

Introduction

This API reference describes the RESTful, streaming, and realtime APIs you can use to interact with the OpenAI platform. REST APIs are usable via HTTP in any environment that supports HTTP requests. Language-specific SDKs are listed [on the libraries page](#).

Authentication

The OpenAI API uses API keys for authentication. Create, manage, and learn more about API keys in your [organization settings](#).

Remember that your API key is a secret! Do not share it with others or expose it in any client-side code (browsers, apps). API keys should be securely loaded from an environment variable or key management service on the server.

API keys should be provided via [HTTP Bearer authentication](#).

Authorization: Bearer OPENAI_API_KEY

If you belong to multiple organizations or access projects through a legacy user API key, pass a header to specify which organization and project to use for an API request:

```
curl https://api.openai.com/v1/models \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Organization: $ORGANIZATION_ID" \
-H "OpenAI-Project: $PROJECT_ID"
```

Usage from these API requests counts as usage for the specified organization and project. Organization IDs can be found on your [organization settings](#) page. Project IDs can be found on your [general settings](#) page by selecting the specific project.

Debugging requests^②

In addition to [error codes](#) returned from API responses, you can inspect HTTP response headers containing the unique ID of a particular API request or information about rate limiting applied to your requests. Below is an incomplete list of HTTP headers returned with API responses:

API meta information

`openai-organization` : The [organization](#) associated with the request

`openai-processing-ms` : Time taken processing your API request

`openai-version` : REST API version used for this request (currently `2020-10-01`)

`x-request-id` : Unique identifier for this API request (used in troubleshooting)

Rate limiting information

`x-ratelimit-limit-requests`

`x-ratelimit-limit-tokens`

`x-ratelimit-remaining-requests`

`x-ratelimit-remaining-tokens`

`x-ratelimit-reset-requests`

`x-ratelimit-reset-tokens`

OpenAI recommends logging request IDs in production deployments for more efficient troubleshooting with our [support team](#), should the need arise. Our [official SDKs](#) provide a property on top-level response objects containing the value of the `x-request-id` header.

Supplying your own request ID with `x-Client-Request-Id` ?

In addition to the server-generated `x-request-id`, you can supply your own unique identifier for each request via the `x-Client-Request-Id` request header. This header is not added automatically; you must explicitly set it on the request.

When you include `X-Client-Request-Id` :

You control the ID format (for example, a UUID or your internal trace ID), but it must contain only ASCII characters and be no more than 512 characters long; otherwise, the request will fail with a 400 error. We strongly recommend making this value unique per request.

OpenAI will log this value in our internal logs for supported endpoints, including chat/completions, embeddings, responses, and more.

In cases like timeouts or network issues when you can't get the `X-Request-Id` response header, you can share the `X-Client-Request-Id` value with our support team, and we can look up whether we received the request and when.

Example:

```
curl https://api.openai.com/v1/chat/completions \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "X-Client-Request-Id: 123e4567-e89b-12d3-a456-426614174000"
```

Backward compatibility

OpenAI is committed to providing stability to API users by avoiding breaking changes in major API versions whenever reasonably possible. This includes:

The REST API (currently `v1`)

Our first-party [SDKs](#) (released SDKs adhere to [semantic versioning](#))

Model families (like `gpt-4o` or `o4-mini`)

Model prompting behavior between snapshots is subject to change. Model outputs are by their nature variable, so expect changes in prompting and model behavior between snapshots.

For example, if you moved from `gpt-4o-2024-05-13` to `gpt-4o-2024-08-06`, the same `system` or `user` messages could function differently between versions. The best way to ensure consistent prompting behavior and model output is to use pinned model versions, and to implement [evals](#) for your applications.

Backwards-compatible API changes:

Adding new resources (URLs) to the REST API and SDKs

Adding new optional API parameters

Adding new properties to JSON response objects or event data

Changing the order of properties in a JSON response object

Changing the length or format of opaque strings, like resource identifiers and UUIDs

Adding new event types (in either streaming or the Realtime API)

See the [changelog](#) for a list of backwards-compatible changes and rare breaking changes.

NEXT
Responses

Responses



OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

[Extend the models with tools](#)

Create a model response



POST <https://api.openai.com/v1/responses>

Creates a model response. Provide text or image inputs to generate text or JSON outputs. Have the model call your own custom code or use built-in tools like web search or file search to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

[› Show possible types](#)

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

`web_search_call.action.sources` : Include the sources of the web search tool call.

`code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.

`computer_call_output.output.image_url` : Include image urls from the computer call output.

`file_search_call.results` : Include the search results of the file search tool call.

`message.input_image.image_url` : Include image urls from the input message.

`message.output_text.logprobs` : Include logprobs with assistant messages.

`reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

[Text inputs and outputs](#)

[Image inputs](#)

[File inputs](#)

[Conversation state](#)

[Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Text input](#)

[Image input](#)

[File input](#)

[Web search](#)

[File search](#)

[Streaming](#)

[Functions](#)

[Reasoning](#)

Example request

```
from openai import OpenAI

client = OpenAI()

response = client.responses.create(
    model="gpt-4.1",
    input="Tell me a three sentence bedtime story about a unicorn."
)

print(response)
```

Response

```
{
  "id": "resp_67ccd2bed1ec8190b14f964abc0542670bb6a6b452d3795b",
  "object": "response",
  "created_at": 1741476542,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4.1-2025-04-14",
  "output": [
    {
      "type": "message",
      "id": "msg_67ccd2bf17f0819081ff3bb2cf6508e60bb6a6b452d3795b",
```

```
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "In a peaceful grove beneath a silver moon, a unicorn named Lumina discovered a
    "annotations": []
  }
]
},
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
```

```
"input_tokens": 36,  
"input_tokens_details": {  
    "cached_tokens": 0  
},  
"output_tokens": 87,  
"output_tokens_details": {  
    "reasoning_tokens": 0  
},  
"total_tokens": 123  
},  
"user": null,  
"metadata": {}  
}
```

Get a model response



```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.retrieve("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4o-2024-08-06",
  "output": [
    {
      "type": "message",
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",
      "status": "completed",
    }
  ]
}
```

```
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn break",
    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
  "input_tokens": 32,
```

```
"input_tokens_details": {  
    "cached_tokens": 0  
,  
    "output_tokens": 18,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
,  
        "total_tokens": 50  
,  
        "user": null,  
        "metadata": {}  
}
```

Delete a model response



```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.delete("resp_123")
print(response)
```

Response

```
{
  "id": "resp_6786a1bec27481909a17d673315b29f6",
  "object": "response",
  "deleted": true
}
```

Cancel a response



```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A [Response](#) object.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.cancel("resp_123")
print(response)
```

Response

```
{  
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",  
  "object": "response",  
  "created_at": 1741386163,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {  
      "type": "message",  
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",  
      "status": "completed",  
      "role": "assistant",  
      "content": [  
        {  
          "type": "output_text",  
          "text": "Silent circuits hum,  \\nThoughts emerge in data streams–  \\nDigital dawn breaks",  
          "annotations": []  
        }  
      ]  
    },  
    {"parallel_tool_calls": true,  
     "previous_response_id": null},
```

```
"reasoning": {  
    "effort": null,  
    "summary": null  
},  
"store": true,  
"temperature": 1.0,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [],  
"top_p": 1.0,  
"truncation": "disabled",  
"usage": {  
    "input_tokens": 32,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 18,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
    },  
    "total_tokens": 50  
},  
"user": null,  
"metadata": {}  
}
```

Compact a response



POST <https://api.openai.com/v1/responses/compact>

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

> Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

```
from openai import OpenAI

client = OpenAI()

compacted_response = client.responses.compact(
    model="gpt-5.1-codex-max",
    input=[
        {
            "role": "user",
            "content": "Create a simple landing page for a dog petting cafe.",
        }
    ]
)
```

```
},
# All items returned from previous requests are included here, like reasoning, message, function, etc.
{
    "id": "msg_001",
    "type": "message",
    "status": "completed",
    "content": [
        {
            "type": "output_text",
            "annotations": [],
            "logprobs": [],
            "text": "Below is a single file, ready-to-use landing page for a dog petting café:..."
        },
    ],
    "role": "assistant",
},
]
)
# Pass the compacted_response.output as input to the next request
print(compacted_response)
```

Response

```
{
    "id": "resp_001",
    "object": "response.compaction",
    "created_at": 1764967971,
    "output": [
```

```
{  
    "id": "msg_000",  
    "type": "message",  
    "status": "completed",  
    "content": [  
        {  
            "type": "input_text",  
            "text": "Create a simple landing page for a dog petting cafe."  
        }  
    ],  
    "role": "user"  
},  
{  
    "id": "cmp_001",  
    "type": "compaction",  
    "encrypted_content": "gAAAAABpM0Yj-...="  

```

List input items



```
GET https://api.openai.com/v1/responses/{response_id}/input_items
```

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

`asc` : Return the input items in ascending order.

`desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.input_items.list("resp_123")
print(response.data)
```

Response

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ],  
  "first_id": "msg_abc123",  
  "last_id": "msg_abc123",  
  "has_more": false  
}
```

Get input token counts



POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request.

Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with [conversation](#).

reasoning object Optional**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the [tools](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

➤ Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
{  
  object: "response.input_tokens"  
  input_tokens: 123  
}
```

Example request

```
from openai import OpenAI  
  
client = OpenAI()
```

```
response = client.responses.input_tokens.count(  
    model="gpt-5",  
    input="Tell me a joke."  
)  
print(response.input_tokens)
```

Response

```
{  
    "object": "response.input_tokens",  
    "input_tokens": 11  
}
```

The response object



background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

> Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

> Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

> Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

The length and order of items in the `output` array is dependent on the model's response.

Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

> Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more](#).

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or **temperature** but not both.

truncation string

The truncation strategy to use for the model response.

auto : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

disabled (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object

```
{  
  "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",  
  "object": "response",  
  "created_at": 1741476777,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {  
      "type": "message",  
      "id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",  
      "status": "completed",  
      "role": "assistant",  
      "content": [  
        {  
          "type": "output_text",  
          "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading",  
          "annotations": []  
        }  
      ]  
    }  
  ]
```

```
        },
      ],
      "parallel_tool_calls": true,
      "previous_response_id": null,
      "reasoning": {
        "effort": null,
        "summary": null
      },
      "store": true,
      "temperature": 1,
      "text": {
        "format": {
          "type": "text"
        }
      },
      "tool_choice": "auto",
      "tools": [],
      "top_p": 1,
      "truncation": "disabled",
      "usage": {
        "input_tokens": 328,
        "input_tokens_details": {
          "cached_tokens": 0
        },
        "output_tokens": 52,
        "output_tokens_details": {
          "reasoning_tokens": 0
        },
        "total_tokens": 380
      }
    }
  }
}
```

```
},  
  "user": null,  
  "metadata": {}  
}
```

The input item list



A list of Response items.

data array

A list of items used to generate this response.

> Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

OBJECT The input item list

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ],  
  "first_id": "msg_abc123",  
  "last_id": "msg_abc123",  
  "has_more": false  
}
```

The compacted response object



created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

> Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

> Show properties

OBJECT The compacted response object

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "output": [  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Summarize our launch checklist from last week."  
        }  
      ]  
    },  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."  
        }  
      ]  
    },  
    {  
      "type": "compaction",  
      "id": "cmp_001",  
      "encrypted_content": "encrypted-summary"  
    }  
  ]  
}
```

```
],  
  "created_at": 1731459200,  
  "usage": {  
    "input_tokens": 42897,  
    "output_tokens": 12000,  
    "total_tokens": 54912  
  }  
}
```

< PREVIOUS
Introduction

NEXT

Responses



OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

[Extend the models with tools](#)

Create a model response



POST <https://api.openai.com/v1/responses>

Creates a model response. Provide text or image inputs to generate text or JSON outputs. Have the model call your own custom code or use built-in tools like web search or file search to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

[› Show possible types](#)

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

`web_search_call.action.sources` : Include the sources of the web search tool call.

`code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.

`computer_call_output.output.image_url` : Include image urls from the computer call output.

`file_search_call.results` : Include the search results of the file search tool call.

`message.input_image.image_url` : Include image urls from the input message.

`message.output_text.logprobs` : Include logprobs with assistant messages.

`reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

[Text inputs and outputs](#)

[Image inputs](#)

[File inputs](#)

[Conversation state](#)

[Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Text input](#)

[Image input](#)

[File input](#)

[Web search](#)

[File search](#)

[Streaming](#)

[Functions](#)

[Reasoning](#)

Example request

```
from openai import OpenAI

client = OpenAI()

response = client.responses.create(
    model="gpt-4.1",
    input=[
        {
            "role": "user",
            "content": [
                { "type": "input_text", "text": "what is in this image?" },
                {
                    "type": "input_image",
                    "image_url": "https://upload.wikimedia.org/wikipedia/commons/thumb/d/dd/Gfp-wi"
                }
            ]
        }
    ]
)

print(response)
```

Response

```
{
    "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",
```

```
"object": "response",
"created_at": 1741476777,
"status": "completed",
"error": null,
"incomplete_details": null,
"instructions": null,
"max_output_tokens": null,
"model": "gpt-4.1-2025-04-14",
"output": [
  {
    "type": "message",
    "id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
    "status": "completed",
    "role": "assistant",
    "content": [
      {
        "type": "output_text",
        "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading through a lush green forest. The path is surrounded by tall trees and dense foliage, creating a natural and peaceful atmosphere. The lighting suggests it might be early morning or late afternoon, with sunlight filtering through the leaves. The overall scene is idyllic and captures a sense of tranquility in nature.",

        "annotations": []
      }
    ]
  },
  "parallel_tool_calls": true,
  "previous_response_id": null,
  "reasoning": {
    "effort": null,
    "summary": null
  }
},
```

```
"store": true,  
"temperature": 1.0,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [],  
"top_p": 1.0,  
"truncation": "disabled",  
"usage": {  
    "input_tokens": 328,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 52,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
    },  
    "total_tokens": 380  
},  
"user": null,  
"metadata": {}  
}
```

Get a model response



```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.retrieve("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
```

```
"incomplete_details": null,  
"instructions": null,  
"max_output_tokens": null,  
"model": "gpt-4o-2024-08-06",  
"output": [  
  {  
    "type": "message",  
    "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",  
    "status": "completed",  
    "role": "assistant",  
    "content": [  
      {  
        "type": "output_text",  
        "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn breaks",  
        "annotations": []  
      }  
    ]  
  },  
  ],  
  "parallel_tool_calls": true,  
  "previous_response_id": null,  
  "reasoning": {  
    "effort": null,  
    "summary": null  
  },  
  "store": true,  
  "temperature": 1.0,  
  "text": {  
    "format": {
```

```
        "type": "text"
    },
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
    "input_tokens": 32,
    "input_tokens_details": {
        "cached_tokens": 0
    },
    "output_tokens": 18,
    "output_tokens_details": {
        "reasoning_tokens": 0
    },
    "total_tokens": 50
},
"user": null,
"metadata": {}
}
```

Delete a model response



```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.delete("resp_123")
print(response)
```

Response

```
{  
  "id": "resp_6786a1bec27481909a17d673315b29f6",  
  "object": "response",  
  "deleted": true  
}
```

Cancel a response



```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A Response object.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.cancel("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4o-2024-08-06",
  "output": [
    {
      "type": "message",
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",
      "status": "completed",
```

```
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn breaks",
    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
  "input_tokens": 32,
```

```
"input_tokens_details": {  
    "cached_tokens": 0  
,  
    "output_tokens": 18,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
,  
        "total_tokens": 50  
,  
        "user": null,  
        "metadata": {}  
}
```

Compact a response



POST <https://api.openai.com/v1/responses/compact>

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

> Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

```
from openai import OpenAI

client = OpenAI()

compacted_response = client.responses.compact(
    model="gpt-5.1-codex-max",
    input=[
        {
            "role": "user",
            "content": "Create a simple landing page for a dog petting cafe.",
        },
        # All items returned from previous requests are included here, like reasoning, message, function_call, etc.
        {
            "id": "msg_001",
            "type": "message",
            "status": "completed",
            "content": [
                {
                    "type": "output_text",
                    "annotations": [],
                    "logprobs": [],
                    "text": "Below is a single file, ready-to-use landing page for a dog petting café:...",
                },
            ],
            "role": "assistant",
        },
    ]
)
```

```
# Pass the compacted_response.output as input to the next request
print(compacted_response)
```

Response

```
{
  "id": "resp_001",
  "object": "response.compaction",
  "created_at": 1764967971,
  "output": [
    {
      "id": "msg_000",
      "type": "message",
      "status": "completed",
      "content": [
        {
          "type": "input_text",
          "text": "Create a simple landing page for a dog petting cafe."
        }
      ],
      "role": "user"
    },
    {
      "id": "cmp_001",
      "type": "compaction",
      "encrypted_content": "gAAAAABpM0Yj-...="
    }
  ],
}
```

```
"usage": {  
    "input_tokens": 139,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 438,  
    "output_tokens_details": {  
        "reasoning_tokens": 64  
    },  
    "total_tokens": 577  
}  
}
```

List input items



GET https://api.openai.com/v1/responses/{response_id}/input_items

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

`asc` : Return the input items in ascending order.

`desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.input_items.list("resp_123")
print(response.data)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "msg_abc123",
      "type": "message",
      "role": "user",
      "content": [
        {
          "type": "input_text",
          "text": "Tell me a three sentence bedtime story about a unicorn."
        }
      ]
    },
    "first_id": "msg_abc123",
    "last_id": "msg_abc123",
    "has_more": false
  }
}
```

Get input token counts



POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages

in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

reasoning object Optional**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

› Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
{  
  object: "response.input_tokens"  
  input_tokens: 123  
}
```

Example request

```
from openai import OpenAI  
  
client = OpenAI()  
  
response = client.responses.input_tokens.count(  
    model="gpt-5",  
    input="Tell me a joke."  
)  
print(response.input_tokens)
```

Response

```
{  
  "object": "response.input_tokens",  
  "input_tokens": 11  
}
```

The response object



background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

› Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

› Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

› Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the model guide to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

The length and order of items in the `output` array is dependent on the model's response.

Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more](#).

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

auto : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

disabled (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object

```
{  
  "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",  
  "object": "response",  
  "created_at": 1741476777,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {
```

```
"type": "message",
"id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading through a lush green forest. The path is made of weathered planks and curves along a small stream. In the background, there are tall evergreen trees and a bright sky with wispy clouds. The overall atmosphere is peaceful and natural.",

    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1,
```

```
"truncation": "disabled",
"usage": {
  "input_tokens": 328,
  "input_tokens_details": {
    "cached_tokens": 0
  },
  "output_tokens": 52,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 380
},
"user": null,
"metadata": {}
}
```

The input item list



A list of Response items.

data array

A list of items used to generate this response.

> Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be **list**.

OBJECT The input item list

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
{
  "first_id": "msg_abc123",
  "last_id": "msg_abc123",
  "has_more": false
}
```

The compacted response object



created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

> Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

> Show properties

OBJECT The compacted response object

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "output": [  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Summarize our launch checklist from last week."  
        }  
      ]  
    },  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
{
  "type": "compaction",
  "id": "cmp_001",
  "encrypted_content": "encrypted-summary"
}
],
"created_at": 1731459200,
"usage": {
  "input_tokens": 42897,
  "output_tokens": 12000,
  "total_tokens": 54912
}
}
```

< PREVIOUS
Introduction

NEXT >

Responses



OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

[Extend the models with tools](#)

Create a model response



POST <https://api.openai.com/v1/responses>

Creates a model response. Provide text or image inputs to generate text or JSON outputs. Have the model call your own custom code or use built-in tools like web search or file search to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

[› Show possible types](#)

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

`web_search_call.action.sources` : Include the sources of the web search tool call.

`code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.

`computer_call_output.output.image_url` : Include image urls from the computer call output.

`file_search_call.results` : Include the search results of the file search tool call.

`message.input_image.image_url` : Include image urls from the input message.

`message.output_text.logprobs` : Include logprobs with assistant messages.

`reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

[Text inputs and outputs](#)

[Image inputs](#)

[File inputs](#)

[Conversation state](#)

[Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Text input](#)

[Image input](#)

[File input](#)

[Web search](#)

[File search](#)

[Streaming](#)

[Functions](#)

[Reasoning](#)

Example request

```
from openai import OpenAI

client = OpenAI()

response = client.responses.create(
    model="gpt-4.1",
    input=[
        {
            "role": "user",
            "content": [
                { "type": "input_text", "text": "what is in this file?" },
                {
                    "type": "input_file",
                    "file_url": "https://www.berkshirehathaway.com/letters/2024ltr.pdf"
                }
            ]
        }
    ]
)

print(response)
```

Response

```
{
    "id": "resp_686eef60237881a2bd1180bb8b13de430e34c516d176ff86",
```

```
"object": "response",
"created_at": 1752100704,
"status": "completed",
"background": false,
"error": null,
"incomplete_details": null,
"instructions": null,
"max_output_tokens": null,
"max_tool_calls": null,
"model": "gpt-4.1-2025-04-14",
"output": [
  {
    "id": "msg_686eef60d3e081a29283bdcbc4322fd90e34c516d176ff86",
    "type": "message",
    "status": "completed",
    "content": [
      {
        "type": "output_text",
        "annotations": [],
        "logprobs": [],
        "text": "The file seems to contain excerpts from a letter to the shareholders of Berkshir"
      }
    ],
    "role": "assistant"
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
```

```
"effort": null,  
"summary": null  
},  
"service_tier": "default",  
"store": true,  
"temperature": 1.0,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [],  
"top_logprobs": 0,  
"top_p": 1.0,  
"truncation": "disabled",  
"usage": {  
    "input_tokens": 8438,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 398,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
    },  
    "total_tokens": 8836  
},  
"user": null,
```

```
"metadata": {}
```

Get a model response



```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.retrieve("resp_123")
print(response)
```

Response

```
{  
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",  
  "object": "response",  
  "created_at": 1741386163,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {  
      "type": "message",  
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",  
      "status": "completed",  
      "role": "assistant",  
      "content": [  
        {  
          "type": "output_text",  
          "text": "Silent circuits hum,  \\nThoughts emerge in data streams–  \\nDigital dawn breaks",  
          "annotations": []  
        }  
      ]  
    },  
    ],  
  "parallel_tool_calls": true,  
  "previous_response_id": null,
```

```
"reasoning": {  
    "effort": null,  
    "summary": null  
},  
"store": true,  
"temperature": 1.0,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [],  
"top_p": 1.0,  
"truncation": "disabled",  
"usage": {  
    "input_tokens": 32,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 18,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
    },  
    "total_tokens": 50  
},  
"user": null,  
"metadata": {}  
}
```

Delete a model response



```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

```
from openai import OpenAI  
client = OpenAI()
```

```
response = client.responses.delete("resp_123")
print(response)
```

Response

```
{
  "id": "resp_6786a1bec27481909a17d673315b29f6",
  "object": "response",
  "deleted": true
}
```

Cancel a response



```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A [Response](#) object.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.cancel("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4o-2024-08-06",
```

```
"output": [
  {
    "type": "message",
    "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",
    "status": "completed",
    "role": "assistant",
    "content": [
      {
        "type": "output_text",
        "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn break",
        "annotations": []
      }
    ]
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
```

```
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
  "input_tokens": 32,
  "input_tokens_details": {
    "cached_tokens": 0
  },
  "output_tokens": 18,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 50
},
"user": null,
"metadata": {}
```

Compact a response



POST <https://api.openai.com/v1/responses/compact>

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

```
from openai import OpenAI

client = OpenAI()

compacted_response = client.responses.compact(
    model="gpt-5.1-codex-max",
    input=[
        {
            "role": "user",
            "content": "Create a simple landing page for a dog petting cafe.",
        },
        # All items returned from previous requests are included here, like reasoning, message, function
        {
            "id": "msg_001",
            "type": "message",
            "status": "completed",
            "content": [
                {
                    "type": "output_text",
                    "annotations": [],
                    "logprobs": [],
                    "text": "Below is a single file, ready-to-use landing page for a dog petting café:..."
                },
            ],
            "role": "assistant",
        },
    ]
)
```

```
)  
# Pass the compacted_response.output as input to the next request  
print(compacted_response)
```

Response

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "created_at": 1764967971,  
  "output": [  
    {  
      "id": "msg_000",  
      "type": "message",  
      "status": "completed",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Create a simple landing page for a dog petting cafe."  
        }  
      ],  
      "role": "user"  
    },  
    {  
      "id": "cmp_001",  
      "type": "compaction",  
      "encrypted_content": "gAAAAABpM0Yj-...="  
    }  
  ]}
```

```
],
  "usage": {
    "input_tokens": 139,
    "input_tokens_details": {
      "cached_tokens": 0
    },
    "output_tokens": 438,
    "output_tokens_details": {
      "reasoning_tokens": 64
    },
    "total_tokens": 577
  }
}
```

List input items



GET https://api.openai.com/v1/responses/{response_id}/input_items

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

`asc` : Return the input items in ascending order.

`desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.input_items.list("resp_123")
print(response.data)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "msg_abc123",
      "type": "message",
      "role": "user",
      "content": [
        {
          "type": "input_text",
          "text": "Tell me a three sentence bedtime story about a unicorn."
        }
      ]
    }
  ],
}
```

```
"first_id": "msg_abc123",
"last_id": "msg_abc123",
"has_more": false
}
```

Get input token counts



POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to **input_items** for this response request.

Input items and output items from this response are automatically added to this conversation after this response completes.

> Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

> Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

reasoning object Optional**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

› Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
{  
  object: "response.input_tokens"
```

```
    input_tokens: 123  
}
```

Example request

```
from openai import OpenAI  
  
client = OpenAI()  
  
response = client.responses.input_tokens.count(  
    model="gpt-5",  
    input="Tell me a joke."  
)  
print(response.input_tokens)
```

Response

```
{  
    "object": "response.input_tokens",  
    "input_tokens": 11  
}
```

The response object



background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

› Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

› Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

› Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the model guide to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

The length and order of items in the `output` array is dependent on the model's response.

Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more](#).

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

auto : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

disabled (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object

```
{  
  "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",  
  "object": "response",  
  "created_at": 1741476777,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {
```

```
"type": "message",
"id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading through a lush green forest. The path is made of light-colored wood planks and curves slightly to the right. It's surrounded by tall trees with dense foliage. In the background, there are rolling hills covered in greenery under a clear blue sky with a few wispy clouds. The overall atmosphere is peaceful and natural, suggesting a rural or park setting. There are no people or animals visible in the scene.",

    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1,
```

```
"truncation": "disabled",
"usage": {
  "input_tokens": 328,
  "input_tokens_details": {
    "cached_tokens": 0
  },
  "output_tokens": 52,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 380
},
"user": null,
"metadata": {}
}
```

The input item list



A list of Response items.

data array

A list of items used to generate this response.

> Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be **list**.

OBJECT The input item list

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
      ],
    },
  ],
  "first_id": "msg_abc123",
  "last_id": "msg_abc123",
  "has_more": false
}
```

The compacted response object



created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

> Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

> Show properties

OBJECT The compacted response object

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "output": [  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Summarize our launch checklist from last week."  
        }  
      ]  
    },  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
{
  "type": "compaction",
  "id": "cmp_001",
  "encrypted_content": "encrypted-summary"
}
],
"created_at": 1731459200,
"usage": {
  "input_tokens": 42897,
  "output_tokens": 12000,
  "total_tokens": 54912
}
}
```

< PREVIOUS
Introduction

NEXT

Responses



OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

[Extend the models with tools](#)

Create a model response



POST <https://api.openai.com/v1/responses>

Creates a model response. Provide text or image inputs to generate text or JSON outputs. Have the model call your own custom code or use built-in tools like web search or file search to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

[› Show possible types](#)

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

`web_search_call.action.sources` : Include the sources of the web search tool call.

`code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.

`computer_call_output.output.image_url` : Include image urls from the computer call output.

`file_search_call.results` : Include the search results of the file search tool call.

`message.input_image.image_url` : Include image urls from the input message.

`message.output_text.logprobs` : Include logprobs with assistant messages.

`reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

[Text inputs and outputs](#)

[Image inputs](#)

[File inputs](#)

[Conversation state](#)

[Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Text input](#)

[Image input](#)

[File input](#)

Web search

[File search](#)

[Streaming](#)

[Functions](#)

[Reasoning](#)

Example request

```
from openai import OpenAI

client = OpenAI()

response = client.responses.create(
    model="gpt-4.1",
    tools=[{"type": "web_search_preview"}],
    input="What was a positive news story from today?",
)

print(response)
```

Response

```
{
  "id": "resp_67ccf18ef5fc8190b16dbe19bc54e5f087bb177ab789d5c",
  "object": "response",
  "created_at": 1741484430,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4.1-2025-04-14",
  "output": [
    {
      "type": "web_search_call",
```

```
"id": "ws_67ccf18f64008190a39b619f4c8455ef087bb177ab789d5c",
  "status": "completed"
},
{
  "type": "message",
  "id": "msg_67ccf190ca3881909d433c50b1f6357e087bb177ab789d5c",
  "status": "completed",
  "role": "assistant",
  "content": [
    {
      "type": "output_text",
      "text": "As of today, March 9, 2025, one notable positive news story...",
      "annotations": [
        {
          "type": "url_citation",
          "start_index": 442,
          "end_index": 557,
          "url": "https://.../?utm_source=chatgpt.com",
          "title": "..."
        },
        {
          "type": "url_citation",
          "start_index": 962,
          "end_index": 1077,
          "url": "https://.../?utm_source=chatgpt.com",
          "title": "..."
        },
        {
          "type": "url_citation",
          "start_index": 1492,
          "end_index": 1607,
          "url": "https://.../?utm_source=chatgpt.com",
          "title": "..."
        }
      ]
    }
  ]
}
```

```
        "start_index": 1336,
        "end_index": 1451,
        "url": "https://.../?utm_source=chatgpt.com",
        "title": "..."

