

Only sent if `BidiGenerateContentSetup.sessionResumption` was set.

Fields	
`newHandle`	`string` New handle that represents a state that can be resumed. Empty if `resumable`=false.
`resumable`	`bool` True if the current session can be resumed at this point. Resumption is not possible at some points in the session. For example, when the model is executing function calls or generating. Resuming the session (using a previous session token) in such a state will result in some data loss. In these cases, `newHandle` will be empty and `resumable` will be false.

SlidingWindow

The SlidingWindow method operates by discarding content at the beginning of the context window. The resulting context will always begin at the start of a USER role turn. System instructions and any `BidiGenerateContentSetup.prefixTurns` will always remain at the beginning of the result.

Fields	
`targetTokens`	`int64` The target number of tokens to keep. The default value is trigger_tokens/2. Discarding parts of the context window causes a temporary latency increase so this value should be calibrated to avoid frequent compression operations.

StartSensitivity

Determines how start of speech is detected.

Enums	
`START_SENSITIVITY_UNSPECIFIED`	The default is START_SENSITIVITY_HIGH.
`START_SENSITIVITY_HIGH`	Automatic detection will detect the start of speech more often.
`START_SENSITIVITY_LOW`	Automatic detection will detect the start of speech less often.

TurnCoverage

Options about which input is included in the user's turn.

Enums	
`TURN_COVERAGE_UNSPECIFIED`	If unspecified, the default behavior is `TURN_INCLUDES_ONLY_ACTIVITY`.
`TURN_INCLUDES_ONLY_ACTIVITY`	The users turn only includes activity since the last turn, excluding inactivity (e.g. silence on the audio stream). This is the default behavior.
`TURN_INCLUDES_ALL_INPUT`	The users turn includes all realtime input since the last turn, including inactivity (e.g. silence on the audio stream).

UsageMetadata

Usage metadata about response(s).

Fields	
`promptTokenCount`	`int32` Output only. Number of tokens in the prompt. When `cachedContent` is set, this is still the total effective prompt size meaning this includes the number of tokens in the cached content.
`cachedContentTokenCount`	`int32` Number of tokens in the cached part of the prompt (the cached content)
`responseTokenCount`	`int32` Output only. Total number of tokens across all the generated response candidates.
`toolUsePromptTokenCount`	`int32` Output only. Number of tokens present in tool-use prompt(s).
`thoughtsTokenCount`	`int32`

Fields	
	Output only. Number of tokens of thoughts for thinking models.
`totalTokenCount`	`int32` Output only. Total token count for the generation request (prompt + response candidates).
`promptTokensDetails[]`	`ModalityTokenCount` (#ModalityTokenCount) Output only. List of modalities that were processed in the request input.
`cacheTokensDetails[]`	`ModalityTokenCount` (#ModalityTokenCount) Output only. List of modalities of the cached content in the request input.
`responseTokensDetails[]`	`ModalityTokenCount` (#ModalityTokenCount) Output only. List of modalities that were returned in the response.
`toolUsePromptTokensDetails[]`	`ModalityTokenCount` (#ModalityTokenCount) Output only. List of modalities that were processed for tool-use request inputs.

More information on common types

For more information on the commonly-used API resource types `Blob`, `Content`, `FunctionCall`, `FunctionResponse`, `GenerationConfig`, `GroundingMetadata`, `ModalityTokenCount`, and `Tool`, see Generating content (/api/generate-content).

Last updated 2025-05-12 UTC.