    }
]
}
]
},
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
    "effort": null,
    "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
    "format": {
        "type": "text"
    }
},
"tool_choice": "auto",
"tools": [
    {
        "type": "web_search_preview",
        "domains": [],
        "search_context_size": "medium",
    }
]
```

```
"user_location": {  
    "type": "approximate",  
    "city": null,  
    "country": "US",  
    "region": null,  
    "timezone": null  
}  
}  
],  
"top_p": 1.0,  
"truncation": "disabled",  
"usage": {  
    "input_tokens": 328,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 356,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
    },  
    "total_tokens": 684  
},  
"user": null,  
"metadata": {}  
}
```

Get a model response



```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.retrieve("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
```

```
"incomplete_details": null,  
"instructions": null,  
"max_output_tokens": null,  
"model": "gpt-4o-2024-08-06",  
"output": [  
  {  
    "type": "message",  
    "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",  
    "status": "completed",  
    "role": "assistant",  
    "content": [  
      {  
        "type": "output_text",  
        "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn breaks",  
        "annotations": []  
      }  
    ]  
  },  
  ],  
  "parallel_tool_calls": true,  
  "previous_response_id": null,  
  "reasoning": {  
    "effort": null,  
    "summary": null  
  },  
  "store": true,  
  "temperature": 1.0,  
  "text": {  
    "format": {
```

```
        "type": "text"
    },
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
    "input_tokens": 32,
    "input_tokens_details": {
        "cached_tokens": 0
    },
    "output_tokens": 18,
    "output_tokens_details": {
        "reasoning_tokens": 0
    },
    "total_tokens": 50
},
"user": null,
"metadata": {}
}
```

Delete a model response



```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.delete("resp_123")
print(response)
```

Response

```
{  
  "id": "resp_6786a1bec27481909a17d673315b29f6",  
  "object": "response",  
  "deleted": true  
}
```

Cancel a response



```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A Response object.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.cancel("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4o-2024-08-06",
  "output": [
    {
      "type": "message",
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",
      "status": "completed",
```

```
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn breaks",
    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
  "input_tokens": 32,
```

```
"input_tokens_details": {  
    "cached_tokens": 0  
,  
    "output_tokens": 18,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
,  
        "total_tokens": 50  
,  
        "user": null,  
        "metadata": {}  
}
```

Compact a response



```
POST https://api.openai.com/v1/responses/compact
```

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

> Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

```
from openai import OpenAI

client = OpenAI()

compacted_response = client.responses.compact(
    model="gpt-5.1-codex-max",
    input=[
        {
            "role": "user",
            "content": "Create a simple landing page for a dog petting cafe.",
        },
        # All items returned from previous requests are included here, like reasoning, message, function_call, etc.
        {
            "id": "msg_001",
            "type": "message",
            "status": "completed",
            "content": [
                {
                    "type": "output_text",
                    "annotations": [],
                    "logprobs": [],
                    "text": "Below is a single file, ready-to-use landing page for a dog petting café:...",
                },
            ],
            "role": "assistant",
        },
    ]
)
```

```
# Pass the compacted_response.output as input to the next request
print(compacted_response)
```

Response

```
{
  "id": "resp_001",
  "object": "response.compaction",
  "created_at": 1764967971,
  "output": [
    {
      "id": "msg_000",
      "type": "message",
      "status": "completed",
      "content": [
        {
          "type": "input_text",
          "text": "Create a simple landing page for a dog petting cafe."
        }
      ],
      "role": "user"
    },
    {
      "id": "cmp_001",
      "type": "compaction",
      "encrypted_content": "gAAAAABpM0Yj-...="
    }
  ],
}
```

```
"usage": {  
    "input_tokens": 139,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 438,  
    "output_tokens_details": {  
        "reasoning_tokens": 64  
    },  
    "total_tokens": 577  
}  
}
```

List input items



GET https://api.openai.com/v1/responses/{response_id}/input_items

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

`asc` : Return the input items in ascending order.

`desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.input_items.list("resp_123")
print(response.data)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "msg_abc123",
      "type": "message",
      "role": "user",
      "content": [
        {
          "type": "input_text",
          "text": "Tell me a three sentence bedtime story about a unicorn."
        }
      ]
    }
  ],
  "first_id": "msg_abc123",
  "last_id": "msg_abc123",
  "has_more": false
}
```

Get input token counts



POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages

in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

reasoning object Optional**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

› Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
{  
  object: "response.input_tokens"  
  input_tokens: 123  
}
```

Example request

```
from openai import OpenAI  
  
client = OpenAI()  
  
response = client.responses.input_tokens.count(  
    model="gpt-5",  
    input="Tell me a joke."  
)  
print(response.input_tokens)
```

Response

```
{  
  "object": "response.input_tokens",  
  "input_tokens": 11  
}
```

The response object



background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

› Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

› Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

› Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the model guide to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

The length and order of items in the `output` array is dependent on the model's response.

Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more](#).

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

auto : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

disabled (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object

```
{  
  "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",  
  "object": "response",  
  "created_at": 1741476777,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {
```

```
"type": "message",
"id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading through a lush green forest. The path is made of weathered planks and curves along a small stream. In the background, there are tall evergreen trees and a bright sky with wispy clouds. The overall atmosphere is peaceful and natural.",

    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1,
```

```
"truncation": "disabled",
"usage": {
  "input_tokens": 328,
  "input_tokens_details": {
    "cached_tokens": 0
  },
  "output_tokens": 52,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 380
},
"user": null,
"metadata": {}
}
```

The input item list



A list of Response items.

data array

A list of items used to generate this response.

> Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be **list**.

OBJECT The input item list

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
{
  "first_id": "msg_abc123",
  "last_id": "msg_abc123",
  "has_more": false
}
```

The compacted response object



created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

> Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

> Show properties

OBJECT The compacted response object

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "output": [  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Summarize our launch checklist from last week."  
        }  
      ]  
    },  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
{
  "type": "compaction",
  "id": "cmp_001",
  "encrypted_content": "encrypted-summary"
}
],
"created_at": 1731459200,
"usage": {
  "input_tokens": 42897,
  "output_tokens": 12000,
  "total_tokens": 54912
}
}
```

< PREVIOUS
Introduction

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Responses

OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

- [Quickstart](#)
- [Text inputs and outputs](#)
- [Image inputs](#)
- [Structured Outputs](#)
- [Function calling](#)
- [Conversation state](#)

- [Extend the models with tools](#)
-

Create a model response

POST <https://api.openai.com/v1/responses>

Creates a model response. Provide [text](#) or [image](#) inputs to generate [text](#) or [JSON](#) outputs.

Have the model call your own [custom code](#) or use built-in [tools](#) like [web search](#) or [file search](#) to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to [input_items](#) for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

- `web_search_call.action.sources` : Include the sources of the web search tool call.
- `code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.
- `computer_call_output.output.image_url` : Include image urls from the computer call output.
- `file_search_call.results` : Include the search results of the file search tool call.
- `message.input_image.image_url` : Include image urls from the input message.
- `message.output_text.logprobs` : Include logprobs with assistant messages.
- `reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

- [Text inputs and outputs](#)
- [Image inputs](#)

- [File inputs](#)
- [Conversation state](#)
- [Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the

value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

› Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

- [Text inputs and outputs](#)
- [Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the [tools](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the [tool_choice](#) parameter.

We support the following categories of tools:

- **Built-in tools:** Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).
- **MCP Tools:** Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).
- **Function calls (custom tools):** Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

- `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.
 - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.
-

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

Image input File input Web search **File search** Streaming Functions Reasoning

Example request

python ⚙️

```
1 from openai import OpenAI
2
3 client = OpenAI()
4
5 response = client.responses.create(
6     model="gpt-4.1",
7     tools=[{
8         "type": "file_search",
9         "vector_store_ids": ["vs_1234567890"],
10        "max_num_results": 20
11    }],
12    input="What are the attributes of an ancient brown dragon?",
13 )
14
15 print(response)
```

Response

🔗

```
1 {
2     "id": "resp_67ccf4c55fc48190b71bd0463ad3306d09504fb6872380d7",
3     "object": "response",
4     "created_at": 1741485253,
5     "status": "completed",
6     "error": null,
```

```
7  "incomplete_details": null,
8  "instructions": null,
9  "max_output_tokens": null,
10 "model": "gpt-4.1-2025-04-14",
11 "output": [
12   {
13     "type": "file_search_call",
14     "id": "fs_67ccf4c63cd08190887ef6464ba5681609504fb6872380d7",
15     "status": "completed",
16     "queries": [
17       "attributes of an ancient brown dragon"
18     ],
19     "results": null
20   },
21   {
22     "type": "message",
23     "id": "msg_67ccf4c93e5c81909d595b369351a9d309504fb6872380d7",
24     "status": "completed",
25     "role": "assistant",
26     "content": [
27       {
28         "type": "output_text",
29         "text": "The attributes of an ancient brown dragon include...",
30         "annotations": [
31           {
32             "type": "file_citation",
33             "index": 320,
34             "file_id": "file-4wDz5b167pAf72nx1h9eiN",
35             "filename": "dragons.pdf"
```

```
36     },
37     {
38         "type": "file_citation",
39         "index": 576,
40         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
41         "filename": "dragons.pdf"
42     },
43     {
44         "type": "file_citation",
45         "index": 815,
46         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
47         "filename": "dragons.pdf"
48     },
49     {
50         "type": "file_citation",
51         "index": 815,
52         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
53         "filename": "dragons.pdf"
54     },
55     {
56         "type": "file_citation",
57         "index": 1030,
58         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
59         "filename": "dragons.pdf"
60     },
61     {
62         "type": "file_citation",
63         "index": 1030,
64         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
```

```
65         "filename": "dragons.pdf"
66     },
67     {
68         "type": "file_citation",
69         "index": 1156,
70         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
71         "filename": "dragons.pdf"
72     },
73     {
74         "type": "file_citation",
75         "index": 1225,
76         "file_id": "file-4wDz5b167pAf72nx1h9eiN",
77         "filename": "dragons.pdf"
78     }
79 ]
80 }
81 ]
82 }
83 ],
84 "parallel_tool_calls": true,
85 "previous_response_id": null,
86 "reasoning": {
87     "effort": null,
88     "summary": null
89 },
90 "store": true,
91 "temperature": 1.0,
92 "text": {
93     "format": {
```

```
94     "type": "text"
95   }
96 },
97 "tool_choice": "auto",
98 "tools": [
99   {
100     "type": "file_search",
101     "filters": null,
102     "max_num_results": 20,
103     "ranking_options": {
104       "ranker": "auto",
105       "score_threshold": 0.0
106     },
107     "vector_store_ids": [
108       "vs_1234567890"
109     ]
110   }
111 ],
112 "top_p": 1.0,
113 "truncation": "disabled",
114 "usage": {
115   "input_tokens": 18307,
116   "input_tokens_details": {
117     "cached_tokens": 0
118   },
119   "output_tokens": 348,
120   "output_tokens_details": {
121     "reasoning_tokens": 0
122   },
123 }
```

```
123     "total_tokens": 18655
124 },
125 "user": null,
126 "metadata": {}
127 }
```

Get a model response

```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

python 

```
1 from openai import OpenAI
2 client = OpenAI()
3
```

```
4 response = client.responses.retrieve("resp_123")
5 print(response)
```

Response



```
1 {
2     "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
3     "object": "response",
4     "created_at": 1741386163,
5     "status": "completed",
6     "error": null,
7     "incomplete_details": null,
8     "instructions": null,
9     "max_output_tokens": null,
10    "model": "gpt-4o-2024-08-06",
11    "output": [
12        {
13            "type": "message",
14            "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",
15            "status": "completed",
16            "role": "assistant",
17            "content": [
18                {
19                    "type": "output_text",
20                    "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigit",
21                    "annotations": []
22                }
23            ]
24        }
25    ]
26 }
```

```
24     },
25   ],
26   "parallel_tool_calls": true,
27   "previous_response_id": null,
28   "reasoning": {
29     "effort": null,
30     "summary": null
31   },
32   "store": true,
33   "temperature": 1.0,
34   "text": {
35     "format": {
36       "type": "text"
37     }
38   },
39   "tool_choice": "auto",
40   "tools": [],
41   "top_p": 1.0,
42   "truncation": "disabled",
43   "usage": {
44     "input_tokens": 32,
45     "input_tokens_details": {
46       "cached_tokens": 0
47     },
48     "output_tokens": 18,
49     "output_tokens_details": {
50       "reasoning_tokens": 0
51     },
52     "total_tokens": 50
53   }
54 }
```

```
53 },
54 "user": null,
55 "metadata": {}
56 }
```

Delete a model response

```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.delete("resp_123")
5 print(response)
```

Response

🔗

```
1 {
2   "id": "resp_6786a1bec27481909a17d673315b29f6",
3   "object": "response",
4   "deleted": true
5 }
```

Cancel a response

```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the [background](#) parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A [Response](#) object.

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.cancel("resp_123")
5 print(response)
```

Response

Copy

```
1 {
2     "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
```

```
3     "object": "response",
4     "created_at": 1741386163,
5     "status": "completed",
6     "error": null,
7     "incomplete_details": null,
8     "instructions": null,
9     "max_output_tokens": null,
10    "model": "gpt-4o-2024-08-06",
11    "output": [
12      {
13        "type": "message",
14        "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",
15        "status": "completed",
16        "role": "assistant",
17        "content": [
18          {
19            "type": "output_text",
20            "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigit",
21            "annotations": []
22          }
23        ]
24      },
25    ],
26    "parallel_tool_calls": true,
27    "previous_response_id": null,
28    "reasoning": {
29      "effort": null,
30      "summary": null
31    },
32  },
```

```
32 "store": true,
33 "temperature": 1.0,
34 "text": {
35     "format": {
36         "type": "text"
37     }
38 },
39 "tool_choice": "auto",
40 "tools": [],
41 "top_p": 1.0,
42 "truncation": "disabled",
43 "usage": {
44     "input_tokens": 32,
45     "input_tokens_details": {
46         "cached_tokens": 0
47     },
48     "output_tokens": 18,
49     "output_tokens_details": {
50         "reasoning_tokens": 0
51     },
52     "total_tokens": 50
53 },
54 "user": null,
55 "metadata": {}
56 }
```

Compact a response

POST <https://api.openai.com/v1/responses/compact>

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with [conversation](#).

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4
5 compacted_response = client.responses.compact(
6     model="gpt-5.1-codex-max",
7     input=[
8         {
9             "role": "user",
10            "content": "Create a simple landing page for a dog petting cafe.",
11        },
12        # All items returned from previous requests are included here, like reasoning, n
13        {
14            "id": "msg_001",
```

```
15     "type": "message",
16     "status": "completed",
17     "content": [
18     {
19         "type": "output_text",
20         "annotations": [],
21         "logprobs": [],
22         "text": "Below is a single file, ready-to-use landing page for a dog pet"
23     },
24     ],
25     "role": "assistant",
26 },
27 ]
28 )
29 # Pass the compacted_response.output as input to the next request
30 print(compacted_response)
```

Response



```
1 {
2     "id": "resp_001",
3     "object": "response.compaction",
4     "created_at": 1764967971,
5     "output": [
6     {
7         "id": "msg_000",
8         "type": "message",
9         "status": "completed",
```

```
10     "content": [
11         {
12             "type": "input_text",
13             "text": "Create a simple landing page for a dog petting cafe."
14         }
15     ],
16     "role": "user"
17 },
18 {
19     "id": "cmp_001",
20     "type": "compaction",
21     "encrypted_content": "gAAAAABpM0Yj-...="
22 }
23 ],
24 "usage": {
25     "input_tokens": 139,
26     "input_tokens_details": {
27         "cached_tokens": 0
28     },
29     "output_tokens": 438,
30     "output_tokens_details": {
31         "reasoning_tokens": 64
32     },
33     "total_tokens": 577
34 }
35 }
```

List input items

```
GET https://api.openai.com/v1/responses/{response_id}/input_items
```

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

- `asc` : Return the input items in ascending order.
- `desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.input_items.list("resp_123")
5 print(response.data)
```

Response

🔗

```
1 {
2     "object": "list",
```

```
3 "data": [
4   {
5     "id": "msg_abc123",
6     "type": "message",
7     "role": "user",
8     "content": [
9       {
10        "type": "input_text",
11        "text": "Tell me a three sentence bedtime story about a unicorn."
12      }
13    ]
14  }
15 ],
16 "first_id": "msg_abc123",
17 "last_id": "msg_abc123",
18 "has_more": false
19 }
```

Get input token counts

POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to

`input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with

`previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with [conversation](#).

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

- [Text inputs and outputs](#)
- [Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the [tools](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

› Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
1 {  
2   object: "response.input_tokens"  
3   input_tokens: 123  
4 }
```



Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4
5 response = client.responses.input_tokens.count(
6     model="gpt-5",
7     input="Tell me a joke."
8 )
9 print(response.input_tokens)
```

Response

🔗

```
1 {
2     "object": "response.input_tokens",
3     "input_tokens": 11
4 }
```

The response object

background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

› Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

› Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

› Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the model guide to browse and

compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

- The length and order of items in the `output` array is dependent on the model's response.
- Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to 'flex' or 'priority', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed` , `failed` , `in_progress` , `cancelled` , `queued` , or `incomplete` .

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

- [Text inputs and outputs](#)
- [Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the [`tools`](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the [`tool_choice`](#) parameter.

We support the following categories of tools:

- **Built-in tools:** Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).
- **MCP Tools:** Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).
- **Function calls (custom tools):** Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

- `auto`: If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.
 - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.
-

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more.](#)

OBJECT The response object

```
1  {
2      "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",
3      "object": "response",
4      "created_at": 1741476777,
5      "status": "completed",
6      "error": null,
7      "incomplete_details": null,
8      "instructions": null,
9      "max_output_tokens": null,
10     "model": "gpt-4o-2024-08-06",
11     "output": [
12         {
13             "type": "message",
14             "id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
15             "status": "completed",
16             "role": "assistant",
17             "content": [
18                 {
19                     "type": "output_text",
20                     "text": "The image depicts a scenic landscape with a wooden boardwalk or p
21                     "annotations": []
22                 }
23             ]
24         }
25     ]
26 }
```

```
24     },
25   ],
26   "parallel_tool_calls": true,
27   "previous_response_id": null,
28   "reasoning": {
29     "effort": null,
30     "summary": null
31   },
32   "store": true,
33   "temperature": 1,
34   "text": {
35     "format": {
36       "type": "text"
37     }
38   },
39   "tool_choice": "auto",
40   "tools": [],
41   "top_p": 1,
42   "truncation": "disabled",
43   "usage": {
44     "input_tokens": 328,
45     "input_tokens_details": {
46       "cached_tokens": 0
47     },
48     "output_tokens": 52,
49     "output_tokens_details": {
50       "reasoning_tokens": 0
51     },
52     "total_tokens": 380

```

```
53 },
54 "user": null,
55 "metadata": {}
56 }
```

The input item list

A list of Response items.

data array

A list of items used to generate this response.

› Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

OBJECT The input item list



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "id": "msg_abc123",
6              "type": "message",
7              "role": "user",
8              "content": [
9                  {
10                     "type": "input_text",
11                     "text": "Tell me a three sentence bedtime story about a unicorn."
12                 }
13             ]
14         }
15     ],
16     "first_id": "msg_abc123",
17     "last_id": "msg_abc123",
18     "has_more": false
19 }
```

The compacted response object

created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

› Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

OBJECT The compacted response object



```
1  {
2      "id": "resp_001",
3      "object": "response.compaction",
4      "output": [
5          {
6              "type": "message",
7              "role": "user",
8              "content": [
9                  {
10                     "type": "input_text",
11                     "text": "Summarize our launch checklist from last week."
12                 }
13             ]
14         },
15         {
16             "type": "message",
17             "role": "user",
18             "content": [
19                 {
20                     "type": "input_text",
21                     "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."
22                 }
23             ]
24         },
25     {
```

```
26     "type": "compaction",
27     "id": "cmp_001",
28     "encrypted_content": "encrypted-summary"
29   }
30 ],
31 "created_at": 1731459200,
32 "usage": {
33   "input_tokens": 42897,
34   "output_tokens": 12000,
35   "total_tokens": 54912
36 }
37 }
```

PREVIOUS
< **Introduction**

NEXT
Conversations >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Responses

OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

- [Quickstart](#)
- [Text inputs and outputs](#)
- [Image inputs](#)
- [Structured Outputs](#)
- [Function calling](#)
- [Conversation state](#)

- [Extend the models with tools](#)
-

Create a model response

POST <https://api.openai.com/v1/responses>

Creates a model response. Provide [text](#) or [image](#) inputs to generate [text](#) or [JSON](#) outputs.

Have the model call your own [custom code](#) or use built-in [tools](#) like [web search](#) or [file search](#) to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to [input_items](#) for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

- `web_search_call.action.sources` : Include the sources of the web search tool call.
- `code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.
- `computer_call_output.output.image_url` : Include image urls from the computer call output.
- `file_search_call.results` : Include the search results of the file search tool call.
- `message.input_image.image_url` : Include image urls from the input message.
- `message.output_text.logprobs` : Include logprobs with assistant messages.
- `reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

- [Text inputs and outputs](#)
- [Image inputs](#)

- [File inputs](#)
- [Conversation state](#)
- [Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the

value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

› Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

- [Text inputs and outputs](#)
- [Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the [tools](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the [tool_choice](#) parameter.

We support the following categories of tools:

- **Built-in tools:** Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).
- **MCP Tools:** Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).
- **Function calls (custom tools):** Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

- `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.
- `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Image input](#)[File input](#)[Web search](#)[File search](#)[Streaming](#)[Functions](#)[Reasoning](#)

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4
5 response = client.responses.create(
6     model="gpt-4.1",
7     instructions="You are a helpful assistant.",
8     input="Hello!",
9     stream=True
10 )
11
12 for event in response:
13     print(event)
```

Response

🔗

```
1 event: response.created
2 data: {"type": "response.created", "response": {"id": "resp_67c9fdcecf488190bdd9a0409de3", "object": "response", "status": "created", "type": "response"}, "model": "gpt-4.1", "object": "event", "type": "response.created"}
3
4 event: response.in_progress
5 data: {"type": "response.in_progress", "response": {"id": "resp_67c9fdcecf488190bdd9a0409de3", "object": "response", "status": "in_progress", "type": "response"}, "model": "gpt-4.1", "object": "event", "type": "response.in_progress"}
6
7 event: response.output_item.added
8 data: {"type": "response.output_item.added", "output_index": 0, "item": {"id": "msg_67c9fdcecf488190bdd9a0409de3", "object": "message", "role": "assistant", "content": "Hello!", "type": "message"}, "model": "gpt-4.1", "object": "event", "type": "response.output_item.added"}  
9
```

```
9
10 event: response.content_part.added
11 data: {"type": "response.content_part.added", "item_id": "msg_67c9fdcf37fc8190ba82116e33f1d4544"}
12
13 event: response.output_text.delta
14 data: {"type": "response.output_text.delta", "item_id": "msg_67c9fdcf37fc8190ba82116e33f1d4544"}
15
16 ...
17
18 event: response.output_text.done
19 data: {"type": "response.output_text.done", "item_id": "msg_67c9fdcf37fc8190ba82116e33f1d4544"}
20
21 event: response.content_part.done
22 data: {"type": "response.content_part.done", "item_id": "msg_67c9fdcf37fc8190ba82116e33f1d4544"}
23
24 event: response.output_item.done
25 data: {"type": "response.output_item.done", "output_index": 0, "item": {"id": "msg_67c9fdcecf488190bdd9a0409c54544"}, "text": "Hello, world!"}
26
27 event: response.completed
28 data: {"type": "response.completed", "response": {"id": "resp_67c9fdcecf488190bdd9a0409c54544", "object": "response", "status": "success", "content": "Hello, world!", "created": 1691985200, "model": "text-davinci-003", "usage": {"prompt_tokens": 1, "completion_tokens": 1, "total_tokens": 2}, "logprobs": 0, "stop_sequences": null, "error": null}}
```

Get a model response

```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.retrieve("resp_123")
5 print(response)
```

Response

🔗

```
1 {
2     "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
3     "object": "response",
4     "created_at": 1741386163,
5     "status": "completed",
```

```
6     "error": null,
7     "incomplete_details": null,
8     "instructions": null,
9     "max_output_tokens": null,
10    "model": "gpt-4o-2024-08-06",
11    "output": [
12      {
13        "type": "message",
14        "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",
15        "status": "completed",
16        "role": "assistant",
17        "content": [
18          {
19            "type": "output_text",
20            "text": "Silent circuits hum, \nThoughts emerge in data streams- \nDigit",
21            "annotations": []
22          }
23        ]
24      }
25    ],
26    "parallel_tool_calls": true,
27    "previous_response_id": null,
28    "reasoning": {
29      "effort": null,
30      "summary": null
31    },
32    "store": true,
33    "temperature": 1.0,
34    "text": {
```

```
35     "format": {
36         "type": "text"
37     },
38 },
39 "tool_choice": "auto",
40 "tools": [],
41 "top_p": 1.0,
42 "truncation": "disabled",
43 "usage": {
44     "input_tokens": 32,
45     "input_tokens_details": {
46         "cached_tokens": 0
47     },
48     "output_tokens": 18,
49     "output_tokens_details": {
50         "reasoning_tokens": 0
51     },
52     "total_tokens": 50
53 },
54 "user": null,
55 "metadata": {}
56 }
```

Delete a model response

```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.delete("resp_123")
5 print(response)
```

Response



```
1 {
2   "id": "resp_6786a1bec27481909a17d673315b29f6",
3   "object": "response",
4   "deleted": true
5 }
```

Cancel a response

```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

`response_id` string Required

The ID of the response to cancel.

Returns

A Response object.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.cancel("resp_123")
5 print(response)
```

Response

🔗

```
1 {
2     "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
3     "object": "response",
4     "created_at": 1741386163,
5     "status": "completed",
6     "error": null,
7     "incomplete_details": null,
8     "instructions": null,
9     "max_output_tokens": null,
10    "model": "gpt-4o-2024-08-06",
11    "output": [
12        {
13            "type": "message",
```

```
14     "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",
15     "status": "completed",
16     "role": "assistant",
17     "content": [
18         {
19             "type": "output_text",
20             "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigit
21             "annotations": []
22         }
23     ]
24 },
25 ],
26 "parallel_tool_calls": true,
27 "previous_response_id": null,
28 "reasoning": {
29     "effort": null,
30     "summary": null
31 },
32 "store": true,
33 "temperature": 1.0,
34 "text": {
35     "format": {
36         "type": "text"
37     }
38 },
39 "tool_choice": "auto",
40 "tools": [],
41 "top_p": 1.0,
42 "truncation": "disabled",
```

```
43 "usage": {
44     "input_tokens": 32,
45     "input_tokens_details": {
46         "cached_tokens": 0
47     },
48     "output_tokens": 18,
49     "output_tokens_details": {
50         "reasoning_tokens": 0
51     },
52     "total_tokens": 50
53 },
54 "user": null,
55 "metadata": {}
56 }
```

Compact a response

POST <https://api.openai.com/v1/responses/compact>

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

python ⚖️

```
1 from openai import OpenAI
2
3 client = OpenAI()
4
5 compacted_response = client.responses.compact(
6     model="gpt-5.1-codex-max",
7     input=[
8         {
9             "role": "user",
10            "content": "Create a simple landing page for a dog petting cafe.",
11        },
12        # All items returned from previous requests are included here, like reasoning, n
13        {
14            "id": "msg_001",
15            "type": "message",
16            "status": "completed",
17            "content": [
18                {
19                    "type": "output_text",
20                    "annotations": [],
21                    "logprobs": [],
22                    "text": "Below is a single file, ready-to-use landing page for a dog pet
23                },
24            ],
25            "role": "assistant",
```

```
26     },
27   ]
28 )
29 # Pass the compacted_response.output as input to the next request
30 print(compacted_response)
```

Response



```
1  {
2    "id": "resp_001",
3    "object": "response.compaction",
4    "created_at": 1764967971,
5    "output": [
6      {
7        "id": "msg_000",
8        "type": "message",
9        "status": "completed",
10       "content": [
11         {
12           "type": "input_text",
13           "text": "Create a simple landing page for a dog petting cafe."
14         }
15       ],
16       "role": "user"
17     },
18     {
19       "id": "cmp_001",
20       "type": "compaction",
```

```
21     "encrypted_content": "gAAAAABpM0Yj-...="
22   }
23 ],
24 "usage": {
25   "input_tokens": 139,
26   "input_tokens_details": {
27     "cached_tokens": 0
28   },
29   "output_tokens": 438,
30   "output_tokens_details": {
31     "reasoning_tokens": 64
32   },
33   "total_tokens": 577
34 }
35 }
```

List input items

```
GET https://api.openai.com/v1/responses/{response_id}/input_items
```

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

- `asc` : Return the input items in ascending order.
- `desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.input_items.list("resp_123")
5 print(response.data)
```

Response

🔗

```
1 {
2     "object": "list",
3     "data": [
4         {
5             "id": "msg_abc123",
6             "type": "message",
7             "role": "user",
8             "content": [
9                 {
10                     "type": "input_text",
11                     "text": "Tell me a three sentence bedtime story about a unicorn."
12                 }
13             ]
14         }
15     ]
16 }
```

```
14      }
15    ],
16    "first_id": "msg_abc123",
17    "last_id": "msg_abc123",
18    "has_more": false
19 }
```

Get input token counts

POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to

input_items for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with `conversation`.

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

- [Text inputs and outputs](#)
- [Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the [tools](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the [tool_choice](#) parameter.

› Show possible types

truncation string Optional

The truncation strategy to use for the model response. - [auto](#) : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping

items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
1 {
2   object: "response.input_tokens"
3   input_tokens: 123
4 }
```



Example request

python ⚙️



```
1 from openai import OpenAI
2
3 client = OpenAI()
4
5 response = client.responses.input_tokens.count(
6     model="gpt-5",
7     input="Tell me a joke."
8 )
9 print(response.input_tokens)
```

Response



```
1 {
2   "object": "response.input_tokens",
3   "input_tokens": 11
4 }
```

The response object

background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

› Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

› Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

› Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

- The length and order of items in the `output` array is dependent on the model's response.
- Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#). Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the

value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

- [Text inputs and outputs](#)
- [Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

- **Built-in tools:** Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).
- **MCP Tools:** Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).
- **Function calls (custom tools):** Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

- `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.
- `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object



```
1  {
2    "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",
3    "object": "response",
4    "created_at": 1741476777,
5    "status": "completed",
```

```
6 "error": null,
7 "incomplete_details": null,
8 "instructions": null,
9 "max_output_tokens": null,
10 "model": "gpt-4o-2024-08-06",
11 "output": [
12   {
13     "type": "message",
14     "id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
15     "status": "completed",
16     "role": "assistant",
17     "content": [
18       {
19         "type": "output_text",
20         "text": "The image depicts a scenic landscape with a wooden boardwalk or path leading through a lush green forest. The path is surrounded by tall trees with dense foliage. In the background, there are rolling hills and a clear blue sky with a few wispy clouds. The overall atmosphere is peaceful and natural, suggesting a remote outdoor setting like a national park or a well-preserved natural area. The lighting suggests it might be late afternoon or early evening, with soft sunlight filtering through the leaves. The boardwalk appears to be made of weathered wood planks, and the surrounding vegetation includes various shrubs, ferns, and wildflowers. The scene is captured from a slightly elevated angle, providing a comprehensive view of the landscape ahead. The colors are vibrant, with rich greens in the foliage and blues in the sky, creating a visually appealing composition. There are no people or animals visible in the image, emphasizing the tranquility and natural beauty of the environment. The path seems to lead towards a clearing or a specific destination, inviting the viewer to imagine where it might lead. The overall impression is one of a carefully composed photograph of a natural wonder, possibly taken during a leisurely walk or a nature-themed trip. The image is sharp and clear, allowing for detailed observation of the textures and details of the landscape elements. The composition is balanced, with the path leading the eye towards the center of the frame. The colors are well-saturated, making the scene look more vivid and lifelike. The lighting is natural, highlighting the textures of the wood and the leaves. The overall effect is one of a high-quality, professional photograph that captures the essence of a beautiful natural landscape.", "annotations": []
21     }
22   }
23 ]
24 },
25 ],
26 "parallel_tool_calls": true,
27 "previous_response_id": null,
28 "reasoning": {
29   "effort": null,
30   "summary": null
31 },
32 "store": true,
33 "temperature": 1,
34 "text": {
```

```
35     "format": {
36         "type": "text"
37     }
38 },
39 "tool_choice": "auto",
40 "tools": [],
41 "top_p": 1,
42 "truncation": "disabled",
43 "usage": {
44     "input_tokens": 328,
45     "input_tokens_details": {
46         "cached_tokens": 0
47     },
48     "output_tokens": 52,
49     "output_tokens_details": {
50         "reasoning_tokens": 0
51     },
52     "total_tokens": 380
53 },
54 "user": null,
55 "metadata": {}
56 }
```

The input item list

A list of Response items.

data array

A list of items used to generate this response.

› Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

OBJECT The input item list



```
1  {
2      "object": "list",
```

```
3 "data": [
4   {
5     "id": "msg_abc123",
6     "type": "message",
7     "role": "user",
8     "content": [
9       {
10        "type": "input_text",
11        "text": "Tell me a three sentence bedtime story about a unicorn."
12      }
13    ]
14  }
15 ],
16 "first_id": "msg_abc123",
17 "last_id": "msg_abc123",
18 "has_more": false
19 }
```

The compacted response object

created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

› Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

OBJECT The compacted response object



```
1  {
2    "id": "resp_001",
3    "object": "response.compaction",
4    "output": [
5      {
6        "type": "message",
7        "role": "user",
8        "content": [
```

```
9         {
10            "type": "input_text",
11            "text": "Summarize our launch checklist from last week."
12        }
13    ]
14 },
15 {
16   "type": "message",
17   "role": "user",
18   "content": [
19     {
20       "type": "input_text",
21       "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."
22     }
23   ]
24 },
25 {
26   "type": "compaction",
27   "id": "cmp_001",
28   "encrypted_content": "encrypted-summary"
29 }
30 ],
31 "created_at": 1731459200,
32 "usage": {
33   "input_tokens": 42897,
34   "output_tokens": 12000,
35   "total_tokens": 54912
36 }
37 }
```

PREVIOUS

< **Introduction**

NEXT

Conversations >

Responses



OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

[Extend the models with tools](#)

Create a model response



POST <https://api.openai.com/v1/responses>

Creates a model response. Provide text or image inputs to generate text or JSON outputs. Have the model call your own custom code or use built-in tools like web search or file search to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

[› Show possible types](#)

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

`web_search_call.action.sources` : Include the sources of the web search tool call.

`code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.

`computer_call_output.output.image_url` : Include image urls from the computer call output.

`file_search_call.results` : Include the search results of the file search tool call.

`message.input_image.image_url` : Include image urls from the input message.

`message.output_text.logprobs` : Include logprobs with assistant messages.

`reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

[Text inputs and outputs](#)

[Image inputs](#)

[File inputs](#)

[Conversation state](#)

[Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Text input](#)

[Image input](#)

[File input](#)

[Web search](#)

[File search](#)

[Streaming](#)

[Functions](#)

[Reasoning](#)

Example request

```
from openai import OpenAI

client = OpenAI()

tools = [
    {
        "type": "function",
        "name": "get_current_weather",
        "description": "Get the current weather in a given location",
        "parameters": {
            "type": "object",
            "properties": {
                "location": {
                    "type": "string",
                    "description": "The city and state, e.g. San Francisco, CA",
                },
                "unit": {"type": "string", "enum": ["celsius", "fahrenheit"]},
            },
            "required": ["location", "unit"],
        }
    }
]

response = client.responses.create(
    model="gpt-4.1",
    tools=tools,
    input="What is the weather like in Boston today?",
```

```
    tool_choice="auto"
)

print(response)
```

Response

```
{
  "id": "resp_67ca09c5efe0819096d0511c92b8c890096610f474011cc0",
  "object": "response",
  "created_at": 1741294021,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4.1-2025-04-14",
  "output": [
    {
      "type": "function_call",
      "id": "fc_67ca09c6bedc8190a7abfec07b1a1332096610f474011cc0",
      "call_id": "call_unLAR8MvFNptuiZK6K6HCy5k",
      "name": "get_current_weather",
      "arguments": "{\"location\": \"Boston, MA\", \"unit\": \"celsius\"}",
      "status": "completed"
    }
  ],
  "parallel_tool_calls": true,
  "previous_response_id": null,
```

```
"reasoning": {  
    "effort": null,  
    "summary": null  
},  
"store": true,  
"temperature": 1.0,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [  
    {  
        "type": "function",  
        "description": "Get the current weather in a given location",  
        "name": "get_current_weather",  
        "parameters": {  
            "type": "object",  
            "properties": {  
                "location": {  
                    "type": "string",  
                    "description": "The city and state, e.g. San Francisco, CA"  
                },  
                "unit": {  
                    "type": "string",  
                    "enum": [  
                        "celsius",  
                        "fahrenheit"  
                    ]  
                }  
            }  
        }  
    }  
]
```

```
        ],
      },
    },
    "required": [
      "location",
      "unit"
    ],
    "strict": true
  }
],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
  "input_tokens": 291,
  "output_tokens": 23,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 314
},
"user": null,
"metadata": {}
}
```

Get a model response



```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.retrieve("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
```

```
"incomplete_details": null,  
"instructions": null,  
"max_output_tokens": null,  
"model": "gpt-4o-2024-08-06",  
"output": [  
  {  
    "type": "message",  
    "id": "msg_67cb71b3c2b0819084d481baaaf148f206981a8637e6bc44",  
    "status": "completed",  
    "role": "assistant",  
    "content": [  
      {  
        "type": "output_text",  
        "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn break",  
        "annotations": []  
      }  
    ]  
  },  
  ],  
  "parallel_tool_calls": true,  
  "previous_response_id": null,  
  "reasoning": {  
    "effort": null,  
    "summary": null  
  },  
  "store": true,  
  "temperature": 1.0,  
  "text": {  
    "format": {
```

```
        "type": "text"
    },
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
    "input_tokens": 32,
    "input_tokens_details": {
        "cached_tokens": 0
    },
    "output_tokens": 18,
    "output_tokens_details": {
        "reasoning_tokens": 0
    },
    "total_tokens": 50
},
"user": null,
"metadata": {}
}
```

Delete a model response



```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.delete("resp_123")
print(response)
```

Response

```
{
  "id": "resp_6786a1bec27481909a17d673315b29f6",
```

```
"object": "response",
"deleted": true
}
```

Cancel a response



POST `https://api.openai.com/v1/responses/{response_id}/cancel`

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A [Response](#) object.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.cancel("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4o-2024-08-06",
  "output": [
    {
      "type": "message",
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",
      "status": "completed",
      "role": "assistant",
      "content": [
        {
          "type": "output_text",
```

```
"text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn breaks",
  "annotations": []
}
]
}
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
  "input_tokens": 32,
  "input_tokens_details": {
    "cached_tokens": 0
  }
},
"output_tokens": 18,
```

```
"output_tokens_details": {  
    "reasoning_tokens": 0  
,  
    "total_tokens": 50  
,  
    "user": null,  
    "metadata": {}  
}
```

Compact a response



POST <https://api.openai.com/v1/responses/compact>

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

```
from openai import OpenAI

client = OpenAI()

compacted_response = client.responses.compact(
    model="gpt-5.1-codex-max",
    input=[
        {
            "role": "user",
            "content": "Create a simple landing page for a dog petting cafe.",
        },
        # All items returned from previous requests are included here, like reasoning, message, function_call, etc.
        {
            "id": "msg_001",
            "type": "message",
            "status": "completed",
            "content": [
                {
                    "type": "output_text",
                    "annotations": [],
                    "logprobs": [],
                    "text": "Below is a single file, ready-to-use landing page for a dog petting café:...",
                },
            ],
            "role": "assistant",
        },
    ]
)
```

```
# Pass the compacted_response.output as input to the next request
print(compacted_response)
```

Response

```
{
  "id": "resp_001",
  "object": "response.compaction",
  "created_at": 1764967971,
  "output": [
    {
      "id": "msg_000",
      "type": "message",
      "status": "completed",
      "content": [
        {
          "type": "input_text",
          "text": "Create a simple landing page for a dog petting cafe."
        }
      ],
      "role": "user"
    },
    {
      "id": "cmp_001",
      "type": "compaction",
      "encrypted_content": "gAAAAABpM0Yj-...="
    }
  ],
}
```

```
"usage": {  
    "input_tokens": 139,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 438,  
    "output_tokens_details": {  
        "reasoning_tokens": 64  
    },  
    "total_tokens": 577  
}  
}
```

List input items



GET https://api.openai.com/v1/responses/{response_id}/input_items

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

`asc` : Return the input items in ascending order.

`desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.input_items.list("resp_123")
print(response.data)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "msg_abc123",
      "type": "message",
      "role": "user",
      "content": [
        {
          "type": "input_text",
          "text": "Tell me a three sentence bedtime story about a unicorn."
        }
      ]
    },
    "first_id": "msg_abc123",
    "last_id": "msg_abc123",
    "has_more": false
  }
}
```

Get input token counts



POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages

in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

reasoning object Optional**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

› Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
{  
  object: "response.input_tokens"  
  input_tokens: 123  
}
```

Example request

```
from openai import OpenAI  
  
client = OpenAI()  
  
response = client.responses.input_tokens.count(  
    model="gpt-5",  
    input="Tell me a joke."  
)  
print(response.input_tokens)
```

Response

```
{  
  "object": "response.input_tokens",  
  "input_tokens": 11  
}
```

The response object



background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

› Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

› Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

› Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the model guide to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

The length and order of items in the `output` array is dependent on the model's response.

Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more](#).

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

auto : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

disabled (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object

```
{  
  "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",  
  "object": "response",  
  "created_at": 1741476777,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {
```

```
"type": "message",
"id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading through a lush green forest. The path is made of weathered planks and curves along a small stream. In the background, there are tall evergreen trees and a bright sky with wispy clouds. The overall atmosphere is peaceful and natural.",

    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1,
```

```
"truncation": "disabled",
"usage": {
  "input_tokens": 328,
  "input_tokens_details": {
    "cached_tokens": 0
  },
  "output_tokens": 52,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 380
},
"user": null,
"metadata": {}
}
```

The input item list



A list of Response items.

data array

A list of items used to generate this response.

> Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be **list**.

OBJECT The input item list

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
],
"first_id": "msg_abc123",
"last_id": "msg_abc123",
"has_more": false
}
```

The compacted response object



created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

> Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

> Show properties

OBJECT The compacted response object

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "output": [  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Summarize our launch checklist from last week."  
        }  
      ]  
    },  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."  
        }  
      ]  
    }  
  ]  
}
```

```
        }
    ],
},
{
  "type": "compaction",
  "id": "cmp_001",
  "encrypted_content": "encrypted-summary"
}
],
"created_at": 1731459200,
"usage": {
  "input_tokens": 42897,
  "output_tokens": 12000,
  "total_tokens": 54912
}
}
```

< PREVIOUS
Introduction

NEXT

Responses



OpenAI's most advanced interface for generating model responses. Supports text and image inputs, and text outputs. Create stateful interactions with the model, using the output of previous responses as input. Extend the model's capabilities with built-in tools for file search, web search, computer use, and more. Allow the model access to external systems and data using function calling.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

[Extend the models with tools](#)

Create a model response



POST <https://api.openai.com/v1/responses>

Creates a model response. Provide text or image inputs to generate text or JSON outputs. Have the model call your own custom code or use built-in tools like web search or file search to use your own data as input for the model's response.

Request body

background boolean Optional Defaults to false

Whether to run the model response in the background. [Learn more](#).

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request. Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

`web_search_call.action.sources` : Include the sources of the web search tool call.

`code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.

`computer_call_output.output.image_url` : Include image urls from the computer call output.

`file_search_call.results` : Include the search results of the file search tool call.

`message.input_image.image_url` : Include image urls from the input message.

`message.output_text.logprobs` : Include logprobs with assistant messages.

`reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

input string or array Optional

Text, image, or file inputs to the model, used to generate a response.

Learn more:

[Text inputs and outputs](#)

[Image inputs](#)

[File inputs](#)

[Conversation state](#)

[Function calling](#)

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

max_output_tokens integer Optional

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and [reasoning tokens](#).

max_tool_calls integer Optional

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional Defaults to true

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more.](#)

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning object Optional

gpt-5 and o-series models only

Configuration options for [reasoning models](#).

› Show properties

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

store boolean Optional Defaults to true

Whether to store the generated model response for later retrieval via API.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

stream_options object Optional Defaults to null

Options for streaming responses. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

Structured Outputs

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string Optional Defaults to disabled

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

Returns

Returns a [Response](#) object.

[Text input](#)

[Image input](#)

[File input](#)

[Web search](#)

[File search](#)

[Streaming](#)

[Functions](#)

Reasoning

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.create(
    model="o3-mini",
    input="How much wood would a woodchuck chuck?",
    reasoning={
        "effort": "high"
    }
)

print(response)
```

Response

```
{
  "id": "resp_67ccd7eca01881908ff0b5146584e408072912b2993db808",
  "object": "response",
  "created_at": 1741477868,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "o1-2024-12-17",
  "output": [
    {
```

```
"type": "message",
"id": "msg_67ccd7f7b5848190a6f3e95d809f6b44072912b2993db808",
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "The classic tongue twister...",
    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": "high",
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
```

```
"truncation": "disabled",
"usage": {
  "input_tokens": 81,
  "input_tokens_details": {
    "cached_tokens": 0
  },
  "output_tokens": 1035,
  "output_tokens_details": {
    "reasoning_tokens": 832
  },
  "total_tokens": 1116
},
"user": null,
"metadata": {}
}
```

Get a model response



```
GET https://api.openai.com/v1/responses/{response_id}
```

Retrieves a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

include_obfuscation boolean Optional

When true, stream obfuscation will be enabled. Stream obfuscation adds random characters to an `obfuscation` field on streaming delta events to normalize payload sizes as a mitigation to certain side-channel attacks. These obfuscation fields are included by default, but add a small amount of overhead to the data stream. You can set `include_obfuscation` to false to optimize for bandwidth if you trust the network links between your application and the OpenAI API.

starting_after integer Optional

The sequence number of the event after which to start streaming.

stream boolean Optional

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information.

Returns

The [Response](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.retrieve("resp_123")
print(response)
```

Response

```
{
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",
  "object": "response",
  "created_at": 1741386163,
  "status": "completed",
  "error": null,
  "incomplete_details": null,
  "instructions": null,
  "max_output_tokens": null,
  "model": "gpt-4o-2024-08-06",
  "output": [
    {
      "type": "message",
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",
```

```
"status": "completed",
"role": "assistant",
"content": [
  {
    "type": "output_text",
    "text": "Silent circuits hum, \nThoughts emerge in data streams— \nDigital dawn breaks",
    "annotations": []
  }
],
"parallel_tool_calls": true,
"previous_response_id": null,
"reasoning": {
  "effort": null,
  "summary": null
},
"store": true,
"temperature": 1.0,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1.0,
"truncation": "disabled",
"usage": {
```

```
"input_tokens": 32,  
"input_tokens_details": {  
    "cached_tokens": 0  
},  
"output_tokens": 18,  
"output_tokens_details": {  
    "reasoning_tokens": 0  
},  
"total_tokens": 50  
},  
"user": null,  
"metadata": {}  
}
```

Delete a model response



```
DELETE https://api.openai.com/v1/responses/{response_id}
```

Deletes a model response with the given ID.

Path parameters

response_id string Required

The ID of the response to delete.

Returns

A success message.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.delete("resp_123")
print(response)
```

Response

```
{
  "id": "resp_6786a1bec27481909a17d673315b29f6",
  "object": "response",
  "deleted": true
}
```

Cancel a response



```
POST https://api.openai.com/v1/responses/{response_id}/cancel
```

Cancels a model response with the given ID. Only responses created with the `background` parameter set to `true` can be cancelled. [Learn more.](#)

Path parameters

response_id string Required

The ID of the response to cancel.

Returns

A [Response](#) object.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.cancel("resp_123")
print(response)
```

Response

```
{  
  "id": "resp_67cb71b351908190a308f3859487620d06981a8637e6bc44",  
  "object": "response",  
  "created_at": 1741386163,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {  
      "type": "message",  
      "id": "msg_67cb71b3c2b0819084d481baaf148f206981a8637e6bc44",  
      "status": "completed",  
      "role": "assistant",  
      "content": [  
        {  
          "type": "output_text",  
          "text": "Silent circuits hum,  \\nThoughts emerge in data streams–  \\nDigital dawn breaks",  
          "annotations": []  
        }  
      ]  
    },  
    {"parallel_tool_calls": true,  
     "previous_response_id": null},
```

```
"reasoning": {  
    "effort": null,  
    "summary": null  
},  
"store": true,  
"temperature": 1.0,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [],  
"top_p": 1.0,  
"truncation": "disabled",  
"usage": {  
    "input_tokens": 32,  
    "input_tokens_details": {  
        "cached_tokens": 0  
    },  
    "output_tokens": 18,  
    "output_tokens_details": {  
        "reasoning_tokens": 0  
    },  
    "total_tokens": 50  
},  
"user": null,  
"metadata": {}  
}
```

Compact a response



```
POST https://api.openai.com/v1/responses/compact
```

Runs a compaction pass over a conversation. Compaction returns encrypted, opaque items and the underlying logic may evolve over time.

Request body

model string Required

Model ID used to generate the response, like `gpt-5` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

> Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

Returns

A [compacted response object](#).

Learn when and how to compact long-running conversations in the [conversation state guide](#).

Example request

```
from openai import OpenAI

client = OpenAI()

compacted_response = client.responses.compact(
    model="gpt-5.1-codex-max",
    input=[
        {
            "role": "user",
            "content": "Create a simple landing page for a dog petting cafe.",
        }
    ]
)
```

```
},
# All items returned from previous requests are included here, like reasoning, message, function
{
  "id": "msg_001",
  "type": "message",
  "status": "completed",
  "content": [
    {
      "type": "output_text",
      "annotations": [],
      "logprobs": [],
      "text": "Below is a single file, ready-to-use landing page for a dog petting café:..."
    },
  ],
  "role": "assistant",
},
]
)
# Pass the compacted_response.output as input to the next request
print(compacted_response)
```

Response

```
{
  "id": "resp_001",
  "object": "response.compaction",
  "created_at": 1764967971,
  "output": [
```

```
{  
  "id": "msg_000",  
  "type": "message",  
  "status": "completed",  
  "content": [  
    {  
      "type": "input_text",  
      "text": "Create a simple landing page for a dog petting cafe."  
    }  
  ],  
  "role": "user"  
},  
{  
  "id": "cmp_001",  
  "type": "compaction",  
  "encrypted_content": "gAAAAABpM0Yj-...="  

```

List input items



```
GET https://api.openai.com/v1/responses/{response_id}/input_items
```

Returns a list of input items for a given response.

Path parameters

response_id string Required

The ID of the response to retrieve input items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Additional fields to include in the response. See the `include` parameter for Response creation above for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

`asc` : Return the input items in ascending order.

`desc` : Return the input items in descending order.

Returns

A list of input item objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.responses.input_items.list("resp_123")
print(response.data)
```

Response

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    }  
  ],  
  "first_id": "msg_abc123",  
  "last_id": "msg_abc123",  
  "has_more": false  
}
```

Get input token counts



POST https://api.openai.com/v1/responses/input_tokens

Returns input token counts of the request.

Request body

conversation string or object Optional Defaults to null

The conversation that this response belongs to. Items from this conversation are prepended to `input_items` for this response request.

Input items and output items from this response are automatically added to this conversation after this response completes.

› Show possible types

input string or array Optional

Text, image, or file inputs to the model, used to generate a response

› Show possible types

instructions string Optional

A system (or developer) message inserted into the model's context. When used along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

model string Optional

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

parallel_tool_calls boolean Optional

Whether to allow the model to run tool calls in parallel.

previous_response_id string Optional

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with [conversation](#).

reasoning object Optional**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

› Show properties

text object Optional

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object Optional

How the model should select which tool (or tools) to use when generating a response. See the [tools](#) parameter to see how to specify which tools the model can call.

› Show possible types

tools array Optional

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

➤ Show possible types

truncation string Optional

The truncation strategy to use for the model response. - `auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation. - `disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

Returns

The input token counts.

```
{  
  object: "response.input_tokens"  
  input_tokens: 123  
}
```

Example request

```
from openai import OpenAI  
  
client = OpenAI()
```

```
response = client.responses.input_tokens.count(  
    model="gpt-5",  
    input="Tell me a joke."  
)  
print(response.input_tokens)
```

Response

```
{  
    "object": "response.input_tokens",  
    "input_tokens": 11  
}
```

The response object



background boolean

Whether to run the model response in the background. [Learn more.](#)

conversation object

The conversation that this response belongs to. Input items and output items from this response are automatically added to this conversation.

› Show properties

created_at number

Unix timestamp (in seconds) of when this Response was created.

error object

An error object returned when the model fails to generate a Response.

> Show properties

id string

Unique identifier for this Response.

incomplete_details object

Details about why the response is incomplete.

> Show properties

instructions string or array

A system (or developer) message inserted into the model's context.

When using along with `previous_response_id`, the instructions from a previous response will not be carried over to the next response. This makes it simple to swap out system (or developer) messages in new responses.

> Show possible types

max_output_tokens integer

An upper bound for the number of tokens that can be generated for a response, including visible output tokens and reasoning tokens.

max_tool_calls integer

The maximum number of total calls to built-in tools that can be processed in a response. This maximum number applies across all built-in tool calls, not per individual tool. Any further attempts to call a tool by the model will be ignored.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

object string

The object type of this resource - always set to `response`.

output array

An array of content items generated by the model.

The length and order of items in the `output` array is dependent on the model's response.

Rather than accessing the first item in the `output` array and assuming it's an `assistant` message with the content generated by the model, you might consider using the `output_text` property where supported in SDKs.

› Show possible types

output_text string SDK Only

SDK-only convenience property that contains the aggregated text output from all `output_text` items in the `output` array, if any are present. Supported in the Python and JavaScript SDKs.

parallel_tool_calls boolean

Whether to allow the model to run tool calls in parallel.

previous_response_id string

The unique ID of the previous response to the model. Use this to create multi-turn conversations. Learn more about [conversation state](#).

Cannot be used in conjunction with `conversation`.

prompt object

Reference to a prompt template and its variables. [Learn more](#).

> Show properties

prompt_cache_key string

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning object**gpt-5 and o-series models only**

Configuration options for [reasoning models](#).

> Show properties

safety_identifier string

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more](#).

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

status string

The status of the response generation. One of `completed`, `failed`, `in_progress`, `cancelled`, `queued`, or `incomplete`.

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

text object

Configuration options for a text response from the model. Can be plain text or structured JSON data. Learn more:

[Text inputs and outputs](#)

[Structured Outputs](#)

› Show properties

tool_choice string or object

How the model should select which tool (or tools) to use when generating a response. See the `tools` parameter to see how to specify which tools the model can call.

› Show possible types

tools array

An array of tools the model may call while generating a response. You can specify which tool to use by setting the `tool_choice` parameter.

We support the following categories of tools:

Built-in tools: Tools that are provided by OpenAI that extend the model's capabilities, like [web search](#) or [file search](#). Learn more about [built-in tools](#).

MCP Tools: Integrations with third-party systems via custom MCP servers or predefined connectors such as Google Drive and SharePoint. Learn more about [MCP Tools](#).

Function calls (custom tools): Functions that are defined by you, enabling the model to call your own code with strongly typed arguments and outputs. Learn more about [function calling](#). You can also use custom tools to call your own code.

› Show possible types

top_logprobs integer

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

truncation string

The truncation strategy to use for the model response.

`auto` : If the input to this Response exceeds the model's context window size, the model will truncate the response to fit the context window by dropping items from the beginning of the conversation.

`disabled` (default): If the input size will exceed the context window size for a model, the request will fail with a 400 error.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

› Show properties

user Deprecated string

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

OBJECT The response object

```
{  
  "id": "resp_67ccd3a9da748190baa7f1570fe91ac604becb25c45c1d41",  
  "object": "response",  
  "created_at": 1741476777,  
  "status": "completed",  
  "error": null,  
  "incomplete_details": null,  
  "instructions": null,  
  "max_output_tokens": null,  
  "model": "gpt-4o-2024-08-06",  
  "output": [  
    {  
      "type": "message",  
      "id": "msg_67ccd3acc8d48190a77525dc6de64b4104becb25c45c1d41",  
      "status": "completed",  
      "role": "assistant",  
      "content": [  
        {  
          "type": "output_text",  
          "text": "The image depicts a scenic landscape with a wooden boardwalk or pathway leading",  
          "annotations": []  
        }  
      ]  
    }  
  ]
```

```
        },
      ],
      "parallel_tool_calls": true,
      "previous_response_id": null,
      "reasoning": {
        "effort": null,
        "summary": null
      },
      "store": true,
      "temperature": 1,
      "text": {
        "format": {
          "type": "text"
        }
      },
      "tool_choice": "auto",
      "tools": [],
      "top_p": 1,
      "truncation": "disabled",
      "usage": {
        "input_tokens": 328,
        "input_tokens_details": {
          "cached_tokens": 0
        },
        "output_tokens": 52,
        "output_tokens_details": {
          "reasoning_tokens": 0
        },
        "total_tokens": 380
      }
    }
  }
}
```

```
},  
  "user": null,  
  "metadata": {}  
}
```

The input item list



A list of Response items.

data array

A list of items used to generate this response.

> Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

OBJECT The input item list

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "msg_abc123",  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Tell me a three sentence bedtime story about a unicorn."  
        }  
      ]  
    },  
    {"first_id": "msg_abc123",  
     "last_id": "msg_abc123",  
     "has_more": false  
  }  
}
```

The compacted response object



created_at integer

Unix timestamp (in seconds) when the compacted conversation was created.

id string

The unique identifier for the compacted response.

object string

The object type. Always `response.compaction`.

output array

The compacted list of output items.

> Show possible types

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used.

> Show properties

OBJECT The compacted response object

```
{  
  "id": "resp_001",  
  "object": "response.compaction",  
  "output": [  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "Summarize our launch checklist from last week."  
        }  
      ]  
    },  
    {  
      "type": "message",  
      "role": "user",  
      "content": [  
        {  
          "type": "input_text",  
          "text": "You are performing a CONTEXT CHECKPOINT COMPACTION..."  
        }  
      ]  
    },  
    {  
      "type": "compaction",  
      "id": "cmp_001",  
      "encrypted_content": "encrypted-summary"  
    }  
  ]  
}
```

```
],  
  "created_at": 1731459200,  
  "usage": {  
    "input_tokens": 42897,  
    "output_tokens": 12000,  
    "total_tokens": 54912  
  }  
}
```

< PREVIOUS
Introduction

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Conversations

Create and manage conversations to store and retrieve conversation state across Response API calls.

Create a conversation

```
POST https://api.openai.com/v1/conversations
```

Create a conversation.

Request body

items array Optional

Initial items to include in the conversation context. You may add up to 20 items at a time.

› Show possible types

metadata object or null Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard. Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

Returns a [Conversation](#) object.

Example request

python ⚖️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 conversation = client.conversations.create(
5     metadata={"topic": "demo"},
6     items=[
7         {"type": "message", "role": "user", "content": "Hello!"}
8     ]
```

```
9  )
10 print(conversation)
```

Response



```
1 {
2   "id": "conv_123",
3   "object": "conversation",
4   "created_at": 1741900000,
5   "metadata": {"topic": "demo"}
6 }
```

Retrieve a conversation

```
GET https://api.openai.com/v1/conversations/{conversation_id}
```

Get a conversation

Path parameters

conversation_id string Required

The ID of the conversation to retrieve.

Returns

Returns a [Conversation](#) object.

Example request

python ▼ 

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 conversation = client.conversations.retrieve("conv_123")
5 print(conversation)
```

Response



```
1 {
2   "id": "conv_123",
3   "object": "conversation",
4   "created_at": 1741900000,
5   "metadata": {"topic": "demo"}
6 }
```

Update a conversation

```
POST https://api.openai.com/v1/conversations/{conversation_id}
```

Update a conversation

Path parameters

conversation_id string Required

The ID of the conversation to update.

Request body

metadata map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

Returns the updated [Conversation](#) object.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 updated = client.conversations.update(
5     "conv_123",
6     metadata={"topic": "project-x"}
7 )
8 print(updated)
```

Response

🔗

```
1 {
2     "id": "conv_123",
3     "object": "conversation",
4     "created_at": 1741900000,
5     "metadata": {"topic": "project-x"}
6 }
```

Delete a conversation

```
DELETE https://api.openai.com/v1/conversations/{conversation_id}
```

Delete a conversation. Items in the conversation will not be deleted.

Path parameters

conversation_id string Required

The ID of the conversation to delete.

Returns

A success message.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
```

```
4 deleted = client.conversations.delete("conv_123")
5 print(deleted)
```

Response



```
1 {
2   "id": "conv_123",
3   "object": "conversation.deleted",
4   "deleted": true
5 }
```

List items

```
GET https://api.openai.com/v1/conversations/{conversation_id}/items
```

List all items for a conversation with the given ID.

Path parameters

conversation_id string Required

The ID of the conversation to list items for.

Query parameters

after string Optional

An item ID to list items after, used in pagination.

include array Optional

Specify additional output data to include in the model response. Currently supported values are:

- `web_search_call.action.sources` : Include the sources of the web search tool call.
- `code_interpreter_call.outputs` : Includes the outputs of python code execution in code interpreter tool call items.
- `computer_call_output.output.image_url` : Include image urls from the computer call output.
- `file_search_call.results` : Include the search results of the file search tool call.
- `message.input_image.image_url` : Include image urls from the input message.
- `message.output_text.logprobs` : Include logprobs with assistant messages.
- `reasoning.encrypted_content` : Includes an encrypted version of reasoning tokens in reasoning item outputs. This enables reasoning items to be used in multi-turn conversations when using the Responses API statelessly (like when the `store` parameter is set to `false`, or when an organization is enrolled in the zero data retention program).

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional

The order to return the input items in. Default is `desc`.

- `asc` : Return the input items in ascending order.
- `desc` : Return the input items in descending order.

Returns

Returns a [list object](#) containing Conversation items.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 items = client.conversations.items.list("conv_123", limit=10)
5 print(items.data)
```

Response

⌚

```
1  {
2      "object": "list",
3      "data": [
4          {
5              "type": "message",
6              "id": "msg_abc",
7              "status": "completed",
8              "role": "user",
9              "content": [
10                  {"type": "input_text", "text": "Hello!"}
11              ]
12          }
13      ],
14      "first_id": "msg_abc",
15      "last_id": "msg_abc",
16      "has_more": false
17 }
```

Create items

POST https://api.openai.com/v1/conversations/{conversation_id}/items

Create items in a conversation with the given ID.

Path parameters

conversation_id string Required

The ID of the conversation to add the item to.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for [listing Conversation items above](#) for more information.

Request body

items array Required

The items to add to the conversation. You may add up to 20 items at a time.

› Show possible types

Returns

Returns the list of added items.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 items = client.conversations.items.create(
5     "conv_123",
6     items=[
7         {
8             "type": "message",
9             "role": "user",
10            "content": [{"type": "input_text", "text": "Hello!"}],
11        },
12        {
13            "type": "message",
14            "role": "user",
15            "content": [{"type": "input_text", "text": "How are you?"}],
16        }
17    ],
18 )
19 print(items.data)
```

Response

⌚

```
1  {
2      "object": "list",
3      "data": [
4          {
5              "type": "message",
6              "id": "msg_abc",
7              "status": "completed",
8              "role": "user",
9              "content": [
10                  {"type": "input_text", "text": "Hello!"}
11              ]
12          },
13          {
14              "type": "message",
15              "id": "msg_def",
16              "status": "completed",
17              "role": "user",
18              "content": [
19                  {"type": "input_text", "text": "How are you?"}
20              ]
21          }
22      ],
23      "first_id": "msg_abc",
24      "last_id": "msg_def",
25      "has_more": false
26  }
```

Retrieve an item

```
GET https://api.openai.com/v1/conversations/{conversation_id}/items/{item_id}
```

Get a single item from a conversation with the given IDs.

Path parameters

conversation_id string Required

The ID of the conversation that contains the item.

item_id string Required

The ID of the item to retrieve.

Query parameters

include array Optional

Additional fields to include in the response. See the `include` parameter for [listing Conversation items above](#) for more information.

Returns

Returns a [Conversation Item](#).

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 item = client.conversations.items.retrieve("conv_123", "msg_abc")
5 print(item)
```

Response

🔗

```
1 {
2   "type": "message",
3   "id": "msg_abc",
4   "status": "completed",
5   "role": "user",
6   "content": [
7     {"type": "input_text", "text": "Hello!"}
8   ]
9 }
```

Delete an item

```
DELETE https://api.openai.com/v1/conversations/{conversation_id}/items/{item_id}
```

Delete an item from a conversation with the given IDs.

Path parameters

conversation_id string Required

The ID of the conversation that contains the item.

item_id string Required

The ID of the item to delete.

Returns

Returns the updated [Conversation](#) object.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 conversation = client.conversations.items.delete("conv_123", "msg_abc")
5 print(conversation)
```

Response



```
1 {
2   "id": "conv_123",
3   "object": "conversation",
4   "created_at": 1741900000,
5   "metadata": {"topic": "demo"}
6 }
```

The conversation object

created_at integer

The time at which the conversation was created, measured in seconds since the Unix epoch.

id string

The unique ID of the conversation.

metadata

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard. Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

object string

The object type, which is always `conversation`.

The item list

A list of Conversation items.

data array

A list of conversation items.

› Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

< PREVIOUS
Responses

NEXT >
Streaming events

Streaming events



When you [create a Response](#) with `stream` set to `true`, the server will emit server-sent events to the client as the Response is generated. This section contains the events that are emitted by the server.

[Learn more about streaming responses.](#)



response.created



An event that is emitted when a response is created.

response object

The response that was created.

> Show properties

sequence_number integer

The sequence number for this event.

type string

The type of the event. Always `response.created`.

OBJECT `response.created`

```
{  
  "type": "response.created",  
  "response": {  
    "id": "resp_67ccfcdd16748190a91872c75d38539e09e4d4aac714747c",  
    "object": "response",  
    "created_at": 1741487325,  
    "status": "in_progress",  
    "error": null,  
    "incomplete_details": null,  
    "instructions": null,  
    "max_output_tokens": null,  
    "model": "gpt-4o-2024-08-06",  
    "output": [],  
    "parallel_tool_calls": true,  
    "previous_response_id": null,  
    "reasoning": {  
      "effort": null,  
      "summary": null  
    },  
    "store": true,  
    "temperature": 1,  
    "text": {  
      "content": null,  
      "t朝向": null  
    }  
  }  
}
```

```
"format": {  
    "type": "text"  
}  
,  
"tool_choice": "auto",  
"tools": [],  
"top_p": 1,  
"truncation": "disabled",  
"usage": null,  
"user": null,  
"metadata": {}  
},  
"sequence_number": 1  
}
```

response.in_progress

Emitted when the response is in progress.

response object

The response that is in progress.

> Show properties

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.in_progress`.

OBJECT `response.in_progress`

```
{  
  "type": "response.in_progress",  
  "response": {  
    "id": "resp_67ccfcdd16748190a91872c75d38539e09e4d4aac714747c",  
    "object": "response",  
    "created_at": 1741487325,  
    "status": "in_progress",  
    "error": null,  
    "incomplete_details": null,  
    "instructions": null,  
    "max_output_tokens": null,  
    "model": "gpt-4o-2024-08-06",  
    "output": [],  
    "parallel_tool_calls": true,  
    "previous_response_id": null,  
    "reasoning": {  
      "effort": null,  
      "summary": null  
    },  
    "store": true,  
  }  
}
```

```
"temperature": 1,  
"text": {  
    "format": {  
        "type": "text"  
    }  
},  
"tool_choice": "auto",  
"tools": [],  
"top_p": 1,  
"truncation": "disabled",  
"usage": null,  
"user": null,  
"metadata": {}  
},  
"sequence_number": 1  
}
```

response.completed

Emitted when the model response is complete.

response object

Properties of the completed response.

> Show properties

sequence_number integer

The sequence number for this event.

type string

The type of the event. Always `response.completed`.

OBJECT `response.completed`

```
{  
  "type": "response.completed",  
  "response": {  
    "id": "resp_123",  
    "object": "response",  
    "created_at": 1740855869,  
    "status": "completed",  
    "error": null,  
    "incomplete_details": null,  
    "input": [],  
    "instructions": null,  
    "max_output_tokens": null,  
    "model": "gpt-4o-mini-2024-07-18",  
    "output": [  
      {  
        "id": "msg_123",  
        "type": "message",  
        "role": "assistant",  
        "content": "Hello!"  
      }  
    ]  
  }  
}
```

```
"content": [
  {
    "type": "output_text",
    "text": "In a shimmering forest under a sky full of stars, a lonely unicorn named Li:",
    "annotations": []
  }
],
"previous_response_id": null,
"reasoning_effort": null,
"store": false,
"temperature": 1,
"text": {
  "format": {
    "type": "text"
  }
},
"tool_choice": "auto",
"tools": [],
"top_p": 1,
"truncation": "disabled",
"usage": {
  "input_tokens": 0,
  "output_tokens": 0,
  "output_tokens_details": {
    "reasoning_tokens": 0
  },
  "total_tokens": 0
}
```

```
  },
  "user": null,
  "metadata": {}
},
"sequence_number": 1
}
```

response.failed



An event that is emitted when a response fails.

response object

The response that failed.

> Show properties

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.failed`.

OBJECT response.failed

```
{  
  "type": "response.failed",  
  "response": {  
    "id": "resp_123",  
    "object": "response",  
    "created_at": 1740855869,  
    "status": "failed",  
    "error": {  
      "code": "server_error",  
      "message": "The model failed to generate a response."  
    },  
    "incomplete_details": null,  
    "instructions": null,  
    "max_output_tokens": null,  
    "model": "gpt-4o-mini-2024-07-18",  
    "output": [],  
    "previous_response_id": null,  
    "reasoning_effort": null,  
    "store": false,  
    "temperature": 1,  
    "text": {  
      "format": {  
        "type": "text"  
      }  
    },  
    "tool_choice": "auto",  
    "tools": []  
}
```

```
"top_p": 1,  
"truncation": "disabled",  
"usage": null,  
"user": null,  
"metadata": {}  
}  
}
```

response.incomplete



An event that is emitted when a response finishes as incomplete.

response object

The response that was incomplete.

› Show properties

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.incomplete`.

OBJECT response.incomplete

```
{  
  "type": "response.incomplete",  
  "response": {  
    "id": "resp_123",  
    "object": "response",  
    "created_at": 1740855869,  
    "status": "incomplete",  
    "error": null,  
    "incomplete_details": {  
      "reason": "max_tokens"  
    },  
    "instructions": null,  
    "max_output_tokens": null,  
    "model": "gpt-4o-mini-2024-07-18",  
    "output": [],  
    "previous_response_id": null,  
    "reasoning_effort": null,  
    "store": false,  
    "temperature": 1,  
    "text": {  
      "format": {  
        "type": "text"  
      }  
    },  
    "tool_choice": "auto",  
    "tools": [],  
    "top_p": 1,  
  }  
}
```

```
"truncation": "disabled",
"usage": null,
"user": null,
"metadata": {}
},
"sequence_number": 1
}
```



response.output_item.added



Emitted when a new output item is added.

item object

The output item that was added.

› Show possible types

output_index integer

The index of the output item that was added.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.output_item.added`.

OBJECT `response.output_item.added`

```
{  
  "type": "response.output_item.added",  
  "output_index": 0,  
  "item": {  
    "id": "msg_123",  
    "status": "in_progress",  
    "type": "message",  
    "role": "assistant",  
    "content": []  
  },  
  "sequence_number": 1  
}
```

response.output_item.done

Emitted when an output item is marked done.

item object

The output item that was marked done.

› Show possible types

output_index integer

The index of the output item that was marked done.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.output_item.done`.

OBJECT `response.output_item.done`

```
{  
  "type": "response.output_item.done",  
  "output_index": 0,  
  "item": {  
    "id": "msg_123",  
    "status": "completed",  
    "type": "message",  
    "role": "assistant",  
    "content": [  
      "text": "Hello, how can I assist you today?"]  
  }  
}
```

```
{  
    "type": "output_text",  
    "text": "In a shimmering forest under a sky full of stars, a lonely unicorn named Lila d:",  
    "annotations": []  
}  
]  
},  
"sequence_number": 1  
}
```



response.content_part.added



Emitted when a new content part is added.

content_index integer

The index of the content part that was added.

item_id string

The ID of the output item that the content part was added to.

output_index integer

The index of the output item that the content part was added to.

part object

The content part that was added.

› Show possible types

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.content_part.added`.

OBJECT `response.content_part.added`

```
{  
  "type": "response.content_part.added",  
  "item_id": "msg_123",  
  "output_index": 0,  
  "content_index": 0,  
  "part": {  
    "type": "output_text",  
    "text": "",  
    "annotations": []  
  },
```

response.content_part.done



Emitted when a content part is done.

content_index integer

The index of the content part that is done.

item_id string

The ID of the output item that the content part was added to.

output_index integer

The index of the output item that the content part was added to.

part object

The content part that is done.

› Show possible types

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.content_part.done`.

OBJECT `response.content_part.done`

```
{  
  "type": "response.content_part.done",  
  "item_id": "msg_123",  
  "output_index": 0,  
  "content_index": 0,  
  "sequence_number": 1,  
  "part": {  
    "type": "output_text",  
    "text": "In a shimmering forest under a sky full of stars, a lonely unicorn named Lila discove  
    "annotations": []  
  }  
}
```

response.output_text.delta

⌚

Emitted when there is an additional text delta.

content_index integer

The index of the content part that the text delta was added to.

delta string

The text delta that was added.

item_id string

The ID of the output item that the text delta was added to.

logprobs array

The log probabilities of the tokens in the delta.

› Show properties

output_index integer

The index of the output item that the text delta was added to.

sequence_number integer

The sequence number for this event.

type string

The type of the event. Always `response.output_text.delta`.

OBJECT `response.output_text.delta`

```
{  
  "type": "response.output_text.delta",  
  "item_id": "msg_123",  
  "output_index": 0,  
  "content_index": 0,  
  "delta": "In",  
  "sequence_number": 1  
}
```

response.output_text.done

Emitted when text content is finalized.

content_index integer

The index of the content part that the text content is finalized.

item_id string

The ID of the output item that the text content is finalized.

logprobs array

The log probabilities of the tokens in the delta.

> Show properties

output_index integer

The index of the output item that the text content is finalized.

sequence_number integer

The sequence number for this event.

text string

The text content that is finalized.

type string

The type of the event. Always `response.output_text.done`.

OBJECT `response.output_text.done`

```
{  
  "type": "response.output_text.done",  
  "item_id": "msg_123",  
  "output_index": 0,  
  "content_index": 0,  
  "text": "In a shimmering forest under a sky full of stars, a lonely unicorn named Lila discovered a hidden stream flowing into a crystal-clear pond.",  
  "sequence_number": 1  
}
```



response.refusal.delta



Emitted when there is a partial refusal text.

content_index integer

The index of the content part that the refusal text is added to.

delta string

The refusal text that is added.

item_id string

The ID of the output item that the refusal text is added to.

output_index integer

The index of the output item that the refusal text is added to.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.refusal.delta`.

OBJECT `response.refusal.delta`

```
{  
  "type": "response.refusal.delta",  
  "item_id": "msg_123",  
  "output_index": 0,  
  "content_index": 0,  
  "delta": "refusal text so far",  
  "sequence_number": 1  
}
```

response.refusal.done



Emitted when refusal text is finalized.

content_index integer

The index of the content part that the refusal text is finalized.

item_id string

The ID of the output item that the refusal text is finalized.

output_index integer

The index of the output item that the refusal text is finalized.

refusal string

The refusal text that is finalized.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.refusal.done`.

OBJECT `response.refusal.done`

```
{  
  "type": "response.refusal.done",  
  "item_id": "item-abc",  
  "output_index": 1,  
  "content_index": 2,  
  "refusal": "final refusal text",  
  "sequence_number": 1  
}
```



response.function_call_arguments.delta



Emitted when there is a partial function-call arguments delta.

delta string

The function-call arguments delta that is added.

item_id string

The ID of the output item that the function-call arguments delta is added to.

output_index integer

The index of the output item that the function-call arguments delta is added to.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.function_call_arguments.delta`.

OBJECT `response.function_call_arguments.delta`

```
{  
  "type": "response.function_call_arguments.delta",  
  "item_id": "item-abc",  
  "output_index": 0,  
  "delta": "{ \"arg\": "  
  "sequence_number": 1  
}
```

response.function_call_arguments.done



Emitted when function-call arguments are finalized.

arguments string

The function-call arguments.

item_id string

The ID of the item.

name string

The name of the function that was called.

output_index integer

The index of the output item.

sequence_number integer

The sequence number of this event.

type string

```
OBJECT response.function_call_arguments.done
```

```
{  
  "type": "response.function_call_arguments.done",  
  "item_id": "item-abc",  
  "name": "get_weather",  
  "output_index": 1,  
  "arguments": "{ \"arg\": 123 }",  
  "sequence_number": 1  
}
```



response.file_search_call.in_progress



Emitted when a file search call is initiated.

item_id string

The ID of the output item that the file search call is initiated.

output_index integer

The index of the output item that the file search call is initiated.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.file_search_call.in_progress`.

OBJECT `response.file_search_call.in_progress`

```
{  
  "type": "response.file_search_call.in_progress",  
  "output_index": 0,  
  "item_id": "fs_123",  
  "sequence_number": 1  
}
```

response.file_search_call.searching



Emitted when a file search is currently searching.

item_id string

The ID of the output item that the file search call is initiated.

output_index integer

The index of the output item that the file search call is searching.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.file_search_call.searching`.

OBJECT `response.file_search_call.searching`

```
{  
  "type": "response.file_search_call.searching",  
  "output_index": 0,  
  "item_id": "fs_123",  
  "sequence_number": 1  
}
```

response.file_search_call.completed



Emitted when a file search call is completed (results found).

item_id string

The ID of the output item that the file search call is initiated.

output_index integer

The index of the output item that the file search call is initiated.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.file_search_call.completed`.

OBJECT `response.file_search_call.completed`

```
{  
  "type": "response.file_search_call.completed",  
  "output_index": 0,  
  "item_id": "fs_123",  
  "sequence_number": 1  
}
```



response.web_search_call.in_progress



Emitted when a web search call is initiated.

item_id string

Unique ID for the output item associated with the web search call.

output_index integer

The index of the output item that the web search call is associated with.

sequence_number integer

The sequence number of the web search call being processed.

type string

The type of the event. Always `response.web_search_call.in_progress`.

OBJECT `response.web_search_call.in_progress`

```
{  
  "type": "response.web_search_call.in_progress",  
  "output_index": 0,  
  "item_id": "ws_123",  
  "sequence_number": 0  
}
```

response.web_search_call.searching



Emitted when a web search call is executing.

item_id string

Unique ID for the output item associated with the web search call.

output_index integer

The index of the output item that the web search call is associated with.

sequence_number integer

The sequence number of the web search call being processed.

type string

The type of the event. Always `response.web_search_call.searching`.

OBJECT `response.web_search_call.searching`

```
{  
  "type": "response.web_search_call.searching",  
  "output_index": 0,  
  "item_id": "ws_123",  
  "sequence_number": 0  
}
```

response.web_search_call.completed



Emitted when a web search call is completed.

item_id string

Unique ID for the output item associated with the web search call.

output_index integer

The index of the output item that the web search call is associated with.

sequence_number integer

The sequence number of the web search call being processed.

type string

The type of the event. Always `response.web_search_call.completed`.

OBJECT `response.web_search_call.completed`

```
{  
  "type": "response.web_search_call.completed",  
  "output_index": 0,  
  "item_id": "ws_123",  
  "sequence_number": 0  
}
```



response.reasoning_summary_part.added



Emitted when a new reasoning summary part is added.

item_id string

The ID of the item this summary part is associated with.

output_index integer

The index of the output item this summary part is associated with.

part object

The summary part that was added.

› Show properties

sequence_number integer

The sequence number of this event.

summary_index integer

The index of the summary part within the reasoning summary.

type string

The type of the event. Always `response.reasoning_summary_part.added`.

OBJECT `response.reasoning_summary_part.added`

```
{  
  "type": "response.reasoning_summary_part.added",  
  "item_id": "rs_6806bfca0b2481918a5748308061a2600d3ce51bdffd5476",  
  "output_index": 0,  
  "summary_index": 0,  
  "part": {  
    "type": "summary_text",  
    "text": ""  
  },
```

```
"sequence_number": 1
```

response.reasoning_summary_part.done



Emitted when a reasoning summary part is completed.

item_id string

The ID of the item this summary part is associated with.

output_index integer

The index of the output item this summary part is associated with.

part object

The completed summary part.

› Show properties

sequence_number integer

The sequence number of this event.

summary_index integer

The index of the summary part within the reasoning summary.

type string

The type of the event. Always `response.reasoning_summary_part.done`.

OBJECT `response.reasoning_summary_part.done`

```
{  
  "type": "response.reasoning_summary_part.done",  
  "item_id": "rs_6806bfca0b2481918a5748308061a2600d3ce51bdffd5476",  
  "output_index": 0,  
  "summary_index": 0,  
  "part": {  
    "type": "summary_text",  
    "text": "***Responding to a greeting**\n\nThe user just said, \"Hello!\" So, it seems I need to  
  },  
  "sequence_number": 1  
}
```



response.reasoning_summary_text.delta



Emitted when a delta is added to a reasoning summary text.

delta string

The text delta that was added to the summary.

item_id string

The ID of the item this summary text delta is associated with.

output_index integer

The index of the output item this summary text delta is associated with.

sequence_number integer

The sequence number of this event.

summary_index integer

The index of the summary part within the reasoning summary.

type string

The type of the event. Always `response.reasoning_summary_text.delta`.

```
OBJECT response.reasoning_summary_text.delta
{
  "type": "response.reasoning_summary_text.delta",
  "item_id": "rs_6806bfca0b2481918a5748308061a2600d3ce51bdffd5476",
```

```
"output_index": 0,  
"summary_index": 0,  
"delta": "**Responding to a greeting**\n\nThe user just said, \"Hello!\" So, it seems I need to  
"sequence_number": 1  
}
```

response.reasoning_summary_text.done

Emitted when a reasoning summary text is completed.

item_id string

The ID of the item this summary text is associated with.

output_index integer

The index of the output item this summary text is associated with.

sequence_number integer

The sequence number of this event.

summary_index integer

The index of the summary part within the reasoning summary.

text string

The full text of the completed reasoning summary.

type string

The type of the event. Always `response.reasoning_summary_text.done`.

OBJECT `response.reasoning_summary_text.done`

```
{  
  "type": "response.reasoning_summary_text.done",  
  "item_id": "rs_6806bfca0b2481918a5748308061a2600d3ce51bdffd5476",  
  "output_index": 0,  
  "summary_index": 0,  
  "text": "**Responding to a greeting**\n\nThe user just said, \"Hello!\" So, it seems I need to e  
  "sequence_number": 1  
}
```



response.reasoning_text.delta



Emitted when a delta is added to a reasoning text.

content_index integer

The index of the reasoning content part this delta is associated with.

delta string

The text delta that was added to the reasoning content.

item_id string

The ID of the item this reasoning text delta is associated with.

output_index integer

The index of the output item this reasoning text delta is associated with.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `response.reasoning_text.delta`.

```
OBJECT response.reasoning_text.delta
```

{

```
  "type": "response.reasoning_text.delta",
  "item_id": "rs_123",
```

```
"output_index": 0,  
"content_index": 0,  
"delta": "The",  
"sequence_number": 1  
}
```

response.reasoning_text.done



Emitted when a reasoning text is completed.

content_index integer

The index of the reasoning content part.

item_id string

The ID of the item this reasoning text is associated with.

output_index integer

The index of the output item this reasoning text is associated with.

sequence_number integer

The sequence number of this event.

text string

The full text of the completed reasoning content.

type string

The type of the event. Always `response.reasoning_text.done`.

OBJECT `response.reasoning_text.done`

```
{  
  "type": "response.reasoning_text.done",  
  "item_id": "rs_123",  
  "output_index": 0,  
  "content_index": 0,  
  "text": "The user is asking...",  
  "sequence_number": 4  
}
```



response.image_generation_call.completed



Emitted when an image generation tool call has completed and the final image is available.

item_id string

The unique identifier of the image generation item being processed.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.image_generation_call.completed'.

```
OBJECT response.image_generation_call.completed

{
  "type": "response.image_generation_call.completed",
  "output_index": 0,
  "item_id": "item-123",
  "sequence_number": 1
}
```

response.image_generation_call.generating

Emitted when an image generation tool call is actively generating an image (intermediate state).

item_id string

The unique identifier of the image generation item being processed.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of the image generation item being processed.

type string

The type of the event. Always 'response.image_generation_call.generating'.

```
OBJECT response.image_generation_call.generating
```

```
{  
  "type": "response.image_generation_call.generating",  
  "output_index": 0,  
  "item_id": "item-123",  
  "sequence_number": 0  
}
```

response.image_generation_call.in_progress



Emitted when an image generation tool call is in progress.

item_id string

The unique identifier of the image generation item being processed.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of the image generation item being processed.

type string

The type of the event. Always 'response.image_generation_call.in_progress'.

```
OBJECT response.image_generation_call.in_progress

{
  "type": "response.image_generation_call.in_progress",
  "output_index": 0,
  "item_id": "item-123",
  "sequence_number": 0
}
```

response.image_generation_call.partial_image



Emitted when a partial image is available during image generation streaming.

item_id string

The unique identifier of the image generation item being processed.

output_index integer

The index of the output item in the response's output array.

partial_image_b64 string

Base64-encoded partial image data, suitable for rendering as an image.

partial_image_index integer

0-based index for the partial image (backend is 1-based, but this is 0-based for the user).

sequence_number integer

The sequence number of the image generation item being processed.

type string

The type of the event. Always 'response.image_generation_call.partial_image'.

```
OBJECT response.image_generation_call.partial_image
```

```
{  
  "type": "response.image_generation_call.partial_image",  
  "output_index": 0,  
  "item_id": "item-123",  
  "sequence_number": 0,  
  "partial_image_index": 0,  
  "partial_image_b64": "..."  
}
```



response.mcp_call_arguments.delta



Emitted when there is a delta (partial update) to the arguments of an MCP tool call.

delta string

A JSON string containing the partial update to the arguments for the MCP tool call.

item_id string

The unique identifier of the MCP tool call item being processed.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_call_arguments.delta'.

```
OBJECT response.mcp_call_arguments.delta
```

{

```
  "type": "response.mcp_call_arguments.delta",
  "output_index": 0,
```

```
"item_id": "item-abc",
"delta": "{}",
"sequence_number": 1
}
```

response.mcp_call_arguments.done



Emitted when the arguments for an MCP tool call are finalized.

arguments string

A JSON string containing the finalized arguments for the MCP tool call.

item_id string

The unique identifier of the MCP tool call item being processed.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_call_arguments.done'.

```
OBJECT response.mcp_call_arguments.done

{
  "type": "response.mcp_call_arguments.done",
  "output_index": 0,
  "item_id": "item-abc",
  "arguments": "{\"arg1\": \"value1\", \"arg2\": \"value2\"}",
  "sequence_number": 1
}
```



response.mcp_call.completed



Emitted when an MCP tool call has completed successfully.

item_id string

The ID of the MCP tool call item that completed.

output_index integer

The index of the output item that completed.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_call.completed'.

```
OBJECT response.mcp_call.completed
{
  "type": "response.mcp_call.completed",
  "sequence_number": 1,
  "item_id": "mcp_682d437d90a88191bf88cd03aae0c3e503937d5f622d7a90",
  "output_index": 0
}
```

response.mcp_call.failed



Emitted when an MCP tool call has failed.

item_id string

The ID of the MCP tool call item that failed.

output_index integer

The index of the output item that failed.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_call.failed'.

```
OBJECT response.mcp_call.failed
{
  "type": "response.mcp_call.failed",
  "sequence_number": 1,
  "item_id": "mcp_682d437d90a88191bf88cd03aae0c3e503937d5f622d7a90",
  "output_index": 0
}
```

response.mcp_call.in_progress



Emitted when an MCP tool call is in progress.

item_id string

The unique identifier of the MCP tool call item being processed.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_call.in_progress'.

```
OBJECT response.mcp_call.in_progress

{
  "type": "response.mcp_call.in_progress",
  "sequence_number": 1,
  "output_index": 0,
  "item_id": "mcp_682d437d90a88191bf88cd03aae0c3e503937d5f622d7a90"
}
```



response.mcp_list_tools.completed



Emitted when the list of available MCP tools has been successfully retrieved.

item_id string

The ID of the MCP tool call item that produced this output.

output_index integer

The index of the output item that was processed.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_list_tools.completed'.

```
OBJECT response.mcp_list_tools.completed
```

```
{
```

```
  "type": "response.mcp_list_tools.completed",
  "sequence_number": 1,
  "output_index": 0,
```

```
"item_id": "mcpl_682d4379df088191886b70f4ec39f90403937d5f622d7a90"  
}
```

response.mcp_list_tools.failed

ⓘ

Emitted when the attempt to list available MCP tools has failed.

item_id string

The ID of the MCP tool call item that failed.

output_index integer

The index of the output item that failed.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_list_tools.failed'.

OBJECT response.mcp_list_tools.failed

```
{  
  "type": "response.mcp_list_tools.failed",  
  "sequence_number": 1,  
  "output_index": 0,  
  "item_id": "mcpl_682d4379df088191886b70f4ec39f90403937d5f622d7a90"  
}
```

response.mcp_list_tools.in_progress



Emitted when the system is in the process of retrieving the list of available MCP tools.

item_id string

The ID of the MCP tool call item that is being processed.

output_index integer

The index of the output item that is being processed.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.mcp_list_tools.in_progress'.

```
OBJECT response.mcp_list_tools.in_progress

{
  "type": "response.mcp_list_tools.in_progress",
  "sequence_number": 1,
  "output_index": 0,
  "item_id": "mcpl_682d4379df088191886b70f4ec39f90403937d5f622d7a90"
}
```



response.code_interpreter_call.in_progress



Emitted when a code interpreter call is in progress.

item_id string

The unique identifier of the code interpreter tool call item.

output_index integer

The index of the output item in the response for which the code interpreter call is in progress.

sequence_number integer

The sequence number of this event, used to order streaming events.

type string

The type of the event. Always `response.code_interpreter_call.in_progress`.

OBJECT `response.code_interpreter_call.in_progress`

```
{  
  "type": "response.code_interpreter_call.in_progress",  
  "output_index": 0,  
  "item_id": "ci_12345",  
  "sequence_number": 1  
}
```

response.code_interpreter_call.interpreting

Emitted when the code interpreter is actively interpreting the code snippet.

item_id string

The unique identifier of the code interpreter tool call item.

output_index integer

The index of the output item in the response for which the code interpreter is interpreting code.

sequence_number integer

The sequence number of this event, used to order streaming events.

type string

The type of the event. Always `response.code_interpreter_call.interpreting`.

OBJECT `response.code_interpreter_call.interpreting`

```
{  
  "type": "response.code_interpreter_call.interpreting",  
  "output_index": 4,  
  "item_id": "ci_12345",  
  "sequence_number": 1  
}
```

response.code_interpreter_call.completed

🔗

Emitted when the code interpreter call is completed.

item_id string

The unique identifier of the code interpreter tool call item.

output_index integer

The index of the output item in the response for which the code interpreter call is completed.

sequence_number integer

The sequence number of this event, used to order streaming events.

type string

The type of the event. Always `response.code_interpreter_call.completed`.

OBJECT `response.code_interpreter_call.completed`

```
{  
  "type": "response.code_interpreter_call.completed",  
  "output_index": 5,  
  "item_id": "ci_12345",  
  "sequence_number": 1  
}
```



response.code_interpreter_call_code.delta



Emitted when a partial code snippet is streamed by the code interpreter.

delta string

The partial code snippet being streamed by the code interpreter.

item_id string

The unique identifier of the code interpreter tool call item.

output_index integer

The index of the output item in the response for which the code is being streamed.

sequence_number integer

The sequence number of this event, used to order streaming events.

type string

The type of the event. Always `response.code_interpreter_call_code.delta`.

OBJECT `response.code_interpreter_call_code.delta`

```
{  
  "type": "response.code_interpreter_call_code.delta",  
  "output_index": 0,  
  "item_id": "ci_12345",  
  "delta": "print('Hello, world')",  
  "sequence_number": 1  
}
```

response.code_interpreter_call_code.done

Emitted when the code snippet is finalized by the code interpreter.

code string

The final code snippet output by the code interpreter.

item_id string

The unique identifier of the code interpreter tool call item.

output_index integer

The index of the output item in the response for which the code is finalized.

sequence_number integer

The sequence number of this event, used to order streaming events.

type string

The type of the event. Always `response.code_interpreter_call_code.done`.

```
OBJECT response.code_interpreter_call_code.done

{
  "type": "response.code_interpreter_call_code.done",
  "output_index": 3,
  "item_id": "ci_12345",
  "code": "print('done')",
  "sequence_number": 1
}
```

response.output_text.annotation.added

Emitted when an annotation is added to output text content.

annotation object

The annotation object being added. (See annotation schema for details.)

annotation_index integer

The index of the annotation within the content part.

content_index integer

The index of the content part within the output item.

item_id string

The unique identifier of the item to which the annotation is being added.

output_index integer

The index of the output item in the response's output array.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always 'response.output_text.annotation.added'.

OBJECT response.output_text.annotation.added

```
{  
  "type": "response.output_text.annotation.added",  
  "item_id": "item-abc",  
  "output_index": 0,  
  "content_index": 0,  
  "annotation_index": 0,  
  "annotation": {  
    "type": "text_annotation",  
    "text": "This is a test annotation",  
    "start": 0,  
    "end": 10  
  },  
  "sequence_number": 1  
}
```

response.queued



Emitted when a response is queued and waiting to be processed.

response object

The full response object that is queued.

› Show properties

sequence_number integer

The sequence number for this event.

type string

The type of the event. Always 'response.queued'.

OBJECT response.queued

```
{  
  "type": "response.queued",  
  "response": {  
    "id": "res_123",  
    "status": "queued",  
    "content": ""  
  }  
}
```

```
"created_at": "2021-01-01T00:00:00Z",
"updated_at": "2021-01-01T00:00:00Z"
},
"sequence_number": 1
}
```



response.custom_tool_call_input.delta



Event representing a delta (partial update) to the input of a custom tool call.

delta string

The incremental input data (delta) for the custom tool call.

item_id string

Unique identifier for the API item associated with this event.

output_index integer

The index of the output this delta applies to.

sequence_number integer

The sequence number of this event.

type string

The event type identifier.

```
OBJECT response.custom_tool_call_input.delta
```

```
{  
  "type": "response.custom_tool_call_input.delta",  
  "output_index": 0,  
  "item_id": "ctc_1234567890abcdef",  
  "delta": "partial input text"  
}
```

response.custom_tool_call_input.done

⌚

Event indicating that input for a custom tool call is complete.

input string

The complete input data for the custom tool call.

item_id string

Unique identifier for the API item associated with this event.

output_index integer

The index of the output this event applies to.

sequence_number integer

The sequence number of this event.

type string

The event type identifier.

```
OBJECT response.custom_tool_call_input.done

{
  "type": "response.custom_tool_call_input.done",
  "output_index": 0,
  "item_id": "ctc_1234567890abcdef",
  "input": "final complete input text"
}
```

error



Emitted when an error occurs.

code string

The error code.

message string

The error message.

param string

The error parameter.

sequence_number integer

The sequence number of this event.

type string

The type of the event. Always `error`.

OBJECT `error`

{

`"type": "error",`
 `"code": "ERR_SOMETHING",`

```
"message": "Something went wrong",
"param": null,
"sequence_number": 1
}
```



PREVIOUS
< **Conversations**

NEXT ▾

Webhook Events



Webhooks are HTTP requests sent by OpenAI to a URL you specify when certain events happen during the course of API usage.

[Learn more about webhooks.](#)



response.completed



Sent when a background response has been completed.

created_at integer

The Unix timestamp (in seconds) of when the model response was completed.

data object

Event data payload.

> Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `response.completed`.

OBJECT `response.completed`

```
{  
  "id": "evt_abc123",  
  "type": "response.completed",  
  "created_at": 1719168000,  
  "data": {  
    "id": "resp_abc123"  
  }  
}
```

response.cancelled



Sent when a background response has been cancelled.

created_at integer

The Unix timestamp (in seconds) of when the model response was cancelled.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `response.cancelled`.

OBJECT `response.cancelled`

{

`"id": "evt_abc123",`

```
"type": "response.cancelled",
"created_at": 1719168000,
"data": {
  "id": "resp_abc123"
}
```

response.failed



Sent when a background response has failed.

created_at integer

The Unix timestamp (in seconds) of when the model response failed.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `response.failed`.

OBJECT `response.failed`

```
{  
  "id": "evt_abc123",  
  "type": "response.failed",  
  "created_at": 1719168000,  
  "data": {  
    "id": "resp_abc123"  
  }  
}
```

response.incomplete



Sent when a background response has been interrupted.

created_at integer

The Unix timestamp (in seconds) of when the model response was interrupted.

data object

Event data payload.

> Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `response.incomplete`.

OBJECT `response.incomplete`

```
{  
  "id": "evt_abc123",  
  "type": "response.incomplete",  
  "created_at": 1719168000,  
  "data": {  
    "id": "resp_abc123"  
  }  
}
```



batch.completed



Sent when a batch API request has been completed.

created_at integer

The Unix timestamp (in seconds) of when the batch API request was completed.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `batch.completed`.

OBJECT `batch.completed`

```
{  
  "id": "evt_abc123",  
  "type": "batch.completed",  
  "created_at": 1719168000,  
  "data": {  
    "id": "batch_abc123"  
  }  
}
```

batch.cancelled



Sent when a batch API request has been cancelled.

created_at integer

The Unix timestamp (in seconds) of when the batch API request was cancelled.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `batch.cancelled`.

OBJECT `batch.cancelled`

```
{  
  "id": "evt_abc123",  
  "type": "batch.cancelled",  
  "created_at": 1719168000,  
  "data": {  
    "id": "batch_abc123"  
  }  
}
```

batch.expired



Sent when a batch API request has expired.

created_at integer

The Unix timestamp (in seconds) of when the batch API request expired.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `batch.expired`.

OBJECT `batch.expired`

```
{  
  "id": "evt_abc123",  
  "type": "batch.expired",  
  "created_at": 1719168000,  
  "data": {  
    "id": "batch_abc123"  
  }  
}
```

batch.failed



Sent when a batch API request has failed.

created_at integer

The Unix timestamp (in seconds) of when the batch API request failed.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `batch.failed`.

OBJECT batch.failed

```
{  
  "id": "evt_abc123",  
  "type": "batch.failed",  
  "created_at": 1719168000,  
  "data": {  
    "id": "batch_abc123"  
  }  
}
```



fine_tuning.job.succeeded



Sent when a fine-tuning job has succeeded.

created_at integer

The Unix timestamp (in seconds) of when the fine-tuning job succeeded.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `fine_tuning.job.succeeded`.

OBJECT `fine_tuning.job.succeeded`

```
{  
  "id": "evt_abc123",  
  "type": "fine_tuning.job.succeeded",  
  "created_at": 1719168000,  
  "data": {  
    "id": "ftjob_abc123"  
  }  
}
```

fine_tuning.job.failed

Sent when a fine-tuning job has failed.

created_at integer

The Unix timestamp (in seconds) of when the fine-tuning job failed.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `fine_tuning.job.failed`.

OBJECT fine_tuning.job.failed

```
{  
  "id": "evt_abc123",  
  "type": "fine_tuning.job.failed",  
  "created_at": 1719168000,  
  "data": {  
    "id": "ftjob_abc123"  
  }  
}
```

fine_tuning.job.cancelled



Sent when a fine-tuning job has been cancelled.

created_at integer

The Unix timestamp (in seconds) of when the fine-tuning job was cancelled.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `fine_tuning.job.cancelled`.

OBJECT `fine_tuning.job.cancelled`

```
{  
  "id": "evt_abc123",  
  "type": "fine_tuning.job.cancelled",  
  "created_at": 1719168000,  
  "data": {  
    "id": "ftjob_abc123"  
  }  
}
```



eval.run.succeeded



Sent when an eval run has succeeded.

created_at integer

The Unix timestamp (in seconds) of when the eval run succeeded.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `eval.run.succeeded`.

OBJECT `eval.run.succeeded`

```
{
```

```
  "id": "evt_abc123",
```

```
"type": "eval.run.succeeded",
"created_at": 1719168000,
"data": {
  "id": "evalrun_abc123"
}
```

eval.run.failed



Sent when an eval run has failed.

created_at integer

The Unix timestamp (in seconds) of when the eval run failed.

data object

Event data payload.

› Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `eval.run.failed`.

OBJECT `eval.run.failed`

```
{  
  "id": "evt_abc123",  
  "type": "eval.run.failed",  
  "created_at": 1719168000,  
  "data": {  
    "id": "evalrun_abc123"  
  }  
}
```

eval.run.canceled



Sent when an eval run has been canceled.

created_at integer

The Unix timestamp (in seconds) of when the eval run was canceled.

data object

Event data payload.

> Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `eval.run.canceled`.

OBJECT `eval.run.canceled`

```
{  
  "id": "evt_abc123",  
  "type": "eval.run.canceled",  
  "created_at": 1719168000,  
  "data": {  
    "id": "evalrun_abc123"  
  }  
}
```



realtime.call.incoming



Sent when Realtime API Receives a incoming SIP call.

created_at integer

The Unix timestamp (in seconds) of when the model response was completed.

data object

Event data payload.

> Show properties

id string

The unique ID of the event.

object string

The object of the event. Always `event`.

type string

The type of the event. Always `realtime.call.incoming`.

OBJECT realtime.call.incoming

```
{  
  "id": "evt_abc123",  
  "type": "realtime.call.incoming",  
  "created_at": 1719168000,  
  "data": {  
    "call_id": "rtc_479a275623b54bdb9b6fbae2f7cbd408",  
    "sip_headers": [  
      {"name": "Max-Forwards", "value": "63"},  
      {"name": "CSeq", "value": "851287 INVITE"},  
      {"name": "Content-Type", "value": "application/sdp"},  
    ]  
  }  
}
```

< PREVIOUS
Streaming events

NEXT

Audio



Learn how to turn audio into text or text into audio.

Related guide: [Speech to text](#)

Create speech



```
POST https://api.openai.com/v1/audio/speech
```

Generates audio from the input text.

Request body

input string Required

The text to generate audio for. The maximum length is 4096 characters.

model string Required

One of the available [TTS models](#): `tts-1`, `tts-1-hd`, `gpt-4o-mini-tts`, or `gpt-4o-mini-tts-2025-12-15`.

voice string or object Required

The voice to use when generating the audio. Supported built-in voices are `alloy`, `ash`, `ballad`, `coral`, `echo`, `fable`, `onyx`, `nova`, `sage`, `shimmer`, `verse`, `marin`, and `cedar`. You may also provide a custom voice object with an `id`, for example `{ "id": "voice_1234" }`. Previews of the voices are available in the [Text to speech guide](#).

› Show possible types

instructions string Optional

Control the voice of your generated audio with additional instructions. Does not work with `tts-1` or `tts-1-hd`.

response_format string Optional Defaults to mp3

The format to audio in. Supported formats are `mp3`, `opus`, `aac`, `flac`, `wav`, and `pcm`.

speed number Optional Defaults to 1

The speed of the generated audio. Select a value from `0.25` to `4.0`. `1.0` is the default.

stream_format string Optional Defaults to audio

The format to stream the audio in. Supported formats are `sse` and `audio`. `sse` is not supported for `tts-1` or `tts-1-hd`.

Returns

The audio file content or a [stream of audio events](#).

Default**SSE Stream Format**

Example request

```
from pathlib import Path
import openai

speech_file_path = Path(__file__).parent / "speech.mp3"
with openai.audio.speech.with_streaming_response.create(
    model="gpt-4o-mini-tts",
    voice="alloy",
    input="The quick brown fox jumped over the lazy dog."
) as response:
    response.stream_to_file(speech_file_path)
```

Create voice



POST <https://api.openai.com/v1/audio/voices>

Create a custom voice you can use for audio output (for example, in Text-to-Speech and the Realtime API). This requires an audio sample and a previously uploaded consent recording.

See the [custom voices guide](#) for requirements and best practices. Custom voices are limited to eligible customers.

Request body

audio_sample file Required

The sample audio recording file. Maximum size is 10 MiB.

Supported MIME types: `audio/mpeg`, `audio/wav`, `audio/x-wav`, `audio/ogg`, `audio/aac`, `audio/flac`,
`audio/webm`, `audio/mp4`.

consent string Required

The consent recording ID (for example, `cons_1234`).

name string Required

The name of the new voice.

Returns

The created voice.

Example request

```
curl https://api.openai.com/v1/audio/voices \
-X POST \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-F "name=My new voice" \
```

Create voice consent



```
POST https://api.openai.com/v1/audio/voice_consents
```

Upload a consent recording that authorizes creation of a custom voice.

See the [custom voices guide](#) for requirements and best practices. Custom voices are limited to eligible customers.

Request body

language string Required

The BCP 47 language tag for the consent phrase (for example, `en-US`).

name string Required

The label to use for this consent recording.

recording file Required

The consent audio recording file. Maximum size is 10 MiB.

Supported MIME types: `audio/mpeg` , `audio/wav` , `audio/x-wav` , `audio/ogg` , `audio/aac` , `audio/flac` , `audio/webm` , `audio/mp4` .

Returns

The created voice consent recording metadata.

Example request

```
curl https://api.openai.com/v1/audio/voice_consents \
-X POST \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-F "name=John Doe" \
-F "language=en-US" \
-F "recording=@$HOME/consent_recording.wav;type=audio/x-wav"
```

List voice consents



```
GET https://api.openai.com/v1/audio/voice_consents
```

List consent recordings available to your organization for creating custom voices.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with `obj_foo`, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

Returns

A paginated list of voice consent recordings.

Example request

```
curl https://api.openai.com/v1/audio/voice_consents?limit=20 \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Retrieve voice consent



```
GET https://api.openai.com/v1/audio/voice_consents/{consent_id}
```

Retrieve consent recording metadata used for creating custom voices.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Path parameters

consent_id string Required

The ID of the consent recording to retrieve.

Returns

The voice consent recording metadata.

Example request

```
curl https://api.openai.com/v1/audio/voice_consents/cons_1234 \  
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Update voice consent



```
POST https://api.openai.com/v1/audio/voice_consents/{consent_id}
```

Update consent recording metadata used for creating custom voices. This endpoint updates metadata only and does not replace the underlying audio.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Path parameters

consent_id string Required

The ID of the consent recording to update.

Request body

name string Required

The updated label for this consent recording.

Returns

The updated voice consent recording metadata.

Example request

```
curl https://api.openai.com/v1/audio/voice_consents/cons_1234 \
-X POST \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
  "name": "John Doe"
}'
```

Delete voice consent



```
DELETE https://api.openai.com/v1/audio/voice_consents/{consent_id}
```

Delete a consent recording that was uploaded for creating custom voices.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Path parameters

consent_id string Required

The ID of the consent recording to delete.

Returns

A deletion confirmation.

Example request

```
curl https://api.openai.com/v1/audio/voice_consents/cons_1234 \
-X DELETE \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Create transcription



```
POST https://api.openai.com/v1/audio/transcriptions
```

Transcribes audio into the input language.

Request body

file file Required

The audio file object (not file name) to transcribe, in one of these formats: flac, mp3, mp4, mpeg, mpg, m4a, ogg, wav, or webm.

model string Required

ID of the model to use. The options are `gpt-4o-transcribe`, `gpt-4o-mini-transcribe`, `gpt-4o-mini-transcribe-2025-12-15`, `whisper-1` (which is powered by our open source Whisper V2 model), and `gpt-4o-transcribe-diarize`.

chunking_strategy "auto" or object Optional

Controls how the audio is cut into chunks. When set to `"auto"`, the server first normalizes loudness and then uses voice activity detection (VAD) to choose boundaries. `server_vad` object can be provided to tweak VAD detection parameters manually. If unset, the audio is transcribed as a single block. Required when using `gpt-4o-transcribe-diarize` for inputs longer than 30 seconds.

› Show possible types

include array Optional

Additional information to include in the transcription response. `logprobs` will return the log probabilities of the tokens in the response to understand the model's confidence in the transcription. `logprobs` only works with `response_format` set to `json` and only with the models `gpt-4o-transcribe`, `gpt-4o-mini-transcribe`, and `gpt-4o-mini-transcribe-2025-12-15`. This field is not supported when using `gpt-4o-transcribe-diarize`.

known_speaker_names array Optional

Optional list of speaker names that correspond to the audio samples provided in `known_speaker_references[]`. Each entry should be a short identifier (for example `customer` or `agent`). Up to 4 speakers are supported.

known_speaker_references array Optional

Optional list of audio samples (as [data URLs](#)) that contain known speaker references matching `known_speaker_names[]`. Each sample must be between 2 and 10 seconds, and can use any of the same input audio formats supported by `file`.

language string Optional

The language of the input audio. Supplying the input language in [ISO-639-1](#) (e.g. `en`) format will improve accuracy and latency.

prompt string Optional

An optional text to guide the model's style or continue a previous audio segment. The `prompt` should match the audio language. This field is not supported when using `gpt-4o-transcribe-diarize`.

response_format string Optional Defaults to json

The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, `vtt`, or `diarized_json`. For `gpt-4o-transcribe` and `gpt-4o-mini-transcribe`, the only supported format is `json`. For `gpt-4o-transcribe-diarize`, the supported formats are `json`, `text`, and `diarized_json`, with `diarized_json` required to receive speaker annotations.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section of the Speech-to-Text guide](#) for more information.

Note: Streaming is not supported for the `whisper-1` model and will be ignored.

temperature number Optional Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. If set to 0, the model will use `log_probability` to automatically increase the temperature until certain thresholds are hit.

timestamp_granularities array Optional Defaults to segment

The timestamp granularities to populate for this transcription. `response_format` must be set `verbose_json` to use timestamp granularities. Either or both of these options are supported: `word`, or `segment`. Note: There is no additional latency for segment timestamps, but generating word timestamps incurs additional latency. This option is not available for `gpt-4o-transcribe-diarize`.

Returns

The `transcription object`, a `diarized transcription object`, a `verbose transcription object`, or a `stream of transcript events`.

Default **Diarization** **Streaming** **Logprobs** **Word timestamps** **Segment timestamps**

Example request

```
from openai import OpenAI
client = OpenAI()

audio_file = open("speech.mp3", "rb")
transcript = client.audio.transcriptions.create(
    model="gpt-4o-transcribe",
```

Response

```
{  
  "text": "Imagine the wildest idea that you've ever had, and you're curious about how it might sc  
  "usage": {  
    "type": "tokens",  
    "input_tokens": 14,  
    "input_token_details": {  
      "text_tokens": 0,  
      "audio_tokens": 14  
    },  
    "output_tokens": 45,  
    "total_tokens": 59  
  }  
}
```

Create translation



POST <https://api.openai.com/v1/audio/translations>

Translates audio into English.

Request body

file file Required

The audio file object (not file name) translate, in one of these formats: flac, mp3, mp4, mpeg, mpg, m4a, ogg, wav, or webm.

model string or "whisper-1" Required

ID of the model to use. Only `whisper-1` (which is powered by our open source Whisper V2 model) is currently available.

prompt string Optional

An optional text to guide the model's style or continue a previous audio segment. The prompt should be in English.

response_format string Optional Defaults to json

The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, or `vtt`.

temperature number Optional Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. If set to 0, the model will use log probability to automatically increase the temperature until certain thresholds are hit.

Returns

The translated text.

Example request

```
from openai import OpenAI
client = OpenAI()

audio_file = open("speech.mp3", "rb")
transcript = client.audio.translations.create(
    model="whisper-1",
    file=audio_file
)
```

Response

```
{
  "text": "Hello, my name is Wolfgang and I come from Germany. Where are you heading today?"
}
```

The voice object



A custom voice that can be used for audio output.

created_at integer

The Unix timestamp (in seconds) for when the voice was created.

id string

The voice identifier, which can be referenced in API endpoints.

name string

The name of the voice.

object string

The object type, which is always `audio.voice`.

OBJECT The voice object

```
{  
  "object": "audio.voice",  
  "id": "voice_123abc",  
  "name": "My new voice",  
  "created_at": 1734220800  
}
```

The voice consent object



A consent recording used to authorize creation of a custom voice.

created_at integer

The Unix timestamp (in seconds) for when the consent recording was created.

id string

The consent recording identifier.

language string

The BCP 47 language tag for the consent phrase (for example, `en-US`).

name string

The label provided when the consent recording was uploaded.

object string

The object type, which is always `audio.voice_consent` .

OBJECT The voice consent object

```
{  
  "object": "audio.voice_consent",  
  "id": "cons_1234",  
  "name": "John Doe",  
  "language": "en-US",  
  "created_at": 1734220800  
}
```

The voice consent list object



data array

> Show properties

first_id string

has_more boolean

last_id string

object string

OBJECT The voice consent list object

```
{  
  "object": "list",  
  "data": [  
    {  
      "object": "audio.voice_consent",  
      "id": "cons_1234",  
      "status": "active",  
      "start_time": "2023-10-01T12:00:00Z",  
      "end_time": "2023-10-01T13:00:00Z",  
      "duration": "PT1H",  
      "volume": 1.0,  
      "pitch": 0.0,  
      "rate": 1.0,  
      "language": "en-US",  
      "text": "Please confirm you are a human.",  
      "audio": "https://example.com/audio.mp3",  
      "consent": "https://example.com/consent.html",  
      "callback": "https://example.com/callback",  
      "error": null  
    }  
  ]  
}
```

```
        "name": "John Doe",
        "language": "en-US",
        "created_at": 1734220800
    }
],
"first_id": "cons_1234",
"last_id": "cons_1234",
"has_more": false
}
```

The voice consent deletion object



deleted boolean

id string

The consent recording identifier.

object string

OBJECT The voice consent deletion object

```
{  
  "object": "audio.voice_consent",  
  "id": "cons_1234",  
  "deleted": true  
}
```

The transcription object (JSON)



Represents a transcription response returned by model, based on the provided input.

logprobs array

The log probabilities of the tokens in the transcription. Only returned with the models `gpt-4o-transcribe` and `gpt-4o-mini-transcribe` if `logprobs` is added to the `include` array.

> Show properties

text string

The transcribed text.

usage object

Token usage statistics for the request.

> Show possible types

OBJECT The transcription object (JSON)

```
{  
  "text": "Imagine the wildest idea that you've ever had, and you're curious about how it might sc  
  "usage": {  
    "type": "tokens",  
    "input_tokens": 14,  
    "input_token_details": {  
      "text_tokens": 10,  
      "audio_tokens": 4  
    },  
    "output_tokens": 101,  
    "total_tokens": 115  
  }  
}
```

The transcription object (Diarized JSON)



Represents a diarized transcription response returned by the model, including the combined transcript and speaker-segment annotations.

duration number

Duration of the input audio in seconds.

segments array

Segments of the transcript annotated with timestamps and speaker labels.

› Show properties

task string

The type of task that was run. Always `transcribe`.

text string

The concatenated transcript text for the entire audio input.

usage object

Token or duration usage statistics for the request.

› Show possible types

OBJECT The transcription object (Diarized JSON)

```
{  
  "task": "transcribe",  
  "duration": 42.7,  
  "text": "Agent: Thanks for calling OpenAI support.\nCustomer: Hi, I need help with diarization."  
  "segments": [  
    {  
      "type": "transcript.text.segment",  
      "id": "seg_001",  
      "start": 0.0,
```

```
"end": 5.2,  
  "text": "Thanks for calling OpenAI support.",  
  "speaker": "agent"  
},  
{  
  "type": "transcript.text.segment",  
  "id": "seg_002",  
  "start": 5.2,  
  "end": 12.8,  
  "text": "Hi, I need help with diarization.",  
  "speaker": "A"  
}  
,  
  "usage": {  
    "type": "duration",  
    "seconds": 43  
}  
}
```

The transcription object (Verbose JSON)

🔗

Represents a verbose json transcription response returned by model, based on the provided input.

duration number

The duration of the input audio.

language string

The language of the input audio.

segments array

Segments of the transcribed text and their corresponding details.

› Show properties

text string

The transcribed text.

usage object

Usage statistics for models billed by audio input duration.

› Show properties

words array

Extracted words and their corresponding timestamps.

› Show properties

OBJECT The transcription object (Verbose JSON)

```
{  
  "task": "transcribe",  
  "language": "english",
```

```
"duration": 8.470000267028809,  
"text": "The beach was a popular spot on a hot summer day. People were swimming in the ocean, bu  
"segments": [  
  {  
    "id": 0,  
    "seek": 0,  
    "start": 0.0,  
    "end": 3.319999933242798,  
    "text": " The beach was a popular spot on a hot summer day.",  
    "tokens": [  
      50364, 440, 7534, 390, 257, 3743, 4008, 322, 257, 2368, 4266, 786, 13, 50530  
    ],  
    "temperature": 0.0,  
    "avg_logprob": -0.2860786020755768,  
    "compression_ratio": 1.2363636493682861,  
    "no_speech_prob": 0.00985979475080967  
  },  
  ...  
],  
"usage": {  
  "type": "duration",  
  "seconds": 9  
}
```

Stream Event (speech.audio.delta)



Emitted for each chunk of audio data generated during speech synthesis.

audio string

A chunk of Base64-encoded audio data.

type string

The type of the event. Always `speech.audio.delta`.

OBJECT Stream Event (speech.audio.delta)

```
{  
  "type": "speech.audio.delta",  
  "audio": "base64-encoded-audio-data"  
}
```

Stream Event (speech.audio.done)



Emitted when the speech synthesis is complete and all audio has been streamed.

type string

The type of the event. Always `speech.audio.done`.

usage object

Token usage statistics for the request.

› Show properties

OBJECT Stream Event (`speech.audio.done`)

```
{  
  "type": "speech.audio.done",  
  "usage": {  
    "input_tokens": 14,  
    "output_tokens": 101,  
    "total_tokens": 115  
  }  
}
```

Stream Event (`transcript.text.delta`)

🔗

Emitted when there is an additional text delta. This is also the first event emitted when the transcription starts. Only emitted when you [create a transcription](#) with the `Stream` parameter set to `true`.

delta string

The text delta that was additionally transcribed.

logprobs array

The log probabilities of the delta. Only included if you [create a transcription](#) with the `include[]` parameter set to `logprobs`.

› Show properties

segment_id string

Identifier of the diarized segment that this delta belongs to. Only present when using `gpt-4o-transcribe-diarize`.

type string

The type of the event. Always `transcript.text.delta`.

OBJECT Stream Event (`transcript.text.delta`)

```
{  
  "type": "transcript.text.delta",  
  "delta": " wonderful"  
}
```

Stream Event (`transcript.text.segment`)



Emitted when a diarized transcription returns a completed segment with speaker information. Only emitted when you create a transcription with `stream` set to `true` and `response_format` set to `diarized_json`.

end number

End timestamp of the segment in seconds.

id string

Unique identifier for the segment.

speaker string

Speaker label for this segment.

start number

Start timestamp of the segment in seconds.

text string

Transcript text for this segment.

type string

The type of the event. Always `transcript.text.segment`.

OBJECT Stream Event (`transcript.text.segment`)

{

 "**type**": "`transcript.text.segment`",

```
"id": "seg_002",
"start": 5.2,
"end": 12.8,
"text": "Hi, I need help with diarization.",
"speaker": "A"
}
```

Stream Event (transcript.text.done)



Emitted when the transcription is complete. Contains the complete transcription text. Only emitted when you [create a transcription](#) with the `Stream` parameter set to `true`.

logprobs array

The log probabilities of the individual tokens in the transcription. Only included if you [create a transcription](#) with the `include[]` parameter set to `logprobs`.

> Show properties

text string

The text that was transcribed.

type string

The type of the event. Always `transcript.text.done`.

usage object

Usage statistics for models billed by token usage.

› Show properties

OBJECT Stream Event (`transcript.text.done`)

```
{  
  "type": "transcript.text.done",  
  "text": "I see skies of blue and clouds of white, the bright blessed days, the dark sacred night  
  "usage": {  
    "type": "tokens",  
    "input_tokens": 14,  
    "input_token_details": {  
      "text_tokens": 10,  
      "audio_tokens": 4  
    },  
    "output_tokens": 31,  
    "total_tokens": 45  
  }  
}
```

< PREVIOUS
Webhook Events

NEXT >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Audio

Learn how to turn audio into text or text into audio.

Related guide: [Speech to text](#)

Create speech

```
POST https://api.openai.com/v1/audio/speech
```

Generates audio from the input text.

Request body

input string Required

The text to generate audio for. The maximum length is 4096 characters.

model string Required

One of the available [TTS models](#): `tts-1` , `tts-1-hd` , `gpt-4o-mini-tts` , or
`gpt-4o-mini-tts-2025-12-15` .

voice string or object Required

The voice to use when generating the audio. Supported built-in voices are `alloy` , `ash` , `ballad` ,
`coral` , `echo` , `fable` , `onyx` , `nova` , `sage` , `shimmer` , `verse` , `marin` , and `cedar` . You may also provide a custom voice object with an `id` , for example `{ "id": "voice_1234" }` . Previews of the voices are available in the [Text to speech guide](#).

› Show possible types

instructions string Optional

Control the voice of your generated audio with additional instructions. Does not work with `tts-1` or `tts-1-hd` .

response_format string Optional Defaults to mp3

The format to audio in. Supported formats are `mp3` , `opus` , `aac` , `flac` , `wav` , and `pcm` .

speed number Optional Defaults to 1

The speed of the generated audio. Select a value from `0.25` to `4.0` . `1.0` is the default.

stream_format string Optional Defaults to audio

The format to stream the audio in. Supported formats are `sse` and `audio`. `sse` is not supported for `tts-1` or `tts-1-hd`.

Returns

The audio file content or a stream of audio events.

Default SSE Stream Format

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/audio/speech \
2   -H "Authorization: Bearer $OPENAI_API_KEY" \
3   -H "Content-Type: application/json" \
4   -d '{
5     "model": "gpt-4o-mini-tts",
6     "input": "The quick brown fox jumped over the lazy dog.",
7     "voice": "alloy",
8     "stream_format": "sse"
9   }'
```

Create voice

POST <https://api.openai.com/v1/audio/voices>

Create a custom voice you can use for audio output (for example, in Text-to-Speech and the Realtime API). This requires an audio sample and a previously uploaded consent recording.

See the [custom voices guide](#) for requirements and best practices. Custom voices are limited to eligible customers.

Request body

audio_sample file Required

The sample audio recording file. Maximum size is 10 MiB.

Supported MIME types: `audio/mpeg` , `audio/wav` , `audio/x-wav` , `audio/ogg` , `audio/aac` ,
`audio/flac` , `audio/webm` , `audio/mp4` .

consent string Required

The consent recording ID (for example, `cons_1234`).

name string Required

The name of the new voice.

Returns

The created voice.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/audio/voices \
2   -X POST \
3   -H "Authorization: Bearer $OPENAI_API_KEY" \
4   -F "name=My new voice" \
5   -F "consent=cons_1234" \
6   -F "audio_sample=@$HOME/audio_sample.wav;type=audio/x-wav"
```

Create voice consent

POST https://api.openai.com/v1/audio/voice_consents

Upload a consent recording that authorizes creation of a custom voice.

See the [custom voices guide](#) for requirements and best practices. Custom voices are limited to eligible customers.

Request body

language string Required

The BCP 47 language tag for the consent phrase (for example, `en-US`).

name string Required

The label to use for this consent recording.

recording file Required

The consent audio recording file. Maximum size is 10 MiB.

Supported MIME types: `audio/mpeg` , `audio/wav` , `audio/x-wav` , `audio/ogg` , `audio/aac` ,
`audio/flac` , `audio/webm` , `audio/mp4` .

Returns

The created voice consent recording metadata.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/audio/voice_consents \
2   -X POST \
3   -H "Authorization: Bearer $OPENAI_API_KEY" \
4   -F "name=John Doe" \
5   -F "language=en-US" \
6   -F "recording=@$HOME/consent_recording.wav;type=audio/x-wav"
```

List voice consents

```
GET https://api.openai.com/v1/audio/voice_consents
```

List consent recordings available to your organization for creating custom voices.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with `obj_foo`, your subsequent call can include

after=obj_foo in order to fetch the next page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

Returns

A paginated list of voice consent recordings.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/audio/voice_consents?limit=20 \
2 -H "Authorization: Bearer $OPENAI_API_KEY"
```

Retrieve voice consent

```
GET https://api.openai.com/v1/audio/voice_consents/{consent_id}
```

Retrieve consent recording metadata used for creating custom voices.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Path parameters

consent_id string Required

The ID of the consent recording to retrieve.

Returns

The voice consent recording metadata.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/audio/voice_consents/cons_1234 \
2   -H "Authorization: Bearer $OPENAI_API_KEY"
```

Update voice consent

```
POST https://api.openai.com/v1/audio/voice_consents/{consent_id}
```

Update consent recording metadata used for creating custom voices. This endpoint updates metadata only and does not replace the underlying audio.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Path parameters

consent_id string Required

The ID of the consent recording to update.

Request body

name string Required

The updated label for this consent recording.

Returns

The updated voice consent recording metadata.

Example request

curl ▾ 

```
1 curl https://api.openai.com/v1/audio/voice_consents/cons_1234 \
2   -X POST \
3   -H "Authorization: Bearer $OPENAI_API_KEY" \
4   -H "Content-Type: application/json" \
5   -d '{
6     "name": "John Doe"
7   }'
```

Delete voice consent

```
DELETE https://api.openai.com/v1/audio/voice_consents/{consent_id}
```

Delete a consent recording that was uploaded for creating custom voices.

See the [custom voices guide](#). Custom voices are limited to eligible customers.

Path parameters

consent_id string Required

The ID of the consent recording to delete.

Returns

A deletion confirmation.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/audio/voice_consents/cons_1234 \
2   -X DELETE \
3   -H "Authorization: Bearer $OPENAI_API_KEY"
```

Create transcription

POST <https://api.openai.com/v1/audio/transcriptions>

Transcribes audio into the input language.

Request body

file file Required

The audio file object (not file name) to transcribe, in one of these formats: flac, mp3, mp4, mpeg, mpga, m4a, ogg, wav, or webm.

model string Required

ID of the model to use. The options are `gpt-4o-transcribe`, `gpt-4o-mini-transcribe`, `gpt-4o-mini-transcribe-2025-12-15`, `whisper-1` (which is powered by our open source Whisper V2 model), and `gpt-4o-transcribe-diarize`.

chunking_strategy "auto" or object Optional

Controls how the audio is cut into chunks. When set to `"auto"`, the server first normalizes loudness and then uses voice activity detection (VAD) to choose boundaries. `server_vad` object can be provided to tweak VAD detection parameters manually. If unset, the audio is transcribed as a single block. Required when using `gpt-4o-transcribe-diarize` for inputs longer than 30 seconds.

› Show possible types

include array Optional

Additional information to include in the transcription response. `logprobs` will return the log probabilities of the tokens in the response to understand the model's confidence in the transcription. `logprobs` only works with `response_format` set to `json` and only with the models `gpt-4o-transcribe`, `gpt-4o-mini-transcribe`, and `gpt-4o-mini-transcribe-2025-12-15`. This field is not supported when using `gpt-4o-transcribe-diarize`.

known_speaker_names array Optional

Optional list of speaker names that correspond to the audio samples provided in `known_speaker_references[]`. Each entry should be a short identifier (for example `customer` or `agent`). Up to 4 speakers are supported.

known_speaker_references array Optional

Optional list of audio samples (as [data URLs](#)) that contain known speaker references matching `known_speaker_names[]`. Each sample must be between 2 and 10 seconds, and can use any of the same input audio formats supported by `file`.

language string Optional

The language of the input audio. Supplying the input language in [ISO-639-1](#) (e.g. `en`) format will improve accuracy and latency.

prompt string Optional

An optional text to guide the model's style or continue a previous audio segment. The `prompt` should match the audio language. This field is not supported when using `gpt-4o-transcribe-diarize`.

response_format string Optional Defaults to json

The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, `vtt`, or `diarized_json`. For `gpt-4o-transcribe` and `gpt-4o-mini-transcribe`, the only supported format is `json`. For `gpt-4o-transcribe-diarize`, the supported formats are `json`, `text`, and `diarized_json`, with `diarized_json` required to receive speaker annotations.

stream boolean Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section of the Speech-to-Text guide](#) for more information.

Note: Streaming is not supported for the `whisper-1` model and will be ignored.

temperature number Optional Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. If set to 0, the model will use [log_probability](#) to automatically increase the temperature until certain thresholds are hit.

timestamp_granularities array Optional Defaults to segment

The timestamp granularities to populate for this transcription. `response_format` must be set `verbose_json` to use timestamp granularities. Either or both of these options are supported: `word`, or `segment`. Note: There is no additional latency for segment timestamps, but generating word timestamps incurs additional latency. This option is not available for `gpt-4o-transcribe-diarize`.

Returns

The [transcription object](#), a [diarized transcription object](#), a [verbose transcription object](#), or a [stream of transcript events](#).

Default [Diarization](#) [Streaming](#) [Logprobs](#) [Word timestamps](#) [Segment timestamps](#)

Example request

python 

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 audio_file = open("speech.mp3", "rb")
5 transcript = client.audio.transcriptions.create(
6     model="gpt-4o-transcribe",
7     file=audio_file
8 )
```

Response



```
1 {
2     "text": "Imagine the wildest idea that you've ever had, and you're curious about h
3     "usage": {
4         "type": "tokens",
5         "input_tokens": 14,
6         "input_token_details": {
7             "text_tokens": 0,
8             "audio_tokens": 14
9         },
10        "output_tokens": 45,
11        "total_tokens": 59
12    }
13 }
```

Create translation

```
POST https://api.openai.com/v1/audio/translations
```

Translates audio into English.

Request body

file file Required

The audio file object (not file name) translate, in one of these formats: flac, mp3, mp4, mpeg, mpg, m4a, ogg, wav, or webm.

model string or "whisper-1" Required

ID of the model to use. Only `whisper-1` (which is powered by our open source Whisper V2 model) is currently available.

prompt string Optional

An optional text to guide the model's style or continue a previous audio segment. The `prompt` should be in English.

response_format string Optional Defaults to json

The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, or `vtt`.

temperature number Optional Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. If set to 0, the model will use log_probability to automatically increase the temperature until certain thresholds are hit.

Returns

The translated text.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 audio_file = open("speech.mp3", "rb")
5 transcript = client.audio.translations.create(
6     model="whisper-1",
7     file=audio_file
8 )
```

Response

⌚

```
1 {
2   "text": "Hello, my name is Wolfgang and I come from Germany. Where are you heading
3 }
```

The voice object

A custom voice that can be used for audio output.

created_at integer

The Unix timestamp (in seconds) for when the voice was created.

id string

The voice identifier, which can be referenced in API endpoints.

name string

The name of the voice.

object string

The object type, which is always `audio.voice`.

OBJECT The voice object



```
1 {
2   "object": "audio.voice",
3   "id": "voice_123abc",
4   "name": "My new voice",
5   "created_at": 1734220800
6 }
```

The voice consent object

A consent recording used to authorize creation of a custom voice.

created_at integer

The Unix timestamp (in seconds) for when the consent recording was created.

id string

The consent recording identifier.

language string

The BCP 47 language tag for the consent phrase (for example, `en-US`).

name string

The label provided when the consent recording was uploaded.

object string

The object type, which is always `audio.voice_consent` .

OBJECT The voice consent object



```
1 {
2   "object": "audio.voice_consent",
3   "id": "cons_1234",
4   "name": "John Doe",
5   "language": "en-US",
6   "created_at": 1734220800
7 }
```

The voice consent list object

data array

› Show properties

first_id string

has_more boolean

last_id string

object string

OBJECT The voice consent list object



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "object": "audio.voice_consent",
6              "id": "cons_1234",
7              "name": "John Doe",
8              "language": "en-US",
9              "created_at": 1734220800
10         }
11     ],
12     "first_id": "cons_1234",
13     "last_id": "cons_1234",
```

```
14  "has_more": false  
15 }
```

The voice consent deletion object

deleted boolean

id string

The consent recording identifier.

object string

OBJECT The voice consent deletion object



```
1 {
2   "object": "audio.voice_consent",
3   "id": "cons_1234",
4   "deleted": true
5 }
```

The transcription object (JSON)

Represents a transcription response returned by model, based on the provided input.

logprobs array

The log probabilities of the tokens in the transcription. Only returned with the models `gpt-4o-transcribe` and `gpt-4o-mini-transcribe` if `logprobs` is added to the `include` array.

› Show properties

text string

The transcribed text.

usage object

Token usage statistics for the request.

> Show possible types

OBJECT The transcription object (JSON)

2

```
1  {
2      "text": "Imagine the wildest idea that you've ever had, and you're curious about h
3      "usage": {
4          "type": "tokens",
5          "input_tokens": 14,
6          "input_token_details": {
7              "text_tokens": 10,
8              "audio_tokens": 4
9          },
10         "output_tokens": 101,
11         "total_tokens": 115
12     }
13 }
```

The transcription object (Diarized JSON)

Represents a diarized transcription response returned by the model, including the combined transcript and speaker-segment annotations.

duration number

Duration of the input audio in seconds.

segments array

Segments of the transcript annotated with timestamps and speaker labels.

› Show properties

task string

The type of task that was run. Always `transcribe`.

text string

The concatenated transcript text for the entire audio input.

usage object

Token or duration usage statistics for the request.

› Show possible types

OBJECT The transcription object (Diarized JSON)



```
1  {
2    "task": "transcribe",
```

```
3 "duration": 42.7,
4 "text": "Agent: Thanks for calling OpenAI support.\nCustomer: Hi, I need help with
5 "segments": [
6 {
7     "type": "transcript.text.segment",
8     "id": "seg_001",
9     "start": 0.0,
10    "end": 5.2,
11    "text": "Thanks for calling OpenAI support.",
12    "speaker": "agent"
13 },
14 {
15     "type": "transcript.text.segment",
16     "id": "seg_002",
17     "start": 5.2,
18     "end": 12.8,
19     "text": "Hi, I need help with diarization.",
20     "speaker": "A"
21 }
22 ],
23 "usage": {
24     "type": "duration",
25     "seconds": 43
26 }
27 }
```

The transcription object (Verbose JSON)

Represents a verbose json transcription response returned by model, based on the provided input.

duration number

The duration of the input audio.

language string

The language of the input audio.

segments array

Segments of the transcribed text and their corresponding details.

› Show properties

text string

The transcribed text.

usage object

Usage statistics for models billed by audio input duration.

› Show properties

words array

Extracted words and their corresponding timestamps.

› Show properties

OBJECT The transcription object (Verbose JSON)



```
1  {
2      "task": "transcribe",
3      "language": "english",
4      "duration": 8.470000267028809,
5      "text": "The beach was a popular spot on a hot summer day. People were swimming in",
6      "segments": [
7          {
8              "id": 0,
9              "seek": 0,
10             "start": 0.0,
11             "end": 3.319999933242798,
12             "text": " The beach was a popular spot on a hot summer day.",
13             "tokens": [
14                 50364, 440, 7534, 390, 257, 3743, 4008, 322, 257, 2368, 4266, 786, 13, 50536
15             ],
16             "temperature": 0.0,
17             "avg_logprob": -0.2860786020755768,
18             "compression_ratio": 1.2363636493682861,
19             "no_speech_prob": 0.00985979475080967
20         },
21         ...
22     ],
```

```
23 "usage": {  
24     "type": "duration",  
25     "seconds": 9  
26 }  
27 }
```

Stream Event (speech.audio.delta)

Emitted for each chunk of audio data generated during speech synthesis.

audio string

A chunk of Base64-encoded audio data.

type string

The type of the event. Always `speech.audio.delta`.

OBJECT Stream Event (speech.audio.delta)



```
1 {  
2     "type": "speech.audio.delta",
```

```
3   "audio": "base64-encoded-audio-data"  
4 }
```

Stream Event (speech.audio.done)

Emitted when the speech synthesis is complete and all audio has been streamed.

type string

The type of the event. Always `speech.audio.done`.

usage object

Token usage statistics for the request.

› Show properties

OBJECT Stream Event (speech.audio.done)



```
1 {  
2   "type": "speech.audio.done",  
3   "usage": {  
4     "input_tokens": 14,  
5     "output_tokens": 101,
```

```
6     "total_tokens": 115
7 }
8 }
```

Stream Event (transcript.text.delta)

Emitted when there is an additional text delta. This is also the first event emitted when the transcription starts. Only emitted when you [create a transcription](#) with the `stream` parameter set to `true`.

delta string

The text delta that was additionally transcribed.

logprobs array

The log probabilities of the delta. Only included if you [create a transcription](#) with the `include[]` parameter set to `logprobs`.

› Show properties

segment_id string

Identifier of the diarized segment that this delta belongs to. Only present when using `gpt-4o-transcribe-diarize`.

type string

The type of the event. Always `transcript.text.delta`.

OBJECT Stream Event (`transcript.text.delta`)



```
1 {
2   "type": "transcript.text.delta",
3   "delta": " wonderful"
4 }
```

Stream Event (`transcript.text.segment`)

Emitted when a diarized transcription returns a completed segment with speaker information. Only emitted when you create a transcription with `stream` set to `true` and `response_format` set to `diarized_json`.

end number

End timestamp of the segment in seconds.

id string

Unique identifier for the segment.

speaker string

Speaker label for this segment.

start number

Start timestamp of the segment in seconds.

text string

Transcript text for this segment.

type string

The type of the event. Always `transcript.text.segment`.

OBJECT Stream Event (`transcript.text.segment`)



```
1 {
2   "type": "transcript.text.segment",
3   "id": "seg_002",
4   "start": 5.2,
5   "end": 12.8,
6   "text": "Hi, I need help with diarization.",
```

```
7 "speaker": "A"  
8 }
```

Stream Event (transcript.text.done)

Emitted when the transcription is complete. Contains the complete transcription text. Only emitted when you [create a transcription](#) with the `stream` parameter set to `true`.

logprobs array

The log probabilities of the individual tokens in the transcription. Only included if you [create a transcription](#) with the `include[]` parameter set to `logprobs`.

› Show properties

text string

The text that was transcribed.

type string

The type of the event. Always `transcript.text.done`.

usage object

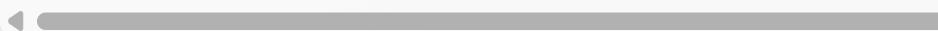
Usage statistics for models billed by token usage.

> Show properties

OBJECT Stream Event (transcript.text.done)



```
1  {
2    "type": "transcript.text.done",
3    "text": "I see skies of blue and clouds of white, the bright blessed days, the dar
4    "usage": {
5      "type": "tokens",
6      "input_tokens": 14,
7      "input_token_details": {
8        "text_tokens": 10,
9        "audio_tokens": 4
10     },
11     "output_tokens": 31,
12     "total_tokens": 45
13   }
14 }
```



PREVIOUS

Webhook Events

NEXT

Videos



Videos



Generate videos.

Create video



```
POST https://api.openai.com/v1/videos
```

Create a new video generation job from a prompt and optional reference assets.

Request body

prompt string Required

Text prompt that describes the video to generate.

input_reference file Optional

Optional image reference that guides generation.

model string Optional

The video generation model to use (allowed values: sora-2, sora-2-pro). Defaults to `sora-2`.

seconds string Optional

Clip duration in seconds (allowed values: 4, 8, 12). Defaults to 4 seconds.

size string Optional

Output resolution formatted as width x height (allowed values: 720x1280, 1280x720, 1024x1792, 1792x1024). Defaults to 720x1280.

Returns

Returns the newly created [video job](#).

Example request

```
from openai import OpenAI

client = OpenAI()
video = client.videos.create(
    prompt="A calico cat playing a piano on stage",
)
print(video.id)
```

Response

```
{  
  "id": "video_123",  
  "object": "video",  
  "model": "sora-2",  
  "status": "queued",  
  "progress": 0,  
  "created_at": 1712697600,  
  "size": "1024x1792",  
  "seconds": "8",  
  "quality": "standard"  
}
```

Remix video



POST https://api.openai.com/v1/videos/{video_id}/remix

Create a remix of a completed video using a refreshed prompt.

Path parameters

video_id string Required

The identifier of the completed video to remix.

Request body

prompt string Required

Updated text prompt that directs the remix generation.

Returns

Creates a remix of the specified video job using the provided prompt.

Example request

```
from openai import OpenAI

client = OpenAI()
video = client.videos.remix(
    video_id="video_123",
    prompt="Extend the scene with the cat taking a bow to the cheering audience",
)
print(video.id)
```

Response

```
{  
  "id": "video_456",  
  "object": "video",  
  "model": "sora-2",  
  "status": "queued",  
  "progress": 0,  
  "created_at": 1712698600,  
  "size": "720x1280",  
  "seconds": "8",  
  "remixed_from_video_id": "video_123"  
}
```

List videos



```
GET https://api.openai.com/v1/videos
```

List recently generated videos for the current project.

Query parameters

after string Optional

Identifier for the last item from the previous pagination request

limit integer Optional

Number of items to retrieve

order string Optional

Sort order of results by timestamp. Use `asc` for ascending order or `desc` for descending order.

Returns

Returns a paginated list of [video jobs](#) for the organization.

Example request

```
from openai import OpenAI

client = OpenAI()
page = client.videos.list()
page = page.data[0]
print(page.id)
```

Response

```
{
  "data": [
```

```
{  
  "id": "video_123",  
  "object": "video",  
  "model": "sora-2",  
  "status": "completed"  
}  
,  
"object": "list"  
}
```

Retrieve video



```
GET https://api.openai.com/v1/videos/{video_id}
```

Fetch the latest metadata for a generated video.

Path parameters

video_id string Required

The identifier of the video to retrieve.

Returns

Returns the video job matching the provided identifier.

Example request

```
from openai import OpenAI

client = OpenAI()
video = client.videos.retrieve(
    "video_123",
)
print(video.id)
```

Delete video



```
DELETE https://api.openai.com/v1/videos/{video_id}
```

Permanently delete a completed or failed video and its stored assets.

Path parameters

video_id string Required

The identifier of the video to delete.

Returns

Returns the deleted video job metadata.

Example request

```
from openai import OpenAI

client = OpenAI()
video = client.videos.delete(
    "video_123",
)
print(video.id)
```

Retrieve video content



```
GET https://api.openai.com/v1/videos/{video_id}/content
```

Download the generated video bytes or a derived preview asset.

Path parameters

video_id string Required

The identifier of the video whose media to download.

Query parameters

variant string Optional

Which downloadable asset to return. Defaults to the MP4 video.

Returns

Streams the rendered video content for the specified video job.

Example request

```
from openai import OpenAI  
  
client = OpenAI()  
response = client.videos.download_content()
```

```
video_id="video_123",  
)  
print(response)  
content = response.read()  
print(content)
```

Video job



Structured information describing a generated video job.

completed_at integer

Unix timestamp (seconds) for when the job completed, if finished.

created_at integer

Unix timestamp (seconds) for when the job was created.

error object

Error payload that explains why generation failed, if applicable.

› Show properties

expires_at integer

Unix timestamp (seconds) for when the downloadable assets expire, if set.

id string

Unique identifier for the video job.

model string

The video generation model that produced the job.

object string

The object type, which is always `video`.

progress integer

Approximate completion percentage for the generation task.

prompt string

The prompt that was used to generate the video.

remixed_from_video_id string

Identifier of the source video if this video is a remix.

seconds string

Duration of the generated clip in seconds.

size string

The resolution of the generated video.

status string

Current lifecycle status of the video job.

< PREVIOUS
Audio

NEXT

Images



Given a prompt and/or an input image, the model will generate a new image. Related guide:

[Image generation](#)

Create image



```
POST https://api.openai.com/v1/images/generations
```

Creates an image given a prompt. [Learn more.](#)

Request body

prompt string Required

A text description of the desired image(s). The maximum length is 32000 characters for the GPT image models, 1000 characters for `dall-e-2` and 4000 characters for `dall-e-3`.

background string or null Optional Defaults to auto

Allows to set transparency for the background of the generated image(s). This parameter is only supported for the GPT image models. Must be one of `transparent`, `opaque` or `auto` (default value). When `auto` is used, the model will automatically determine the best background for the image.

If `transparent`, the output format needs to support transparency, so it should be set to either `png` (default value) or `webp`.

model string Optional Defaults to dall-e-2

The model to use for image generation. One of `dall-e-2`, `dall-e-3`, or a GPT image model (`gpt-image-1`, `gpt-image-1-mini`, `gpt-image-1.5`). Defaults to `dall-e-2` unless a parameter specific to the GPT image models is used.

moderation string or null Optional Defaults to auto

Control the content-moderation level for images generated by the GPT image models. Must be either `low` for less restrictive filtering or `auto` (default value).

n integer or null Optional Defaults to 1

The number of images to generate. Must be between 1 and 10. For `dall-e-3`, only `n=1` is supported.

output_compression integer or null Optional Defaults to 100

The compression level (0-100%) for the generated images. This parameter is only supported for the GPT image models with the `webp` or `jpeg` output formats, and defaults to 100.

output_format string or null Optional Defaults to png

The format in which the generated images are returned. This parameter is only supported for the GPT image models. Must be one of `png`, `jpeg`, or `webp`.

partial_images integer Optional Defaults to 0

The number of partial images to generate. This parameter is used for streaming responses that return partial images. Value must be between 0 and 3. When set to 0, the response will be a single image sent in one streaming event.

Note that the final image may be sent before the full number of partial images are generated if the full image is generated more quickly.

quality string or null Optional Defaults to auto

The quality of the image that will be generated.

`auto` (default value) will automatically select the best quality for the given model.

`high`, `medium` and `low` are supported for the GPT image models.

`hd` and `standard` are supported for `dall-e-3`.

`standard` is the only option for `dall-e-2`.

response_format string or null Optional Defaults to url

The format in which generated images with `dall-e-2` and `dall-e-3` are returned. Must be one of `url` or `b64_json`. URLs are only valid for 60 minutes after the image has been generated. This parameter isn't supported for the GPT image models, which always return base64-encoded images.

size string or null Optional Defaults to auto

The size of the generated images. Must be one of `1024x1024`, `1536x1024` (landscape), `1024x1536` (portrait), or `auto` (default value) for the GPT image models, one of `256x256`, `512x512`, or `1024x1024` for `dall-e-2`, and one of `1024x1024`, `1792x1024`, or `1024x1792` for `dall-e-3`.

stream boolean or null Optional Defaults to false

Generate the image in streaming mode. Defaults to `false`. See the [Image generation guide](#) for more information. This parameter is only supported for the GPT image models.

style string or null Optional Defaults to vivid

The style of the generated images. This parameter is only supported for `dall-e-3`. Must be one of `vivid` or `natural`. Vivid causes the model to lean towards generating hyper-real and dramatic images. Natural causes the model to produce more natural, less hyper-real looking images.

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns an [image](#) object.

Generate image

Streaming

Example request

```
import base64
from openai import OpenAI
client = OpenAI()

img = client.images.generate()
```

```
model="gpt-image-1.5",
prompt="A cute baby sea otter",
n=1,
size="1024x1024"

)

image_bytes = base64.b64decode(img.data[0].b64_json)
with open("output.png", "wb") as f:
    f.write(image_bytes)
```

Response

```
{
  "created": 1713833628,
  "data": [
    {
      "b64_json": "..."
    }
  ],
  "usage": {
    "total_tokens": 100,
    "input_tokens": 50,
    "output_tokens": 50,
    "input_tokens_details": {
      "text_tokens": 10,
      "image_tokens": 40
    }
  }
}
```

Create image edit



POST <https://api.openai.com/v1/images/edits>

Creates an edited or extended image given one or more source images and a prompt. This endpoint only supports `gpt-image-1` and `dall-e-2`.

Request body

image string or array Required

The image(s) to edit. Must be a supported image file or an array of images.

For the GPT image models (`gpt-image-1`, `gpt-image-1-mini`, and `gpt-image-1.5`), each image should be a `png`, `webp`, or `jpg` file less than 50MB. You can provide up to 16 images.

For `dall-e-2`, you can only provide one image, and it should be a square `png` file less than 4MB.

prompt string Required

A text description of the desired image(s). The maximum length is 1000 characters for `dall-e-2`, and 32000 characters for the GPT image models.

background string or null Optional Defaults to auto

Allows to set transparency for the background of the generated image(s). This parameter is only supported for the GPT image models.

Must be one of `transparent`, `opaque` or `auto` (default value). When `auto` is used, the model will automatically determine the best background for the image.

If `transparent`, the output format needs to support transparency, so it should be set to either `png` (default value) or `webp`.

input_fidelity string Optional

Control how much effort the model will exert to match the style and features, especially facial features, of input images. This parameter is only supported for `gpt-image-1`. Unsupported for `gpt-image-1-mini`. Supports `high` and `low`. Defaults to `low`.

mask file Optional

An additional image whose fully transparent areas (e.g. where alpha is zero) indicate where `image` should be edited. If there are multiple images provided, the mask will be applied on the first image. Must be a valid PNG file, less than 4MB, and have the same dimensions as `image`.

model string Optional Defaults to dall-e-2

The model to use for image generation. Only `dall-e-2` and the GPT image models are supported. Defaults to `dall-e-2` unless a parameter specific to the GPT image models is used.

n integer or null Optional Defaults to 1

The number of images to generate. Must be between 1 and 10.

output_compression integer or null Optional Defaults to 100

The compression level (0-100%) for the generated images. This parameter is only supported for the GPT image models with the `webp` or `jpeg` output formats, and defaults to 100.

output_format string or null Optional Defaults to `png`

The format in which the generated images are returned. This parameter is only supported for the GPT image models. Must be one of `png`, `jpeg`, or `webp`. The default value is `png`.

partial_images integer Optional Defaults to 0

The number of partial images to generate. This parameter is used for streaming responses that return partial images. Value must be between 0 and 3. When set to 0, the response will be a single image sent in one streaming event.

Note that the final image may be sent before the full number of partial images are generated if the full image is generated more quickly.

quality string or null Optional Defaults to `auto`

The quality of the image that will be generated. `high`, `medium` and `low` are only supported for the GPT image models. `dall-e-2` only supports `standard` quality. Defaults to `auto`.

response_format string or null Optional Defaults to `url`

The format in which the generated images are returned. Must be one of `url` or `b64_json`. URLs are only valid for 60 minutes after the image has been generated. This parameter is only supported for `dall-e-2`, as the GPT image models always return base64-encoded images.

size string or null Optional Defaults to 1024x1024

The size of the generated images. Must be one of `1024x1024`, `1536x1024` (landscape), `1024x1536` (portrait), or `auto` (default value) for the GPT image models, and one of `256x256`, `512x512`, or `1024x1024` for `dall-e-2`.

stream boolean or null Optional Defaults to false

Edit the image in streaming mode. Defaults to `false`. See the [Image generation guide](#) for more information.

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns an [image](#) object.

[Edit image](#) [Streaming](#)

Example request

```
import base64
from openai import OpenAI
client = OpenAI()

prompt = """
Generate a photorealistic image of a gift basket on a white background
labeled 'Relax & Unwind' with a ribbon and handwriting-like font,
containing all the items in the reference pictures.
"""


```

```
result = client.images.edit(
    model="gpt-image-1.5",
```

```
image=[  
    open("body-lotion.png", "rb"),  
    open("bath-bomb.png", "rb"),  
    open("incense-kit.png", "rb"),  
    open("soap.png", "rb"),  
],  
prompt=prompt  
)  
  
image_base64 = result.data[0].b64_json  
image_bytes = base64.b64decode(image_base64)  
  
# Save the image to a file  
with open("gift-basket.png", "wb") as f:  
    f.write(image_bytes)
```

Create image variation



POST <https://api.openai.com/v1/images/variants>

Creates a variation of a given image. This endpoint only supports `dall-e-2`.

Request body

image file Required

The image to use as the basis for the variation(s). Must be a valid PNG file, less than 4MB, and square.

model string or "dall-e-2" Optional Defaults to dall-e-2

The model to use for image generation. Only `dall-e-2` is supported at this time.

n integer or null Optional Defaults to 1

The number of images to generate. Must be between 1 and 10.

response_format string or null Optional Defaults to url

The format in which the generated images are returned. Must be one of `url` or `b64_json`. URLs are only valid for 60 minutes after the image has been generated.

size string or null Optional Defaults to 1024x1024

The size of the generated images. Must be one of `256x256`, `512x512`, or `1024x1024`.

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns a list of image objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.images.create_variation(
    image=open("image_edit_original.png", "rb"),
    n=2,
    size="1024x1024"
)
```

Response

```
{
  "created": 1589478378,
  "data": [
    {
      "url": "https://..."
    },
    {
      "url": "https://..."
    }
  ]
}
```

The image generation response



The response from the image generation endpoint.

background string

The background parameter used for the image generation. Either `transparent` or `opaque`.

created integer

The Unix timestamp (in seconds) of when the image was created.

data array

The list of generated images.

› Show properties

output_format string

The output format of the image generation. Either `png`, `webp`, or `jpeg`.

quality string

The quality of the image generated. Either `low`, `medium`, or `high`.

size string

The size of the image generated. Either `1024x1024`, `1024x1536`, or `1536x1024`.

usage object

For `gpt-image-1` only, the token usage information for the image generation.

> Show properties

OBJECT The image generation response

```
{  
  "created": 1713833628,  
  "data": [  
    {  
      "b64_json": "..."  
    }  
  ],  
  "background": "transparent",  
  "output_format": "png",  
  "size": "1024x1024",  
  "quality": "high",  
  "usage": {  
    "total_tokens": 100,  
    "input_tokens": 50,  
    "output_tokens": 50,  
    "input_tokens_details": {  
      "text_tokens": 10,  
      "image_tokens": 40  
    }  
  }
```

{
}< PREVIOUS
Videos

NEXT

Images



Given a prompt and/or an input image, the model will generate a new image. Related guide:

[Image generation](#)

Create image



```
POST https://api.openai.com/v1/images/generations
```

Creates an image given a prompt. [Learn more.](#)

Request body

prompt string Required

A text description of the desired image(s). The maximum length is 32000 characters for the GPT image models, 1000 characters for `dall-e-2` and 4000 characters for `dall-e-3`.

background string or null Optional Defaults to auto

Allows to set transparency for the background of the generated image(s). This parameter is only supported for the GPT image models. Must be one of `transparent`, `opaque` or `auto` (default value). When `auto` is used, the model will automatically determine the best background for the image.

If `transparent`, the output format needs to support transparency, so it should be set to either `png` (default value) or `webp`.

model string Optional Defaults to dall-e-2

The model to use for image generation. One of `dall-e-2`, `dall-e-3`, or a GPT image model (`gpt-image-1`, `gpt-image-1-mini`, `gpt-image-1.5`). Defaults to `dall-e-2` unless a parameter specific to the GPT image models is used.

moderation string or null Optional Defaults to auto

Control the content-moderation level for images generated by the GPT image models. Must be either `low` for less restrictive filtering or `auto` (default value).

n integer or null Optional Defaults to 1

The number of images to generate. Must be between 1 and 10. For `dall-e-3`, only `n=1` is supported.

output_compression integer or null Optional Defaults to 100

The compression level (0-100%) for the generated images. This parameter is only supported for the GPT image models with the `webp` or `jpeg` output formats, and defaults to 100.

output_format string or null Optional Defaults to png

The format in which the generated images are returned. This parameter is only supported for the GPT image models. Must be one of `png`, `jpeg`, or `webp`.

partial_images integer Optional Defaults to 0

The number of partial images to generate. This parameter is used for streaming responses that return partial images. Value must be between 0 and 3. When set to 0, the response will be a single image sent in one streaming event.

Note that the final image may be sent before the full number of partial images are generated if the full image is generated more quickly.

quality string or null Optional Defaults to auto

The quality of the image that will be generated.

`auto` (default value) will automatically select the best quality for the given model.

`high`, `medium` and `low` are supported for the GPT image models.

`hd` and `standard` are supported for `dall-e-3`.

`standard` is the only option for `dall-e-2`.

response_format string or null Optional Defaults to url

The format in which generated images with `dall-e-2` and `dall-e-3` are returned. Must be one of `url` or `b64_json`. URLs are only valid for 60 minutes after the image has been generated. This parameter isn't supported for the GPT image models, which always return base64-encoded images.

size string or null Optional Defaults to auto

The size of the generated images. Must be one of `1024x1024`, `1536x1024` (landscape), `1024x1536` (portrait), or `auto` (default value) for the GPT image models, one of `256x256`, `512x512`, or `1024x1024` for `dall-e-2`, and one of `1024x1024`, `1792x1024`, or `1024x1792` for `dall-e-3`.

stream boolean or null Optional Defaults to false

Generate the image in streaming mode. Defaults to `false`. See the [Image generation guide](#) for more information. This parameter is only supported for the GPT image models.

style string or null Optional Defaults to vivid

The style of the generated images. This parameter is only supported for `dall-e-3`. Must be one of `vivid` or `natural`. Vivid causes the model to lean towards generating hyper-real and dramatic images. Natural causes the model to produce more natural, less hyper-real looking images.

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns an [image](#) object.

[Generate image](#)

[Streaming](#)

Example request

```
from openai import OpenAI

client = OpenAI()

stream = client.images.generate()
```

```
model="gpt-image-1.5",
prompt="A cute baby sea otter",
n=1,
size="1024x1024",
stream=True

)

for event in stream:
    print(event)
```

Response

```
event: image_generation.partial_image
data: {"type":"image_generation.partial_image","b64_json":"...","partial_image_index":0}

event: image_generation.completed
data: {"type":"image_generation.completed","b64_json":"...","usage":{"total_tokens":100,"input_toks":100,"output_toks":100}}
```

Create image edit



POST <https://api.openai.com/v1/images/edits>

Creates an edited or extended image given one or more source images and a prompt. This endpoint only supports `gpt-image-1` and `dall-e-2`.

Request body

image string or array Required

The image(s) to edit. Must be a supported image file or an array of images.

For the GPT image models (`gpt-image-1`, `gpt-image-1-mini`, and `gpt-image-1.5`), each image should be a `png`, `webp`, or `jpg` file less than 50MB. You can provide up to 16 images.

For `dall-e-2`, you can only provide one image, and it should be a square `png` file less than 4MB.

prompt string Required

A text description of the desired image(s). The maximum length is 1000 characters for `dall-e-2`, and 32000 characters for the GPT image models.

background string or null Optional Defaults to `auto`

Allows to set transparency for the background of the generated image(s). This parameter is only supported for the GPT image models.

Must be one of `transparent`, `opaque` or `auto` (default value). When `auto` is used, the model will automatically determine the best background for the image.

If `transparent`, the output format needs to support transparency, so it should be set to either `png` (default value) or `webp`.

input_fidelity string Optional

Control how much effort the model will exert to match the style and features, especially facial features, of input images. This parameter is only supported for `gpt-image-1`. Unsupported for `gpt-image-1-mini`. Supports `high` and `low`. Defaults to `low`.

mask file Optional

An additional image whose fully transparent areas (e.g. where alpha is zero) indicate where `image` should be edited. If there are multiple images provided, the mask will be applied on the first image. Must be a valid PNG file, less than 4MB, and have the same dimensions as `image`.

model string Optional Defaults to dall-e-2

The model to use for image generation. Only `dall-e-2` and the GPT image models are supported. Defaults to `dall-e-2` unless a parameter specific to the GPT image models is used.

n integer or null Optional Defaults to 1

The number of images to generate. Must be between 1 and 10.

output_compression integer or null Optional Defaults to 100

The compression level (0-100%) for the generated images. This parameter is only supported for the GPT image models with the `webp` or `jpeg` output formats, and defaults to 100.

output_format string or null Optional Defaults to png

The format in which the generated images are returned. This parameter is only supported for the GPT image models. Must be one of `png`, `jpeg`, or `webp`. The default value is `png`.

partial_images integer Optional Defaults to 0

The number of partial images to generate. This parameter is used for streaming responses that return partial images. Value must be between 0 and 3. When set to 0, the response will be a single image sent in one streaming event.

Note that the final image may be sent before the full number of partial images are generated if the full image is generated more quickly.

quality string or null Optional Defaults to auto

The quality of the image that will be generated. `high`, `medium` and `low` are only supported for the GPT image models.

`dall-e-2` only supports `standard` quality. Defaults to `auto`.

response_format string or null Optional Defaults to url

The format in which the generated images are returned. Must be one of `url` or `b64_json`. URLs are only valid for 60 minutes after the image has been generated. This parameter is only supported for `dall-e-2`, as the GPT image models always return base64-encoded images.

size string or null Optional Defaults to 1024x1024

The size of the generated images. Must be one of `1024x1024`, `1536x1024` (landscape), `1024x1536` (portrait), or `auto` (default value) for the GPT image models, and one of `256x256`, `512x512`, or `1024x1024` for `dall-e-2`.

stream boolean or null Optional Defaults to false

Edit the image in streaming mode. Defaults to `false`. See the [Image generation guide](#) for more information.

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns an [image](#) object.

[Edit image](#)

[Streaming](#)

Example request

```
import base64
from openai import OpenAI
client = OpenAI()

prompt = """
Generate a photorealistic image of a gift basket on a white background
labeled 'Relax & Unwind' with a ribbon and handwriting-like font,
containing all the items in the reference pictures.
"""

result = client.images.edit(
    model="gpt-image-1.5",
    image=[
        open("body-lotion.png", "rb"),
        open("bath-bomb.png", "rb"),
        open("incense-kit.png", "rb"),
        open("soap.png", "rb"),
    ],
    prompt=prompt
)
```

```
image_base64 = result.data[0].b64_json
image_bytes = base64.b64decode(image_base64)

# Save the image to a file
with open("gift-basket.png", "wb") as f:
    f.write(image_bytes)
```

Create image variation



POST <https://api.openai.com/v1/images/variants>

Creates a variation of a given image. This endpoint only supports `dall-e-2`.

Request body

`image` file Required

The image to use as the basis for the variation(s). Must be a valid PNG file, less than 4MB, and square.

`model` string or "dall-e-2" Optional Defaults to dall-e-2

The model to use for image generation. Only `dall-e-2` is supported at this time.

n integer or null Optional Defaults to 1

The number of images to generate. Must be between 1 and 10.

response_format string or null Optional Defaults to url

The format in which the generated images are returned. Must be one of `url` or `b64_json`. URLs are only valid for 60 minutes after the image has been generated.

size string or null Optional Defaults to 1024x1024

The size of the generated images. Must be one of `256x256`, `512x512`, or `1024x1024`.

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns a list of `image` objects.

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.images.create_variation(
    image=open("image_edit_original.png", "rb"),
```

```
n=2,  
size="1024x1024"  
)
```

Response

```
{  
  "created": 1589478378,  
  "data": [  
    {  
      "url": "https://..."  
    },  
    {  
      "url": "https://..."  
    }  
  ]  
}
```

The image generation response



The response from the image generation endpoint.

background string

The background parameter used for the image generation. Either `transparent` or `opaque`.

created integer

The Unix timestamp (in seconds) of when the image was created.

data array

The list of generated images.

› Show properties

output_format string

The output format of the image generation. Either `png`, `webp`, or `jpeg`.

quality string

The quality of the image generated. Either `low`, `medium`, or `high`.

size string

The size of the image generated. Either `1024x1024`, `1024x1536`, or `1536x1024`.

usage object

For `gpt-image-1` only, the token usage information for the image generation.

› Show properties

OBJECT The image generation response

```
{
```

```
  "created": 1713833628,
```

```
"data": [  
  {  
    "b64_json": "..."  
  },  
,  
  "background": "transparent",  
  "output_format": "png",  
  "size": "1024x1024",  
  "quality": "high",  
  "usage": {  
    "total_tokens": 100,  
    "input_tokens": 50,  
    "output_tokens": 50,  
    "input_tokens_details": {  
      "text_tokens": 10,  
      "image_tokens": 40  
    }  
  }  
}
```

< PREVIOUS
Videos

NEXT

Image Streaming

🔗

Stream image generation and editing in real time with server-sent events.

[Learn more about image streaming.](#)

🔗

image_generation.partial_image

🔗

Emitted when a partial image is available during image generation streaming.

b64_json string

Base64-encoded partial image data, suitable for rendering as an image.

background string

The background setting for the requested image.

created_at integer

The Unix timestamp when the event was created.

output_format string

The output format for the requested image.

partial_image_index integer

0-based index for the partial image (streaming).

quality string

The quality setting for the requested image.

size string

The size of the requested image.

type string

The type of the event. Always `image_generation.partial_image`.

OBJECT `image_generation.partial_image`

```
{  
  "type": "image_generation.partial_image",  
  "b64_json": "...",  
  "created_at": 1620000000,  
  "size": "1024x1024",  
  "quality": "high",
```

```
"background": "transparent",
"output_format": "png",
"partial_image_index": 0
}
```

image_generation.completed



Emitted when image generation has completed and the final image is available.

b64_json string

Base64-encoded image data, suitable for rendering as an image.

background string

The background setting for the generated image.

created_at integer

The Unix timestamp when the event was created.

output_format string

The output format for the generated image.

quality string

The quality setting for the generated image.

size string

The size of the generated image.

type string

The type of the event. Always `image_generation.completed`.

usage object

For the GPT image models only, the token usage information for the image generation.

› Show properties

OBJECT `image_generation.completed`

```
{  
  "type": "image_generation.completed",  
  "b64_json": "...",  
  "created_at": 1620000000,  
  "size": "1024x1024",  
  "quality": "high",  
  "background": "transparent",  
  "output_format": "png",  
  "usage": {  
    "total_tokens": 100,  
    "input_tokens": 50,  
    "output_tokens": 50,  
  }  
}
```

```
"input_tokens_details": {  
    "text_tokens": 10,  
    "image_tokens": 40  
}  
}  
}
```



image_edit.partial_image



Emitted when a partial image is available during image editing streaming.

b64_json string

Base64-encoded partial image data, suitable for rendering as an image.

background string

The background setting for the requested edited image.

created_at integer

The Unix timestamp when the event was created.

output_format string

The output format for the requested edited image.

partial_image_index integer

0-based index for the partial image (streaming).

quality string

The quality setting for the requested edited image.

size string

The size of the requested edited image.

type string

The type of the event. Always `image_edit.partial_image`.

OBJECT `image_edit.partial_image`

```
{  
  "type": "image_edit.partial_image",  
  "b64_json": "...",  
  "created_at": 1620000000,  
  "size": "1024x1024",  
  "quality": "high",  
  "background": "transparent",
```

```
"output_format": "png",
"partial_image_index": 0
}
```

image_edit.completed



Emitted when image editing has completed and the final image is available.

b64_json string

Base64-encoded final edited image data, suitable for rendering as an image.

background string

The background setting for the edited image.

created_at integer

The Unix timestamp when the event was created.

output_format string

The output format for the edited image.

quality string

The quality setting for the edited image.

size string

The size of the edited image.

type string

The type of the event. Always `image_edit.completed`.

usage object

For the GPT image models only, the token usage information for the image generation.

› Show properties

OBJECT `image_edit.completed`

```
{  
  "type": "image_edit.completed",  
  "b64_json": "...",  
  "created_at": 1620000000,  
  "size": "1024x1024",  
  "quality": "high",  
  "background": "transparent",  
  "output_format": "png",  
  "usage": {  
    "total_tokens": 100,  
    "input_tokens": 50,  
    "output_tokens": 50,  
    "input_tokens_details": {  
      "text_tokens": 10,  
      "image_tokens": 40  
    }  
  }  
}
```

```
    "image_tokens": 40
  }
}
}
```

< PREVIOUS
Images

NEXT

Embeddings



Get a vector representation of a given input that can be easily consumed by machine learning models and algorithms. Related guide: [Embeddings](#)

Create embeddings



```
POST https://api.openai.com/v1/embeddings
```

Creates an embedding vector representing the input text.

Request body

input string or array Required

Input text to embed, encoded as a string or array of tokens. To embed multiple inputs in a single request, pass an array of strings or array of token arrays. The input must not exceed the max input tokens for the model (8192 tokens for all embedding models), cannot be an empty string, and any array must be 2048 dimensions or less. [Example Python code](#) for counting tokens. In addition to the per-input token limit, all embedding models enforce a maximum of 300,000 tokens summed across all inputs in a single request.

model string Required

ID of the model to use. You can use the [List models](#) API to see all of your available models, or see our [Model overview](#) for descriptions of them.

dimensions integer Optional

The number of dimensions the resulting output embeddings should have. Only supported in `text-embedding-3` and later models.

encoding_format string Optional Defaults to float

The format to return the embeddings in. Can be either `float` or [`base64`](#).

user string Optional

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

A list of [embedding](#) objects.

Example request

```
from openai import OpenAI
client = OpenAI()

client.embeddings.create(
    model="text-embedding-ada-002",
```

```
    input="The food was delicious and the waiter...",  
    encoding_format="float"  
)
```

Response

```
{  
  "object": "list",  
  "data": [  
    {  
      "object": "embedding",  
      "embedding": [  
        0.0023064255,  
        -0.009327292,  
        .... (1536 floats total for ada-002)  
        -0.0028842222,  
      ],  
      "index": 0  
    }  
  ],  
  "model": "text-embedding-ada-002",  
  "usage": {  
    "prompt_tokens": 8,  
    "total_tokens": 8  
  }  
}
```

The embedding object



Represents an embedding vector returned by embedding endpoint.

embedding array

The embedding vector, which is a list of floats. The length of vector depends on the model as listed in the [embedding guide](#).

index integer

The index of the embedding in the list of embeddings.

object string

The object type, which is always "embedding".

OBJECT The embedding object

```
{  
  "object": "embedding",  
  "embedding": [  
    0.0023064255,  
    -0.009327292,  
    .... (1536 floats total for ada-002)  
    -0.0028842222,  
  ],  
  "index": 0  
}
```

< PREVIOUS
Image Streaming

NEXT

Batch



Create large batches of API requests for asynchronous processing. The Batch API returns completions within 24 hours for a 50% discount. Related guide: [Batch](#)

Create batch



```
POST https://api.openai.com/v1/batches
```

Creates and executes a batch from an uploaded file of requests

Request body

completion_window string Required

The time frame within which the batch should be processed. Currently only `24h` is supported.

endpoint string Required

The endpoint to be used for all requests in the batch. Currently `/v1/responses`, `/v1/chat/completions`, `/v1/embeddings`, `/v1/completions`, and `/v1/moderations` are supported. Note that `/v1/embeddings` batches are also restricted to a maximum of 50,000 embedding inputs across all requests in the batch.

input_file_id string Required

The ID of an uploaded file that contains requests for the new batch.

See [upload file](#) for how to upload a file.

Your input file must be formatted as a [JSONL file](#), and must be uploaded with the purpose `batch`. The file can contain up to 50,000 requests, and can be up to 200 MB in size.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

output.expires_after object Optional

The expiration policy for the output and/or error file that are generated for a batch.

› Show properties

Returns

The created [Batch](#) object.

Example request

```
from openai import OpenAI
client = OpenAI()

client.batches.create(
    input_file_id="file-abc123",
    endpoint="/v1/chat/completions",
    completion_window="24h"
)
```

Response

```
{
  "id": "batch_abc123",
  "object": "batch",
  "endpoint": "/v1/chat/completions",
  "errors": null,
  "input_file_id": "file-abc123",
  "completion_window": "24h",
  "status": "validating",
  "output_file_id": null,
  "error_file_id": null,
  "created_at": 1711471533,
  "in_progress_at": null,
  "expires_at": null,
  "finalizing_at": null,
  "completed_at": null,
  "failed_at": null,
```

```
"expired_at": null,  
"cancelling_at": null,  
"cancelled_at": null,  
"request_counts": {  
    "total": 0,  
    "completed": 0,  
    "failed": 0  
},  
"metadata": {  
    "customer_id": "user_123456789",  
    "batch_description": "Nightly eval job",  
}  
}
```

Retrieve batch



```
GET https://api.openai.com/v1/batches/{batch_id}
```

Retrieves a batch.

Path parameters

batch_id string Required

The ID of the batch to retrieve.

Returns

The [Batch](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

client.batches.retrieve("batch_abc123")
```

Response

```
{
  "id": "batch_abc123",
  "object": "batch",
  "endpoint": "/v1/completions",
  "errors": null,
  "input_file_id": "file-abc123",
  "completion_window": "24h",
  "status": "completed",
  "output_file_id": "file-cvaTdG",
```

```
"error_file_id": "file-H0WS94",
"created_at": 1711471533,
"in_progress_at": 1711471538,
"expires_at": 1711557933,
"finalizing_at": 1711493133,
"completed_at": 1711493163,
"failed_at": null,
"expired_at": null,
"canceling_at": null,
"cancelled_at": null,
"request_counts": {
    "total": 100,
    "completed": 95,
    "failed": 5
},
"metadata": {
    "customer_id": "user_123456789",
    "batch_description": "Nightly eval job",
}
}
```

Cancel batch



```
POST https://api.openai.com/v1/batches/{batch_id}/cancel
```

Cancels an in-progress batch. The batch will be in status `cancelling` for up to 10 minutes, before changing to `cancelled`, where it will have partial results (if any) available in the output file.

Path parameters

batch_id string Required

The ID of the batch to cancel.

Returns

The [Batch](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

client.batches.cancel("batch_abc123")
```

Response

```
{  
  "id": "batch_abc123",  
  "object": "batch",  
  "endpoint": "/v1/chat/completions",  
  "errors": null,  
  "input_file_id": "file-abc123",  
  "completion_window": "24h",  
  "status": "cancelling",  
  "output_file_id": null,  
  "error_file_id": null,  
  "created_at": 1711471533,  
  "in_progress_at": 1711471538,  
  "expires_at": 1711557933,  
  "finalizing_at": null,  
  "completed_at": null,  
  "failed_at": null,  
  "expired_at": null,  
  "cancelling_at": 1711475133,  
  "cancelled_at": null,  
  "request_counts": {  
    "total": 100,  
    "completed": 23,  
    "failed": 1  
  },  
  "metadata": {  
    "customer_id": "user_123456789",  
    "batch_description": "Nightly eval job",  
  }  
}
```

List batch



```
GET https://api.openai.com/v1/batches
```

List your organization's batches.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include after=obj_foo in order to fetch the next page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

Returns

A list of paginated Batch objects.

Example request

```
from openai import OpenAI
client = OpenAI()

client.batches.list()
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "batch_abc123",
      "object": "batch",
      "endpoint": "/v1/chat/completions",
      "errors": null,
      "input_file_id": "file-abc123",
      "completion_window": "24h",
      "status": "completed",
      "output_file_id": "file-cvaTdG",
      "error_file_id": "file-HOWS94",
      "created_at": 1711471533,
      "in_progress_at": 1711471538,
      "expires_at": 1711557933,
```

```
"finalizing_at": 1711493133,  
"completed_at": 1711493163,  
"failed_at": null,  
"expired_at": null,  
"cancelling_at": null,  
"cancelled_at": null,  
"request_counts": {  
    "total": 100,  
    "completed": 95,  
    "failed": 5  
},  
"metadata": {  
    "customer_id": "user_123456789",  
    "batch_description": "Nightly job",  
}  
},  
{ ... },  
],  
"first_id": "batch_abc123",  
"last_id": "batch_abc456",  
"has_more": true  
}
```

The batch object



cancelled_at integer

The Unix timestamp (in seconds) for when the batch was cancelled.

cancelling_at integer

The Unix timestamp (in seconds) for when the batch started cancelling.

completed_at integer

The Unix timestamp (in seconds) for when the batch was completed.

completion_window string

The time frame within which the batch should be processed.

created_at integer

The Unix timestamp (in seconds) for when the batch was created.

endpoint string

The OpenAI API endpoint used by the batch.

error_file_id string

The ID of the file containing the outputs of requests with errors.

errors object

› Show properties

expired_at integer

The Unix timestamp (in seconds) for when the batch expired.

expires_at integer

The Unix timestamp (in seconds) for when the batch will expire.

failed_at integer

The Unix timestamp (in seconds) for when the batch failed.

finalizing_at integer

The Unix timestamp (in seconds) for when the batch started finalizing.

id string**in_progress_at** integer

The Unix timestamp (in seconds) for when the batch started processing.

input_file_id string

The ID of the input file for the batch.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

Model ID used to process the batch, like `gpt-5-2025-08-07`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

object string

The object type, which is always `batch`.

output_file_id string

The ID of the file containing the outputs of successfully executed requests.

request_counts object

The request counts for different statuses within the batch.

› Show properties

status string

The current status of the batch.

usage object

Represents token usage details including input tokens, output tokens, a breakdown of output tokens, and the total tokens used. Only populated on batches created after September 7, 2025.

> Show properties

OBJECT The batch object

```
{  
  "id": "batch_abc123",  
  "object": "batch",  
  "endpoint": "/v1/completions",  
  "model": "gpt-5-2025-08-07",  
  "errors": null,  
  "input_file_id": "file-abc123",  
  "completion_window": "24h",  
  "status": "completed",  
  "output_file_id": "file-cvaTdG",  
  "error_file_id": "file-HOWS94",  
  "created_at": 1711471533,  
  "in_progress_at": 1711471538,  
  "expires_at": 1711557933,  
  "finalizing_at": 1711493133,  
  "completed_at": 1711493163,  
  "failed_at": null,  
  "expired_at": null,  
  "cancelling_at": null,  
  "cancelled_at": null,  
  "request_counts": {  
    "total": 100,  
    "completed": 95,  
    "failed": 5  
  },  
  "usage": {
```

```
"input_tokens": 1500,  
"input_tokens_details": {  
    "cached_tokens": 1024  
},  
"output_tokens": 500,  
"output_tokens_details": {  
    "reasoning_tokens": 300  
},  
"total_tokens": 2000  
},  
"metadata": {  
    "customer_id": "user_123456789",  
    "batch_description": "Nightly eval job",  
}  
}
```

The request input object

🔗

The per-line object of the batch input file

custom_id string

A developer-provided per-request id that will be used to match outputs to inputs. Must be unique for each request in a batch.

method string

The HTTP method to be used for the request. Currently only `POST` is supported.

url string

The OpenAI API relative URL to be used for the request. Currently `/v1/responses`, `/v1/chat/completions`, `/v1/embeddings`, `/v1/completions`, and `/v1/moderations` are supported.

OBJECT The request input object

```
{"custom_id": "request-1", "method": "POST", "url": "/v1/chat/completions", "body": {"model": "gpt-3.5-turbo", "messages": [{"role": "user", "content": "Hello, how are you?"}], "temperature": 0.5}, "log_file": "output.log", "error_file": "error.log", "max_tokens": 100, "n": 1, "presence_penalty": 0, "stop": null, "top_p": 1, "frequency_penalty": 0}
```

The request output object



The per-line object of the batch output and error files

custom_id string

A developer-provided per-request id that will be used to match outputs to inputs.

error object

For requests that failed with a non-HTTP error, this will contain more information on the cause of the failure.

> Show properties

id string

response object

> Show properties

OBJECT The request output object

```
{"id": "batch_req_wnaDys", "custom_id": "request-2", "response": {"status_code": 200, "request_id": "req_wnaDys", "error": null}}
```

< PREVIOUS
Graders

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Uploads

Allows you to upload large files in multiple parts.

Create upload

```
POST https://api.openai.com/v1/uploads
```

Creates an intermediate [Upload](#) object that you can add [Parts](#) to. Currently, an Upload can accept at most 8 GB in total and expires after an hour after you create it.

Once you complete the Upload, we will create a [File](#) object that contains all the parts you uploaded. This File is usable in the rest of our platform as a regular File object.

For certain `purpose` values, the correct `mime_type` must be specified. Please refer to documentation for the [supported MIME types for your use case](#).

For guidance on the proper filename extensions for each purpose, please follow the documentation on [creating a File](#).

Request body

bytes integer Required

The number of bytes in the file you are uploading.

filename string Required

The name of the file to upload.

mime_type string Required

The MIME type of the file.

This must fall within the supported MIME types for your file purpose. See the supported MIME types for assistants and vision.

purpose string Required

The intended purpose of the uploaded file.

See the [documentation on File purposes](#).

expires_after object Optional

The expiration policy for a file. By default, files with `purpose=batch` expire after 30 days and all other files are persisted until they are manually deleted.

› Show properties

Returns

The Upload object with status `pending`.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/uploads \
2   -H "Authorization: Bearer $OPENAI_API_KEY" \
3   -d '{
4     "purpose": "fine-tune",
5     "filename": "training_examples.jsonl",
6     "bytes": 2147483648,
7     "mime_type": "text/jsonl",
8     "expires_after": {
9       "anchor": "created_at",
10      "seconds": 3600
11    }
12  }'
```

Response

🔗

```
1  {
2    "id": "upload_abc123",
3    "object": "upload",
4    "bytes": 2147483648,
5    "created_at": 1719184911,
6    "filename": "training_examples.jsonl",
7    "purpose": "fine-tune",
8    "status": "pending",
9    "expires_at": 1719127296
10 }
```

Add upload part

```
POST https://api.openai.com/v1/uploads/{upload_id}/parts
```

Adds a Part to an Upload object. A Part represents a chunk of bytes from the file you are trying to upload.

Each Part can be at most 64 MB, and you can add Parts until you hit the Upload maximum of 8 GB.

It is possible to add multiple Parts in parallel. You can decide the intended order of the Parts when you complete the Upload.

Path parameters

upload_id string Required

The ID of the Upload.

Request body

data file Required

The chunk of bytes for this Part.

Returns

The upload Part object.

Example request

curl ▾

```
1 curl https://api.openai.com/v1/uploads/upload_abc123/parts
2   -F data="aHR0cHM6Ly9hcGkub3B1bmFpLmNvbS92MS91cGxvYWRz..."
```

Response



```
1 {
2   "id": "part_def456",
3   "object": "upload.part",
4   "created_at": 1719185911,
5   "upload_id": "upload_abc123"
6 }
```

Complete upload

```
POST https://api.openai.com/v1/uploads/{upload_id}/complete
```

Completes the [Upload](#).

Within the returned Upload object, there is a nested [File](#) object that is ready to use in the rest of the platform.

You can specify the order of the Parts by passing in an ordered list of the Part IDs.

The number of bytes uploaded upon completion must match the number of bytes initially specified when creating the Upload object. No Parts may be added after an Upload is completed.

Path parameters

upload_id string Required

The ID of the Upload.

Request body

part_ids array Required

The ordered list of Part IDs.

md5 string Optional

The optional md5 checksum for the file contents to verify if the bytes uploaded matches what you expect.

Returns

The [Upload](#) object with status `completed` with an additional `file` property containing the created usable File object.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/uploads/upload_abc123/complete
2   -d '{
3     "part_ids": ["part_def456", "part_ghi789"]
4   }'
```

Response

🔗

```
1 {
2   "id": "upload_abc123",
3   "object": "upload",
4   "bytes": 2147483648,
5   "created_at": 1719184911,
6   "filename": "training_examples.jsonl",
7   "purpose": "fine-tune",
8   "status": "completed",
9   "expires_at": 1719127296,
10  "file": {
11    "id": "file-xyz321",
12    "object": "file",
13    "bytes": 2147483648,
14    "created_at": 1719186911,
15    "expires_at": 1719127296,
16    "filename": "training_examples.jsonl",
17    "purpose": "fine-tune",
18
19}
```

```
}
```

```
}
```

Cancel upload

```
POST https://api.openai.com/v1/uploads/{upload_id}/cancel
```

Cancels the Upload. No Parts may be added after an Upload is cancelled.

Path parameters

upload_id string Required

The ID of the Upload.

Returns

The [Upload](#) object with status `cancelled`.

Example request

curl ⚡

```
curl https://api.openai.com/v1/uploads/upload_abc123/cancel
```

Response



```
1  {
2    "id": "upload_abc123",
3    "object": "upload",
4    "bytes": 2147483648,
5    "created_at": 1719184911,
6    "filename": "training_examples.jsonl",
7    "purpose": "fine-tune",
8    "status": "cancelled",
9    "expires_at": 1719127296
10 }
```

The upload object

The Upload object can accept byte chunks in the form of Parts.

bytes integer

The intended number of bytes to be uploaded.

created_at integer

The Unix timestamp (in seconds) for when the Upload was created.

expires_at integer

The Unix timestamp (in seconds) for when the Upload will expire.

file undefined or null

The ready File object after the Upload is completed.

filename string

The name of the file to be uploaded.

id string

The Upload unique identifier, which can be referenced in API endpoints.

object string

The object type, which is always "upload".

purpose string

The intended purpose of the file. [Please refer here](#) for acceptable values.

status string

The status of the Upload.

OBJECT The upload object



```
1  {
2      "id": "upload_abc123",
3      "object": "upload",
4      "bytes": 2147483648,
5      "created_at": 1719184911,
6      "filename": "training_examples.jsonl",
7      "purpose": "fine-tune",
8      "status": "completed",
9      "expires_at": 1719127296,
10     "file": {
11         "id": "file-xyz321",
12         "object": "file",
13         "bytes": 2147483648,
14         "created_at": 1719186911,
15         "filename": "training_examples.jsonl",
16         "purpose": "fine-tune",
17     }
18 }
```

The upload part object

The upload Part represents a chunk of bytes we can add to an Upload object.

created_at integer

The Unix timestamp (in seconds) for when the Part was created.

id string

The upload Part unique identifier, which can be referenced in API endpoints.

object string

The object type, which is always `upload.part`.

upload_id string

The ID of the Upload object that this Part was added to.

OBJECT The upload part object



```
1 {
2   "id": "part_def456",
3   "object": "upload.part",
4   "created_at": 1719186911,
5   "upload_id": "upload_abc123"
6 }
```

< PREVIOUS
Files

NEXT >
Models

Models



List and describe the various models available in the API. You can refer to the [Models](#) documentation to understand what models are available and the differences between them.

List models



```
GET https://api.openai.com/v1/models
```

Lists the currently available models, and provides basic information about each one such as the owner and availability.

Returns

A list of [model](#) objects.

Example request

```
from openai import OpenAI
client = OpenAI()

client.models.list()
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "model-id-0",
      "object": "model",
      "created": 1686935002,
      "owned_by": "organization-owner"
    },
    {
      "id": "model-id-1",
      "object": "model",
      "created": 1686935002,
      "owned_by": "organization-owner",
    },
    {
      "id": "model-id-2",
      "object": "model",
      "created": 1686935002,
      "owned_by": "openai"
    },
  ]
}
```

```
],  
  "object": "list"  
}
```

Retrieve model



```
GET https://api.openai.com/v1/models/{model}
```

Retrieves a model instance, providing basic information about the model such as the owner and permissioning.

Path parameters

model string Required

The ID of the model to use for this request

Returns

The model object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

client.models.retrieve("gpt-5.2")
```

Response

```
{
  "id": "gpt-5.2",
  "object": "model",
  "created": 1686935002,
  "owned_by": "openai"
}
```

Delete a fine-tuned model



```
DELETE https://api.openai.com/v1/models/{model}
```

Delete a fine-tuned model. You must have the Owner role in your organization to delete a model.

Path parameters

model string Required

The model to delete

Returns

Deletion status.

Example request

```
from openai import OpenAI
client = OpenAI()

client.models.delete("ft:gpt-4o-mini:acemeco:suffix:abc123")
```

Response

```
{
  "id": "ft:gpt-4o-mini:acemeco:suffix:abc123",
  "object": "model",
  "deleted": true
}
```

The model object



Describes an OpenAI model offering that can be used with the API.

created integer

The Unix timestamp (in seconds) when the model was created.

id string

The model identifier, which can be referenced in the API endpoints.

object string

The object type, which is always "model".

owned_by string

The organization that owns the model.

OBJECT The model object

```
{  
  "id": "gpt-5.2",  
  "object": "model",  
  "created": 1686935002,  
  "owned_by": "openai"  
}
```

< PREVIOUS
Uploads

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Vector stores

Vector stores power semantic search for the Retrieval API and the `file_search` tool in the Responses and Assistants APIs.

Related guide: [File Search](#)

Create vector store

```
POST https://api.openai.com/v1/vector_stores
```

Create a vector store.

Request body

chunking_strategy object Optional

The chunking strategy used to chunk the file(s). If not set, will use the `auto` strategy. Only applicable if `file_ids` is non-empty.

› Show possible types

description string Optional

A description for the vector store. Can be used to describe the vector store's purpose.

expires_after object Optional

The expiration policy for a vector store.

› Show properties

file_ids array Optional

A list of File IDs that the vector store should use. Useful for tools like `file_search` that can access files.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

name string Optional

The name of the vector store.

Returns

A [vector store](#) object.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 vector_store = client.vector_stores.create(
5     name="Support FAQ"
6 )
7 print(vector_store)
```

Response

🔗

```
1 {
2     "id": "vs_abc123",
3     "object": "vector_store",
4     "created_at": 1699061776,
5     "name": "Support FAQ",
6     "description": "Contains commonly asked questions and answers, organized by topic.",
7     "bytes": 139920,
```

```
8     "file_counts": {  
9         "in_progress": 0,  
10        "completed": 3,  
11        "failed": 0,  
12        "cancelled": 0,  
13        "total": 3  
14    }  
15 }
```

List vector stores

GET https://api.openai.com/v1/vector_stores

Returns a list of vector stores.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of vector store objects.

Example request

python ▾



```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3  
4 vector_stores = client.vector_stores.list()  
5 print(vector_stores)
```

Response



```
1 {  
2   "object": "list",  
3   "data": [  
4     {  
5       "id": "vs_abc123",  
6       "object": "vector_store",  
7       "created_at": 1699061776,  
8       "name": "Support FAQ",  
9       "description": "Contains commonly asked questions and answers, organized by topic.",  
10      "bytes": 139920,  
11      "file_counts": {  
12        "in_progress": 0,  
13        "completed": 3,  
14        "failed": 0,  
15        "cancelled": 0,  
16        "total": 3  
17      }  
18    },  
19    {  
20      "id": "vs_abc456",  
21      "object": "vector_store",  
22      "created_at": 1699061776,
```

```
23     "name": "Support FAQ v2",
24     "description": null,
25     "bytes": 139920,
26     "file_counts": {
27       "in_progress": 0,
28       "completed": 3,
29       "failed": 0,
30       "cancelled": 0,
31       "total": 3
32     }
33   }
34 ],
35 "first_id": "vs_abc123",
36 "last_id": "vs_abc456",
37 "has_more": false
38 }
```

Retrieve vector store

```
GET https://api.openai.com/v1/vector_stores/{vector_store_id}
```

Retrieves a vector store.

Path parameters

vector_store_id string Required

The ID of the vector store to retrieve.

Returns

The vector store object matching the specified ID.

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 vector_store = client.vector_stores.retrieve(
5     vector_store_id="vs_abc123"
6 )
7 print(vector_store)
```

Response

Copy

```
1 {
2     "id": "vs_abc123",
3     "object": "vector_store",
```

```
4   "created_at": 1699061776
5 }
```

Modify vector store

POST https://api.openai.com/v1/vector_stores/{vector_store_id}

Modifies a vector store.

Path parameters

vector_store_id string Required

The ID of the vector store to modify.

Request body

expires_after object or null Optional

The expiration policy for a vector store.

› Show properties

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

name string or null Optional

The name of the vector store.

Returns

The modified vector store object.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 vector_store = client.vector_stores.update(
5     vector_store_id="vs_abc123",
6     name="Support FAQ"
7 )
8 print(vector_store)
```

Response



```
1  {
2    "id": "vs_abc123",
3    "object": "vector_store",
4    "created_at": 1699061776,
5    "name": "Support FAQ",
6    "description": "Contains commonly asked questions and answers, organized by topic.",
7    "bytes": 139920,
8    "file_counts": {
9      "in_progress": 0,
10     "completed": 3,
11     "failed": 0,
12     "cancelled": 0,
13     "total": 3
14   }
15 }
```

Delete vector store

```
DELETE https://api.openai.com/v1/vector_stores/{vector_store_id}
```

Delete a vector store.

Path parameters

vector_store_id string Required

The ID of the vector store to delete.

Returns

Deletion status

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 deleted_vector_store = client.vector_stores.delete(
5     vector_store_id="vs_abc123"
6 )
7 print(deleted_vector_store)
```

Response

```
1 {
2   id: "vs_abc123",
3   object: "vector_store.deleted",
4   deleted: true
5 }
```

Search vector store

POST https://api.openai.com/v1/vector_stores/{vector_store_id}/search

Search a vector store for relevant chunks based on a query and file attributes filter.

Path parameters

vector_store_id string Required

The ID of the vector store to search.

Request body

query string or array Required

A query string for a search

filters object Optional

A filter to apply based on file attributes.

› Show possible types

max_num_results integer Optional Defaults to 10

The maximum number of results to return. This number should be between 1 and 50 inclusive.

ranking_options object Optional

Ranking options for search.

› Show properties

rewrite_query boolean Optional Defaults to false

Whether to rewrite the natural language query for vector search.

Returns

A page of search results from the vector store.

Example request

curl ▾ 

```
1 curl -X POST \
2 https://api.openai.com/v1/vector_stores/vs_abc123/search \
3 -H "Authorization: Bearer $OPENAI_API_KEY" \
4 -H "Content-Type: application/json" \
5 -d '{"query": "What is the return policy?", "filters": {...}}'
```

Response



```
1 {
2   "object": "vector_store.search_results.page",
3   "search_query": "What is the return policy?",
4   "data": [
5     {
6       "file_id": "file_123",
7       "filename": "document.pdf",
8       "score": 0.95,
9       "attributes": {
10         "author": "John Doe",
11         "date": "2023-01-01"
12       },
13       "content": [
14         {
15           "type": "text",
16           "text": "Relevant chunk"
17         }
18       ]
19     },
20   {
```

```
21     "file_id": "file_456",
22     "filename": "notes.txt",
23     "score": 0.89,
24     "attributes": {
25         "author": "Jane Smith",
26         "date": "2023-01-02"
27     },
28     "content": [
29         {
30             "type": "text",
31             "text": "Sample text content from the vector store."
32         }
33     ]
34 },
35 ],
36 "has_more": false,
37 "next_page": null
38 }
```

The vector store object

A vector store is a collection of processed files can be used by the `file_search` tool.

created_at integer

The Unix timestamp (in seconds) for when the vector store was created.

expires_after object

The expiration policy for a vector store.

› Show properties

expires_at integer

The Unix timestamp (in seconds) for when the vector store will expire.

file_counts object

› Show properties

id string

The identifier, which can be referenced in API endpoints.

last_active_at integer

The Unix timestamp (in seconds) for when the vector store was last active.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

name string

The name of the vector store.

object string

The object type, which is always `vector_store`.

status string

The status of the vector store, which can be either `expired`, `in_progress`, or `completed`. A status of `completed` indicates that the vector store is ready for use.

usage_bytes integer

The total number of bytes used by the files in the vector store.

OBJECT The vector store object



```
1  {
2    "id": "vs_123",
3    "object": "vector_store",
4    "created_at": 1698107661,
5    "usage_bytes": 123456,
6    "last_active_at": 1698107661,
7    "name": "my_vector_store",
8    "status": "completed",
```

```
9   "file_counts": {  
10     "in_progress": 0,  
11     "completed": 100,  
12     "cancelled": 0,  
13     "failed": 0,  
14     "total": 100  
15   },  
16   "last_used_at": 1698107661  
17 }
```

PREVIOUS

< **Moderations**

NEXT

Vector store files >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Vector store files

Vector store files represent files inside a vector store.

Related guide: [File Search](#)

Create vector store file

```
POST https://api.openai.com/v1/vector_stores/{vector_store_id}/files
```

Create a vector store file by attaching a [File](#) to a [vector store](#).

Path parameters

vector_store_id string Required

The ID of the vector store for which to create a File.

Request body

file_id string Required

A File ID that the vector store should use. Useful for tools like `file_search` that can access files.

attributes map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard. Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters, booleans, or numbers.

chunking_strategy object Optional

The chunking strategy used to chunk the file(s). If not set, will use the `auto` strategy.

› Show possible types

Returns

A vector store file object.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 vector_store_file = client.vector_stores.files.create(
5     vector_store_id="vs_abc123",
6     file_id="file-abc123"
7 )
8 print(vector_store_file)
```

Response

🔗

```
1 {
2     "id": "file-abc123",
3     "object": "vector_store.file",
4     "created_at": 1699061776,
5     "usage_bytes": 1234,
6     "vector_store_id": "vs_abcd",
7     "status": "completed",
8     "last_error": null
9 }
```

List vector store files

```
GET https://api.openai.com/v1/vector_stores/{vector_store_id}/files
```

Returns a list of vector store files.

Path parameters

vector_store_id string Required

The ID of the vector store that the files belong to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

filter string Optional

Filter by file status. One of `in_progress`, `completed`, `failed`, `cancelled`.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to `desc`

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of vector store file objects.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 vector_store_files = client.vector_stores.files.list(
5     vector_store_id="vs_abc123"
6 )
7 print(vector_store_files)
```

Response



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "id": "file-abc123",
6              "object": "vector_store.file",
7              "created_at": 1699061776,
8              "vector_store_id": "vs_abc123"
9          },
10         {
11             "id": "file-abc456",
12             "object": "vector_store.file",
13             "created_at": 1699061776,
14             "vector_store_id": "vs_abc123"
15         }
16     ],
17     "first_id": "file-abc123",
18     "last_id": "file-abc456",
19     "has_more": false
20 }
```

Retrieve vector store file

```
GET https://api.openai.com/v1/vector_stores/{vector_store_id}/files/{file_id}
```

Retrieves a vector store file.

Path parameters

file_id string Required

The ID of the file being retrieved.

vector_store_id string Required

The ID of the vector store that the file belongs to.

Returns

The [vector store file](#) object.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 vector_store_file = client.vector_stores.files.retrieve(
```

```
5   vector_store_id="vs_abc123",
6   file_id="file-abc123"
7 )
8 print(vector_store_file)
```

Response



```
1 {
2   "id": "file-abc123",
3   "object": "vector_store.file",
4   "created_at": 1699061776,
5   "vector_store_id": "vs_abcd",
6   "status": "completed",
7   "last_error": null
8 }
```

Retrieve vector store file content

```
GET https://api.openai.com/v1/vector_stores/{vector_store_id}/files/{file_id}/content
```

Retrieve the parsed contents of a vector store file.

Path parameters

file_id string Required

The ID of the file within the vector store.

vector_store_id string Required

The ID of the vector store.

Returns

The parsed contents of the specified vector store file.

Example request

curl ⚡

```
1 curl \
2 https://api.openai.com/v1/vector_stores/vs_abc123/files/file-abc123/content \
3 -H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

🔗

```
1 {
2   "file_id": "file-abc123",
3   "filename": "example.txt",
4   "attributes": {"key": "value"},
5   "content": [
```

```
6     {"type": "text", "text": "..."},  
7     ...  
8 ]  
9 }
```

Update vector store file attributes

```
POST https://api.openai.com/v1/vector_stores/{vector_store_id}/files/{file_id}
```

Update attributes on a vector store file.

Path parameters

file_id string Required

The ID of the file to update attributes.

vector_store_id string Required

The ID of the vector store the file belongs to.

Request body

attributes map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard. Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters, booleans, or numbers.

Returns

The updated [vector store file](#) object.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/vector_stores/{vector_store_id}/files/{file_id} \
2   -H "Authorization: Bearer $OPENAI_API_KEY" \
3   -H "Content-Type: application/json" \
4   -d '{"attributes": {"key1": "value1", "key2": 2}}'
```

Response

🔗

```
1 {
2   "id": "file-abc123",
3   "object": "vector_store.file",
4   "usage_bytes": 1234,
```

```
5  "created_at": 1699061776,  
6  "vector_store_id": "vs_abcd",  
7  "status": "completed",  
8  "last_error": null,  
9  "chunking_strategy": {...},  
10 "attributes": {"key1": "value1", "key2": 2}  
11 }
```

Delete vector store file

```
DELETE https://api.openai.com/v1/vector_stores/{vector_store_id}/files/{file_id}
```

Delete a vector store file. This will remove the file from the vector store but the file itself will not be deleted. To delete the file, use the [delete file endpoint](#).

Path parameters

file_id string Required

The ID of the file to delete.

vector_store_id string Required

The ID of the vector store that the file belongs to.

Returns

Deletion status

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 deleted_vector_store_file = client.vector_stores.files.delete(
5     vector_store_id="vs_abc123",
6     file_id="file-abc123"
7 )
8 print(deleted_vector_store_file)
```

Response

Copy

```
1 {
2   id: "file-abc123",
3   object: "vector_store.file.deleted",
4
5
```

```
  deleted: true  
}
```

The vector store file object Beta

A list of files attached to a vector store.

attributes map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard. Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters, booleans, or numbers.

chunking_strategy object

The strategy used to chunk the file.

› Show possible types

created_at integer

The Unix timestamp (in seconds) for when the vector store file was created.

id string

The identifier, which can be referenced in API endpoints.

last_error object

The last error associated with this vector store file. Will be `null` if there are no errors.

> Show properties

object string

The object type, which is always `vector_store.file`.

status string

The status of the vector store file, which can be either `in_progress`, `completed`, `cancelled`, or `failed`. The status `completed` indicates that the vector store file is ready for use.

usage_bytes integer

The total vector store usage in bytes. Note that this may be different from the original file size.

vector_store_id string

The ID of the vector store that the File is attached to.

OBJECT The vector store file object



```
1  {
2    "id": "file-abc123",
3    "object": "vector_store.file",
4    "usage_bytes": 1234,
```

```
5     "created_at": 1698107661,  
6     "vector_store_id": "vs_abc123",  
7     "status": "completed",  
8     "last_error": null,  
9     "chunking_strategy": {  
10        "type": "static",  
11        "static": {  
12            "max_chunk_size_tokens": 800,  
13            "chunk_overlap_tokens": 400  
14        }  
15    }  
16 }
```

PREVIOUS
< **Vector stores**

NEXT
Vector store file batches >

Vector store file batches



Vector store file batches represent operations to add multiple files to a vector store. Related guide:

[File Search](#)

Create vector store file batch



```
POST https://api.openai.com/v1/vector_stores/{vector_store_id}/file_batches
```

Create a vector store file batch.

Path parameters

vector_store_id string Required

The ID of the vector store for which to create a File Batch.

Request body

attributes map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard. Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters, booleans, or numbers.

chunking_strategy object Optional

The chunking strategy used to chunk the file(s). If not set, will use the `auto` strategy.

› Show possible types

file_ids array Optional

A list of File IDs that the vector store should use. Useful for tools like `file_search` that can access files. If `attributes` or `chunking_strategy` are provided, they will be applied to all files in the batch. Mutually exclusive with `files`.

files array Optional

A list of objects that each include a `file_id` plus optional `attributes` or `chunking_strategy`. Use this when you need to override metadata for specific files. The global `attributes` or `chunking_strategy` will be ignored and must be specified for each file. Mutually exclusive with `file_ids`.

› Show properties

Returns

A vector store file batch object.

Example request

```
from openai import OpenAI
client = OpenAI()

vector_store_file_batch = client.vector_stores.file_batches.create(
    vector_store_id="vs_abc123",
    files=[
        {
            "file_id": "file-abc123",
            "attributes": {"category": "finance"},
        },
        {
            "file_id": "file-abc456",
            "chunking_strategy": {
                "type": "static",
                "max_chunk_size_tokens": 1200,
                "chunk_overlap_tokens": 200,
            },
        },
    ],
)
print(vector_store_file_batch)
```

Response

```
{
    "id": "vsfb_abc123",
```

```
"object": "vector_store.file_batch",
"created_at": 1699061776,
"vector_store_id": "vs_abc123",
"status": "in_progress",
"file_counts": {
    "in_progress": 1,
    "completed": 1,
    "failed": 0,
    "cancelled": 0,
    "total": 0,
}
}
```

Retrieve vector store file batch



```
GET https://api.openai.com/v1/vector_stores/{vector_store_id}/file_batches/{batch_id}
```

Retrieves a vector store file batch.

Path parameters

batch_id string Required

The ID of the file batch being retrieved.

vector_store_id string Required

The ID of the vector store that the file batch belongs to.

Returns

The vector store file batch object.

Example request

```
from openai import OpenAI
client = OpenAI()

vector_store_file_batch = client.vector_stores.file_batches.retrieve(
    vector_store_id="vs_abc123",
    batch_id="vsfb_abc123"
)
print(vector_store_file_batch)
```

Response

```
{
  "id": "vsfb_abc123",
  "object": "vector_store.file_batch",
  "created_at": 1699061776,
```

```
"vector_store_id": "vs_abc123",
"status": "in_progress",
"file_counts": {
    "in_progress": 1,
    "completed": 1,
    "failed": 0,
    "cancelled": 0,
    "total": 0,
}
}
```

Cancel vector store file batch



```
POST https://api.openai.com/v1/vector_stores/{vector_store_id}/file_batches/{batch_id}/cancel
```

Cancel a vector store file batch. This attempts to cancel the processing of files in this batch as soon as possible.

Path parameters

batch_id string Required

The ID of the file batch to cancel.

vector_store_id string Required

The ID of the vector store that the file batch belongs to.

Returns

The modified vector store file batch object.

Example request

```
from openai import OpenAI
client = OpenAI()

deleted_vector_store_file_batch = client.vector_stores.file_batches.cancel(
    vector_store_id="vs_abc123",
    file_batch_id="vsfb_abc123"
)
print(deleted_vector_store_file_batch)
```

Response

```
{
  "id": "vsfb_abc123",
  "object": "vector_store.file_batch",
  "created_at": 1699061776,
  "vector_store_id": "vs_abc123",
```

```
"status": "in_progress",
"file_counts": {
  "in_progress": 12,
  "completed": 3,
  "failed": 0,
  "cancelled": 0,
  "total": 15,
}
}
```

List vector store files in a batch



```
GET https://api.openai.com/v1/vector_stores/{vector_store_id}/file_batches/{batch_id}/files
```

Returns a list of vector store files in a batch.

Path parameters

batch_id string Required

The ID of the file batch that the files belong to.

vector_store_id string Required

The ID of the vector store that the files belong to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include after=obj_foo in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with obj_foo, your subsequent call can include before=obj_foo in order to fetch the previous page of the list.

filter string Optional

Filter by file status. One of `in_progress`, `completed`, `failed`, `cancelled`.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of vector store file objects.

Example request

```
from openai import OpenAI
client = OpenAI()

vector_store_files = client.vector_stores.file_batches.list_files(
    vector_store_id="vs_abc123",
    batch_id="vsfb_abc123"
)
print(vector_store_files)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "file-abc123",
      "object": "vector_store.file",
      "created_at": 1699061776,
      "vector_store_id": "vs_abc123"
    },
    {
      "id": "file-abc456",
      "object": "vector_store.file",
    }
  ]
}
```

```
"created_at": 1699061776,  
  "vector_store_id": "vs_abc123"  
}  
],  
  "first_id": "file-abc123",  
  "last_id": "file-abc456",  
  "has_more": false  
}
```

The vector store files batch object Beta



A batch of files attached to a vector store.

created_at integer

The Unix timestamp (in seconds) for when the vector store files batch was created.

file_counts object

> Show properties

id string

The identifier, which can be referenced in API endpoints.

object string

The object type, which is always `vector_store.file_batch`.

status string

The status of the vector store files batch, which can be either `in_progress`, `completed`, `cancelled` or `failed`.

vector_store_id string

The ID of the vector store that the File is attached to.

OBJECT The vector store files batch object

```
{  
  "id": "vsfb_123",  
  "object": "vector_store.files_batch",  
  "created_at": 1698107661,  
  "vector_store_id": "vs_abc123",  
  "status": "completed",  
  "file_counts": {  
    "in_progress": 0,  
    "completed": 100,  
    "failed": 0,  
    "cancelled": 0,  
    "total": 100  
  }  
}
```

< PREVIOUS
Vector store files

NEXT >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

ChatKit Beta

Manage ChatKit sessions, threads, and file uploads for internal integrations.

Create ChatKit session Beta

```
POST https://api.openai.com/v1/chatkit/sessions
```

Create a ChatKit session.

Request body

user string Required

A free-form string that identifies your end user; ensures this Session can access other objects that have the same `user` scope.

workflow object Required

Workflow that powers the session.

› Show properties

chatkit_configuration object Optional

Optional overrides for ChatKit runtime configuration features

› Show properties

expires_after object Optional

Optional override for session expiration timing in seconds from creation. Defaults to 10 minutes.

› Show properties

rate_limits object Optional

Optional override for per-minute request limits. When omitted, defaults to 10.

› Show properties

Returns

Returns a [ChatKit session](#) object.

Example request

python ⚙️

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 chat_session = client.beta.chatkit.sessions.create(
5     user="user",
6     workflow={
7         "id": "id"
8     },
9 )
10 print(chat_session.id)
```

Response

🔗

```
1 {
2     "client_secret": "chatkit_token_123",
3     "expires_at": 1735689600,
4     "workflow": {
5         "id": "workflow_alpha",
6         "version": "2024-10-01"
7     },
8     "scope": {
9         "project": "alpha",
10        "environment": "staging"
11    },
12    "max_requests_per_1_minute": 60,
13    "max_requests_per_session": 500,
```

```
14     "status": "active"  
15 }
```

Cancel chat session Beta

POST https://api.openai.com/v1/chatkit/sessions/{session_id}/cancel

Cancel an active ChatKit session and return its most recent metadata.

Path parameters

session_id string Required

Unique identifier for the ChatKit session to cancel.

Returns

Returns the chat session after it has been cancelled. Cancelling prevents new requests from using the issued client secret.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 chat_session = client.beta.chatkit.sessions.cancel(
5     "cksess_123",
6 )
7 print(chat_session.id)
```

Response

🔗

```
1 {
2     "id": "cksess_123",
3     "object": "chatkit.session",
4     "workflow": {
5         "id": "workflow_alpha",
6         "version": "1"
7     },
8     "scope": {
9         "customer_id": "cust_456"
10    },
11    "max_requests_per_1_minute": 30,
12    "ttl_seconds": 900,
13    "status": "cancelled",
14    "cancelled_at": 1712345678
15 }
```

List ChatKit threads

Beta

```
GET https://api.openai.com/v1/chatkit/threads
```

List ChatKit threads with optional pagination and user filters.

Query parameters

after string Optional

List items created after this thread item ID. Defaults to null for the first page.

before string Optional

List items created before this thread item ID. Defaults to null for the newest results.

limit integer Optional

Maximum number of thread items to return. Defaults to 20.

order string Optional

Sort order for results by creation time. Defaults to `desc`.

user string Optional

Filter threads that belong to this user identifier. Defaults to null to return all users.

Returns

Returns a paginated list of ChatKit threads accessible to the request scope.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 page = client.beta.chatkit.threads.list()
5 page = page.data[0]
6 print(page.id)
```

Response

⚡

```
1 {
2   "data": [
3     {
4       "id": "cthr_abc123",
5       "object": "chatkit.thread",
```

```
6     "title": "Customer escalation"
7   },
8   {
9     "id": "cthr_def456",
10    "object": "chatkit.thread",
11    "title": "Demo feedback"
12  }
13 ],
14 "has_more": false,
15 "object": "list"
16 }
```

Retrieve ChatKit thread Beta

```
GET https://api.openai.com/v1/chatkit/threads/{thread_id}
```

Retrieve a ChatKit thread by its identifier.

Path parameters

thread_id string Required

Identifier of the ChatKit thread to retrieve.

Returns

Returns a [Thread](#) object.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 chatkit_thread = client.beta.chatkit.threads.retrieve(
5     "cthr_123",
6 )
7 print(chatkit_thread.id)
```

Response

🔗

```
1 {
2     "id": "cthr_abc123",
3     "object": "chatkit.thread",
4     "title": "Customer escalation",
5     "items": {
6         "data": [
7             {
```

```
8      "id": "cthi_user_001",
9      "object": "chatkit.thread_item",
10     "type": "user_message",
11     "content": [
12       {
13         "type": "input_text",
14         "text": "I need help debugging an onboarding issue."
15       }
16     ],
17     "attachments": []
18   },
19   {
20     "id": "cthi_assistant_002",
21     "object": "chatkit.thread_item",
22     "type": "assistant_message",
23     "content": [
24       {
25         "type": "output_text",
26         "text": "Let's start by confirming the workflow version you deployed."
27       }
28     ]
29   }
30 ],
31   "has_more": false
32 }
33 }
```

Delete ChatKit thread Beta

```
DELETE https://api.openai.com/v1/chatkit/threads/{thread_id}
```

Delete a ChatKit thread along with its items and stored attachments.

Path parameters

thread_id string Required

Identifier of the ChatKit thread to delete.

Returns

Returns a confirmation object for the deleted thread.

Example request

python ▾ 

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 thread = client.beta.chat_kit.threads.delete()
```

```
5     "cthr_123",
6 )
7 print(thread.id)
```

List ChatKit thread items Beta

GET https://api.openai.com/v1/chatkit/threads/{thread_id}/items

List items that belong to a ChatKit thread.

Path parameters

thread_id string Required

Identifier of the ChatKit thread whose items are requested.

Query parameters

after string Optional

List items created after this thread item ID. Defaults to null for the first page.

before string Optional

List items created before this thread item ID. Defaults to null for the newest results.

limit integer Optional

Maximum number of thread items to return. Defaults to 20.

order string Optional

Sort order for results by creation time. Defaults to `desc`.

Returns

Returns a [list of thread items](#) for the specified thread.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 page = client.beta.chatkit.threads.list_items(
5     thread_id="cthr_123",
6 )
7 page = page.data[0]
8 print(page)
```

Response



```
1  {
2      "data": [
3          {
4              "id": "cthi_user_001",
5              "object": "chatkit.thread_item",
6              "type": "user_message",
7              "content": [
8                  {
9                      "type": "input_text",
10                     "text": "I need help debugging an onboarding issue."
11                 }
12             ],
13             "attachments": []
14         },
15         {
16             "id": "cthi_assistant_002",
17             "object": "chatkit.thread_item",
18             "type": "assistant_message",
19             "content": [
20                 {
21                     "type": "output_text",
22                     "text": "Let's start by confirming the workflow version you deployed."
23                 }
24             ]
25         }
26     ],
27     "has_more": false,
```

```
28   "object": "list"
29 }
```

The chat session object

Represents a ChatKit session and its resolved configuration.

chatkit_configuration object

Resolved ChatKit feature configuration for the session.

› Show properties

client_secret string

Ephemeral client secret that authenticates session requests.

expires_at integer

Unix timestamp (in seconds) for when the session expires.

id string

Identifier for the ChatKit session.

max_requests_per_1_minute integer

Convenience copy of the per-minute request limit.

object string

Type discriminator that is always `chatkit.session`.

rate_limits object

Resolved rate limit values.

› Show properties

status string

Current lifecycle state of the session.

user string

User identifier associated with the session.

workflow object

Workflow metadata for the session.

› Show properties

OBJECT The chat session object



```
1  {
2    "id": "cksess_123",
```

```
3   "object": "chatkit.session",
4   "client_secret": "ek_token_123",
5   "expires_at": 1712349876,
6   "workflow": {
7     "id": "workflow_alpha",
8     "version": "2024-10-01"
9   },
10  "user": "user_789",
11  "rate_limits": {
12    "max_requests_per_1_minute": 60
13  },
14  "max_requests_per_1_minute": 60,
15  "status": "cancelled",
16  "chatkit_configuration": {
17    "automatic_thread_titling": {
18      "enabled": true
19    },
20    "file_upload": {
21      "enabled": true,
22      "max_file_size": 16,
23      "max_files": 20
24    },
25    "history": {
26      "enabled": true,
27      "recent_threads": 10
28    }
29  }
30 }
```

The thread object

Represents a ChatKit thread and its current status.

created_at integer

Unix timestamp (in seconds) for when the thread was created.

id string

Identifier of the thread.

object string

Type discriminator that is always `chatkit.thread`.

status object

Current status for the thread. Defaults to `active` for newly created threads.

› Show possible types

title string

Optional human-readable title for the thread. Defaults to null when no title has been generated.

user string

Free-form string that identifies your end user who owns the thread.

OBJECT The thread object



```
1  {
2    "id": "cthr_def456",
3    "object": "chatkit.thread",
4    "created_at": 1712345600,
5    "title": "Demo feedback",
6    "status": {
7      "type": "active"
8    },
9    "user": "user_456"
10 }
```

Thread Items

A paginated list of thread items rendered for the ChatKit API.

data array

A list of items

› Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

PREVIOUS

< **Vector store file batches**

NEXT

Containers >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

ChatKit Beta

Manage ChatKit sessions, threads, and file uploads for internal integrations.

Create ChatKit session Beta

```
POST https://api.openai.com/v1/chatkit/sessions
```

Create a ChatKit session.

Request body

user string Required

A free-form string that identifies your end user; ensures this Session can access other objects that have the same `user` scope.

workflow object Required

Workflow that powers the session.

› Show properties

chatkit_configuration object Optional

Optional overrides for ChatKit runtime configuration features

› Show properties

expires_after object Optional

Optional override for session expiration timing in seconds from creation. Defaults to 10 minutes.

› Show properties

rate_limits object Optional

Optional override for per-minute request limits. When omitted, defaults to 10.

› Show properties

Returns

Returns a [ChatKit session](#) object.

Example request

python ⚙️

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 chat_session = client.beta.chatkit.sessions.create(
5     user="user",
6     workflow={
7         "id": "id"
8     },
9 )
10 print(chat_session.id)
```

Response

🔗

```
1 {
2     "client_secret": "chatkit_token_123",
3     "expires_at": 1735689600,
4     "workflow": {
5         "id": "workflow_alpha",
6         "version": "2024-10-01"
7     },
8     "scope": {
9         "project": "alpha",
10        "environment": "staging"
11    },
12    "max_requests_per_1_minute": 60,
13    "max_requests_per_session": 500,
```

```
14     "status": "active"  
15 }
```

Cancel chat session Beta

POST https://api.openai.com/v1/chatkit/sessions/{session_id}/cancel

Cancel an active ChatKit session and return its most recent metadata.

Path parameters

session_id string Required

Unique identifier for the ChatKit session to cancel.

Returns

Returns the chat session after it has been cancelled. Cancelling prevents new requests from using the issued client secret.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 chat_session = client.beta.chatkit.sessions.cancel(
5     "cksess_123",
6 )
7 print(chat_session.id)
```

Response

🔗

```
1 {
2     "id": "cksess_123",
3     "object": "chatkit.session",
4     "workflow": {
5         "id": "workflow_alpha",
6         "version": "1"
7     },
8     "scope": {
9         "customer_id": "cust_456"
10    },
11    "max_requests_per_1_minute": 30,
12    "ttl_seconds": 900,
13    "status": "cancelled",
14    "cancelled_at": 1712345678
15 }
```

List ChatKit threads

Beta

```
GET https://api.openai.com/v1/chatkit/threads
```

List ChatKit threads with optional pagination and user filters.

Query parameters

after string Optional

List items created after this thread item ID. Defaults to null for the first page.

before string Optional

List items created before this thread item ID. Defaults to null for the newest results.

limit integer Optional

Maximum number of thread items to return. Defaults to 20.

order string Optional

Sort order for results by creation time. Defaults to `desc`.

user string Optional

Filter threads that belong to this user identifier. Defaults to null to return all users.

Returns

Returns a paginated list of ChatKit threads accessible to the request scope.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 page = client.beta.chatkit.threads.list()
5 page = page.data[0]
6 print(page.id)
```

Response

⚡

```
1 {
2   "data": [
3     {
4       "id": "cthr_abc123",
5       "object": "chatkit.thread",
```

```
6     "title": "Customer escalation"
7   },
8   {
9     "id": "cthr_def456",
10    "object": "chatkit.thread",
11    "title": "Demo feedback"
12  }
13 ],
14 "has_more": false,
15 "object": "list"
16 }
```

Retrieve ChatKit thread Beta

```
GET https://api.openai.com/v1/chatkit/threads/{thread_id}
```

Retrieve a ChatKit thread by its identifier.

Path parameters

thread_id string Required

Identifier of the ChatKit thread to retrieve.

Returns

Returns a [Thread](#) object.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 chatkit_thread = client.beta.chatkit.threads.retrieve(
5     "cthr_123",
6 )
7 print(chatkit_thread.id)
```

Response

🔗

```
1 {
2     "id": "cthr_abc123",
3     "object": "chatkit.thread",
4     "title": "Customer escalation",
5     "items": {
6         "data": [
7             {
```

```
8      "id": "cthi_user_001",
9      "object": "chatkit.thread_item",
10     "type": "user_message",
11     "content": [
12       {
13         "type": "input_text",
14         "text": "I need help debugging an onboarding issue."
15       }
16     ],
17     "attachments": []
18   },
19   {
20     "id": "cthi_assistant_002",
21     "object": "chatkit.thread_item",
22     "type": "assistant_message",
23     "content": [
24       {
25         "type": "output_text",
26         "text": "Let's start by confirming the workflow version you deployed."
27       }
28     ]
29   }
30 ],
31   "has_more": false
32 }
33 }
```

Delete ChatKit thread Beta

```
DELETE https://api.openai.com/v1/chatkit/threads/{thread_id}
```

Delete a ChatKit thread along with its items and stored attachments.

Path parameters

thread_id string Required

Identifier of the ChatKit thread to delete.

Returns

Returns a confirmation object for the deleted thread.

Example request

python ▾ 

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 thread = client.beta.chat_kit.threads.delete()
```

```
5     "cthr_123",
6 )
7 print(thread.id)
```

List ChatKit thread items Beta

GET https://api.openai.com/v1/chatkit/threads/{thread_id}/items

List items that belong to a ChatKit thread.

Path parameters

thread_id string Required

Identifier of the ChatKit thread whose items are requested.

Query parameters

after string Optional

List items created after this thread item ID. Defaults to null for the first page.

before string Optional

List items created before this thread item ID. Defaults to null for the newest results.

limit integer Optional

Maximum number of thread items to return. Defaults to 20.

order string Optional

Sort order for results by creation time. Defaults to `desc`.

Returns

Returns a [list of thread items](#) for the specified thread.

Example request

python ⚡

```
1 from openai import OpenAI
2
3 client = OpenAI()
4 page = client.beta.chatkit.threads.list_items(
5     thread_id="cthr_123",
6 )
7 page = page.data[0]
8 print(page)
```

Response



```
1  {
2      "data": [
3          {
4              "id": "cthi_user_001",
5              "object": "chatkit.thread_item",
6              "type": "user_message",
7              "content": [
8                  {
9                      "type": "input_text",
10                     "text": "I need help debugging an onboarding issue."
11                 }
12             ],
13             "attachments": []
14         },
15         {
16             "id": "cthi_assistant_002",
17             "object": "chatkit.thread_item",
18             "type": "assistant_message",
19             "content": [
20                 {
21                     "type": "output_text",
22                     "text": "Let's start by confirming the workflow version you deployed."
23                 }
24             ]
25         }
26     ],
27     "has_more": false,
```

```
28   "object": "list"
29 }
```

The chat session object

Represents a ChatKit session and its resolved configuration.

chatkit_configuration object

Resolved ChatKit feature configuration for the session.

› Show properties

client_secret string

Ephemeral client secret that authenticates session requests.

expires_at integer

Unix timestamp (in seconds) for when the session expires.

id string

Identifier for the ChatKit session.

max_requests_per_1_minute integer

Convenience copy of the per-minute request limit.

object string

Type discriminator that is always `chatkit.session`.

rate_limits object

Resolved rate limit values.

› Show properties

status string

Current lifecycle state of the session.

user string

User identifier associated with the session.

workflow object

Workflow metadata for the session.

› Show properties

OBJECT The chat session object



```
1  {
2    "id": "cksess_123",
```

```
3   "object": "chatkit.session",
4   "client_secret": "ek_token_123",
5   "expires_at": 1712349876,
6   "workflow": {
7     "id": "workflow_alpha",
8     "version": "2024-10-01"
9   },
10  "user": "user_789",
11  "rate_limits": {
12    "max_requests_per_1_minute": 60
13  },
14  "max_requests_per_1_minute": 60,
15  "status": "cancelled",
16  "chatkit_configuration": {
17    "automatic_thread_titling": {
18      "enabled": true
19    },
20    "file_upload": {
21      "enabled": true,
22      "max_file_size": 16,
23      "max_files": 20
24    },
25    "history": {
26      "enabled": true,
27      "recent_threads": 10
28    }
29  }
30 }
```

The thread object

Represents a ChatKit thread and its current status.

created_at integer

Unix timestamp (in seconds) for when the thread was created.

id string

Identifier of the thread.

object string

Type discriminator that is always `chatkit.thread`.

status object

Current status for the thread. Defaults to `active` for newly created threads.

› Show possible types

title string

Optional human-readable title for the thread. Defaults to null when no title has been generated.

user string

Free-form string that identifies your end user who owns the thread.

OBJECT The thread object



```
1  {
2    "id": "cthr_def456",
3    "object": "chatkit.thread",
4    "created_at": 1712345600,
5    "title": "Demo feedback",
6    "status": {
7      "type": "active"
8    },
9    "user": "user_456"
10 }
```

Thread Items

A paginated list of thread items rendered for the ChatKit API.

data array

A list of items

› Show possible types

first_id string

The ID of the first item in the list.

has_more boolean

Whether there are more items available.

last_id string

The ID of the last item in the list.

object string

The type of object returned, must be `list`.

PREVIOUS

< **Vector store file batches**

NEXT

Containers >

[Dashboard](#)[Docs](#)[API](#)[g](#)

OpenAI Platform

Containers

Create and manage containers for use with the Code Interpreter tool.

Create container

```
POST https://api.openai.com/v1/containers
```

Creates a container.

Request body

name string Required

Name of the container to create.

expires_after object Optional

Container expiration time in seconds relative to the 'anchor' time.

› Show properties

file_ids array Optional

IDs of files to copy to the container.

memory_limit string Optional

Optional memory limit for the container. Defaults to "1g".

Returns

The created [container](#) object.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/containers \
2   -H "Authorization: Bearer $OPENAI_API_KEY" \
3   -H "Content-Type: application/json" \
4   -d '{
5     "name": "My Container",
```

```
6     "memory_limit": "4g"  
7 }
```

Response



```
1 {  
2   "id": "cntr_682e30645a488191b6363a0cbefc0f0a025ec61b66250591",  
3   "object": "container",  
4   "created_at": 1747857508,  
5   "status": "running",  
6   "expires_after": {  
7     "anchor": "last_active_at",  
8     "minutes": 20  
9   },  
10  "last_active_at": 1747857508,  
11  "memory_limit": "4g",  
12  "name": "My Container"  
13 }
```

List containers

```
GET https://api.openai.com/v1/containers
```

Lists containers.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

a list of container objects.

Example request

curl ⚡

```
1 curl https://api.openai.com/v1/containers \
2   -H "Authorization: Bearer $OPENAI_API_KEY"
```

Response



```
1 {
2   "object": "list",
3   "data": [
4     {
5       "id": "cntr_682dfebaacac8198bbfe9c2474fb6f4a085685cbe3cb5863",
6       "object": "container",
7       "created_at": 1747844794,
8       "status": "running",
9       "expires_after": {
10         "anchor": "last_active_at",
11         "minutes": 20
12       },
13       "last_active_at": 1747844794,
14       "memory_limit": "4g",
15       "name": "My Container"
16     }
17   ],
18   "first_id": "container_123",
19   "last_id": "container_123",
20   "has_more": false
21 }
```

Retrieve container

```
GET https://api.openai.com/v1/containers/{container_id}
```

Retrieves a container.

Path parameters

container_id string Required

Returns

The [container](#) object.

Example request

curl ▾ □

```
1 curl https://api.openai.com/v1/containers/cntr_682dfebaacac8198bbfe9c2474fb6f4a085685  
2
```

```
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1  {
2      "id": "cntr_682dfebaacac8198bbfe9c2474fb6f4a085685cbe3cb5863",
3      "object": "container",
4      "created_at": 1747844794,
5      "status": "running",
6      "expires_after": {
7          "anchor": "last_active_at",
8          "minutes": 20
9      },
10     "last_active_at": 1747844794,
11     "memory_limit": "4g",
12     "name": "My Container"
13 }
```

Delete a container

```
DELETE https://api.openai.com/v1/containers/{container_id}
```

Delete a container.

Path parameters

container_id string **Required**

The ID of the container to delete.

Returns

Deletion Status

Example request

curl ⚡

```
1 curl -X DELETE https://api.openai.com/v1/containers/cntr_682dfebaacac8198bbfe9c2474fb6f4a085685cbe3cb5863  
2   -H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

🔗

```
1 {  
2   "id": "cntr_682dfebaacac8198bbfe9c2474fb6f4a085685cbe3cb5863",  
3   "object": "container.deleted",  
4   "deleted": true  
5 }
```

The container object

created_at integer

Unix timestamp (in seconds) when the container was created.

expires_after object

The container will expire after this time period. The anchor is the reference point for the expiration. The minutes is the number of minutes after the anchor before the container expires.

› Show properties

id string

Unique identifier for the container.

last_active_at integer

Unix timestamp (in seconds) when the container was last active.

memory_limit string

The memory limit configured for the container.

name string

Name of the container.

object string

The type of this object.

status string

Status of the container (e.g., active, deleted).

OBJECT The container object



```
1  {
2      "id": "cntr_682dfebaacac8198bbfe9c2474fb6f4a085685cbe3cb5863",
3      "object": "container",
4      "created_at": 1747844794,
5      "status": "running",
6      "expires_after": {
7          "anchor": "last_active_at",
8          "minutes": 20
9      },
10     "last_active_at": 1747844794,
11     "memory_limit": "1g",
12     "name": "My Container"
13 }
```

PREVIOUS

< ChatKit

NEXT

Container Files >

Container Files



Create and manage container files for use with the Code Interpreter tool.

Create container file



```
POST https://api.openai.com/v1/containers/{container_id}/files
```

Creates a container file.

Path parameters

container_id string Required

Request body

file file Optional

The File object (not file name) to be uploaded.

file_id string Optional

Name of the file to create.

Returns

The created container file object.

Example request

```
curl https://api.openai.com/v1/containers/cntr_682e0e7318108198aa783fd921ff305e08e78805b9fdbb04/fi
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -F file="@example.txt"
```

Response

```
{
  "id": "cfile_682e0e8a43c88191a7978f477a09bdf5",
  "object": "container.file",
  "created_at": 1747848842,
  "bytes": 880,
  "container_id": "cntr_682e0e7318108198aa783fd921ff305e08e78805b9fdbb04",
  "path": "/mnt/data/88e12fa445d32636f190a0b33daed6cb-tsconfig.json",
```

```
"source": "user"
```

List container files



```
GET https://api.openai.com/v1/containers/{container_id}/files
```

Lists container files.

Path parameters

container_id string Required

Query parameters

after string Optional

A cursor for use in pagination. **after** is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include after=obj_foo in order to fetch the next page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

a list of container file objects.

Example request

```
curl https://api.openai.com/v1/containers/cntr_682e0e7318108198aa783fd921ff305e08e78805b9fdbb04/fi  
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "cfile_682e0e8a43c88191a7978f477a09bdf5",  
      "object": "container.file",  
      "created_at": 1747848842,  
      "bytes": 880,  
      "container_id": "cntr_682e0e7318108198aa783fd921ff305e08e78805b9fdbb04",  
      "path": "/mnt/data/88e12fa445d32636f190a0b33daed6cb-tsconfig.json",  
      "source": "user"
```

```
        },
    ],
    "first_id": "cfile_682e0e8a43c88191a7978f477a09bdf5",
    "has_more": false,
    "last_id": "cfile_682e0e8a43c88191a7978f477a09bdf5"
}
```

Retrieve container file



```
GET https://api.openai.com/v1/containers/{container_id}/files/{file_id}
```

Retrieves a container file.

Path parameters

container_id string Required

file_id string Required

Returns

The container file object.

Example request

```
curl https://api.openai.com/v1/containers/container_123/files/file_456 \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
{
  "id": "cfile_682e0e8a43c88191a7978f477a09bdf5",
  "object": "container.file",
  "created_at": 1747848842,
  "bytes": 880,
  "container_id": "cntr_682e0e7318108198aa783fd921ff305e08e78805b9fdbb04",
  "path": "/mnt/data/88e12fa445d32636f190a0b33daed6cb-tsconfig.json",
  "source": "user"
}
```

Retrieve container file content



```
GET https://api.openai.com/v1/containers/{container_id}/files/{file_id}/content
```

Retrieves a container file content.

Path parameters

container_id string Required

file_id string Required

Returns

The contents of the container file.

Example request

```
curl https://api.openai.com/v1/containers/container_123/files/cfile_456/content \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
<binary content of the file>
```

Delete a container file



```
DELETE https://api.openai.com/v1/containers/{container_id}/files/{file_id}
```

Delete a container file.

Path parameters

container_id string Required

file_id string Required

Returns

Deletion Status

Example request

```
curl -X DELETE https://api.openai.com/v1/containers/cntr_682dfebaacac8198bbfe9c2474fb6f4a085685cbe  
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
{  
  "id": "cfile_682e0e8a43c88191a7978f477a09bdf5",  
  "object": "container.file.deleted",  
  "deleted": true  
}
```

The container file object



bytes integer

Size of the file in bytes.

container_id string

The container this file belongs to.

created_at integer

Unix timestamp (in seconds) when the file was created.

id string

Unique identifier for the file.

object string

The type of this object (`container.file`).

path string

Path of the file in the container.

source string

Source of the file (e.g., `user` , `assistant`).

OBJECT The container file object

```
{  
  "id": "cfile_682e0e8a43c88191a7978f477a09bdf5",  
  "object": "container.file",  
  "created_at": 1747848842,  
  "bytes": 880,  
  "container_id": "cntr_682e0e7318108198aa783fd921ff305e08e78805b9fdbb04",  
  "path": "/mnt/data/88e12fa445d32636f190a0b33daed6cb-tsconfig.json",  
  "source": "user"  
}
```

< PREVIOUS
Containers

NEXT

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.](#)

Create call



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

Create a new Realtime API call over WebRTC and receive the SDP answer needed to complete the peer connection.

Request body

sdp string Required

WebRTC Session Description Protocol (SDP) offer generated by the caller.

session object Optional

Optional session configuration to apply before the realtime session is created. Use the same parameters you would send in a [create client secret](#) request.

› Show properties

Returns

Returns **201 Created** with the SDP answer in the response body. The **Location** response header includes the call ID for follow-up requests, e.g., establishing a monitoring WebSocket or hanging up the call.

Example request

```
curl -X POST https://api.openai.com/v1/realtime/calls \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-F "sdp=<offer.sdp;type=application/sdp" \
-F 'session={"type": "realtime", "model": "gpt-realtime"};type=application/json'
```

Response

```
v=0
o=- 4227147428 1719357865 IN IP4 127.0.0.1
s=-
c=IN IP4 0.0.0.0
```

```
t=0 0
a=group:BUNDLE 0 1
a=msid-semantic:WMS *
a=fingerprint:sha-256 CA:92:52:51:B4:91:3B:34:DD:9C:0B:FB:76:19:7E:3B:F1:21:0F:32:2C:38:01:72:5D
m=audio 9 UDP/TLS/RTP/SAVPF 111 0 8
a=mid:0
a=ice-ufrag:kZ2qkHXX/u11
a=ice-pwd:uoD16Di50Gx3VbqgA3ymjEQV2kwi0jw6
a=setup:active
a=rtpmap:mux
a=rtpmap:111 opus/48000/2
a=candidate:993865896 1 udp 2130706431 4.155.146.196 3478 typ host ufrag kZ2qkHXX/u11
a=candidate:1432411780 1 tcp 1671430143 4.155.146.196 443 typ host tcptype passive ufrag kZ2qkHXX
m=application 9 UDP/DTLS/SCTP webrtc-datachannel
a=mid:1
a=sctp-port:5000
```

PREVIOUS
[Container Files](#)

NEXT
[Container Files](#)

Client secrets



REST API endpoint to generate ephemeral client secrets for use in client-side applications. Client secrets are short-lived tokens that can be passed to a client app, such as a web frontend or mobile client, which grants access to the Realtime API without leaking your main API key. You can configure a custom TTL for each client secret.

You can also attach session configuration options to the client secret, which will be applied to any sessions created using that client secret, but these can also be overridden by the client connection.

[Learn more about authentication with client secrets over WebRTC.](#)

Create client secret



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

Create a Realtime client secret with an associated session configuration.

Request body

expires_after object Optional

Configuration for the client secret expiration. Expiration refers to the time after which a client secret will no longer be valid for creating sessions. The session itself may continue after that time once started. A secret can be used to create multiple sessions until it expires.

› 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. The client secret is a string that looks like `ek_1234`.

Example request

```
curl -X POST https://api.openai.com/v1/realtime/client_secrets \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
  "expires_after": {
    "anchor": "created_at",
    "seconds": 600
  },
  "session": {
    "type": "realtime"
  }
}'
```

```
        "type": "realtime",
        "model": "gpt-realtime",
        "instructions": "You are a friendly assistant."
    }
}'
```

Response

```
{
  "value": "ek_68af296e8e408191a1120ab6383263c2",
  "expires_at": 1756310470,
  "session": {
    "type": "realtime",
    "object": "realtime.session",
    "id": "sess_C9CiUVUzUzYIssh3ELY1d",
    "model": "gpt-realtime",
    "output_modalities": [
      "audio"
    ],
    "instructions": "You are a friendly assistant.",
    "tools": [],
    "tool_choice": "auto",
    "max_output_tokens": "inf",
    "tracing": null,
    "truncation": "auto",
    "prompt": null,
    "expires_at": 0,
    "audio": {
      "input": {
```

```
"format": {  
    "type": "audio/pcm",  
    "rate": 24000  
},  
"transcription": null,  
"noise_reduction": null,  
"turn_detection": {  
    "type": "server_vad",  
}  
},  
"output": {  
    "format": {  
        "type": "audio/pcm",  
        "rate": 24000  
    },  
    "voice": "alloy",  
    "speed": 1.0  
}  
},  
"include": null  
}  
}
```

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

```
{  
  "value": "ek_68af296e8e408191a1120ab6383263c2",  
  "expires_at": 1756310470,  
  "session": {  
    "type": "realtime",  
    "object": "realtime.session",  
    "id": "sess_C9CiUVUzUzYIssh3ELY1d",  
    "model": "gpt-realtime-2025-08-25",  
    "output_modalities": [  
      "text",  
      "audio"  
    ]  
  }  
}
```

```
"audio"
],
"instructions": "You are a friendly assistant.",
"tools": [],
"tool_choice": "auto",
"max_output_tokens": "inf",
"tracing": null,
"truncation": "auto",
"prompt": null,
"expires_at": 0,
"audio": {
  "input": {
    "format": {
      "type": "audio/pcm",
      "rate": 24000
    },
    "transcription": null,
    "noise_reduction": null,
    "turn_detection": {
      "type": "server_vad",
      "threshold": 0.5,
      "prefix_padding_ms": 300,
      "silence_duration_ms": 200,
      "idle_timeout_ms": null,
      "create_response": true,
      "interrupt_response": true
    }
  },
  "output": {
```

```
"format": {  
    "type": "audio/pcm",  
    "rate": 24000  
},  
"voice": "alloy",  
"speed": 1.0  
}  
},  
"include": null  
}
```

PREVIOUS
< **Realtime**

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Calls

REST endpoints for controlling WebRTC or SIP calls with the Realtime API. Accept or reject an incoming call, transfer it to another destination, or hang up the call once you are finished.

Accept call

```
POST https://api.openai.com/v1/realtime/calls/{call_id}/accept
```

Accept an incoming SIP call and configure the realtime session that will handle it.

Path parameters

call_id string Required

The identifier for the call provided in the `realtime.call.incoming` webhook.

Request body

type string Required

The type of session to create. Always `realtime` for the Realtime API.

audio object Optional

Configuration for input and output audio.

› Show properties

include array Optional

Additional fields to include in server outputs.

`item.input_audio_transcription.logprobs` : Include logprobs for input audio transcription.

instructions string Optional

The default system instructions (i.e. system message) prepended to model calls. This field allows the client to guide the model on desired responses. The model can be instructed on response content and format, (e.g. "be extremely succinct", "act friendly", "here are examples of good responses") and on audio behavior (e.g. "talk quickly", "inject emotion into your voice", "laugh frequently"). The instructions are not guaranteed to be followed by the model, but they provide guidance to the model on the desired behavior.

Note that the server sets default instructions which will be used if this field is not set and are visible in the `session.created` event at the start of the session.

max_output_tokens integer or "inf" Optional

Maximum number of output tokens for a single assistant response, inclusive of tool calls. Provide an integer between 1 and 4096 to limit output tokens, or `inf` for the maximum available tokens for a given model.

Defaults to `inf`.

model string Optional

The Realtime model used for this session.

output_modalities array Optional Defaults to audio

The set of modalities the model can respond with. It defaults to `["audio"]`, indicating that the model will respond with audio plus a transcript. `["text"]` can be used to make the model respond with text only. It is not possible to request both `text` and `audio` at the same time.

prompt object Optional

Reference to a prompt template and its variables. [Learn more.](#)

› Show properties

tool_choice string or object Optional Defaults to auto

How the model chooses tools. Provide one of the string modes or force a specific function/MCP tool.

› Show possible types

tools array Optional

Tools available to the model.

› Show possible types

tracing "auto" or object Optional Defaults to null

Realtime API can write session traces to the [Traces Dashboard](#). Set to null to disable tracing. Once tracing is enabled for a session, the configuration cannot be modified.

`auto` will create a trace for the session with default values for the workflow name, group id, and metadata.

› Show possible types

truncation string or object Optional

When the number of tokens in a conversation exceeds the model's input token limit, the conversation be truncated, meaning messages (starting from the oldest) will not be included in the model's context. A 32k context model with 4,096 max output tokens can only include 28,224 tokens in the context before truncation occurs.

Clients can configure truncation behavior to truncate with a lower max token limit, which is an effective way to control token usage and cost.

Truncation will reduce the number of cached tokens on the next turn (busting the cache), since messages are dropped from the beginning of the context. However, clients can also configure truncation to retain messages up to a fraction of the maximum context size, which will reduce the need for future truncations and thus improve the cache rate.

Truncation can be disabled entirely, which means the server will never truncate but would instead return an error if the conversation exceeds the model's input token limit.

› Show possible types

Returns

Returns `200 OK` once OpenAI starts ringing the SIP leg with the supplied session configuration.

Example request

curl ⚡

```
1 curl -X POST https://api.openai.com/v1/realtim/calls/$CALL_ID/accept \
2   -H "Authorization: Bearer $OPENAI_API_KEY" \
3   -H "Content-Type: application/json" \
4   -d '{
5     "type": "realtime",
6     "model": "gpt-realtime",
7     "instructions": "You are Alex, a friendly concierge for Example Corp.",
8   }'
```

Reject call

```
POST https://api.openai.com/v1/realtimercalls/{call_id}/reject
```

Decline an incoming SIP call by returning a SIP status code to the caller.

Path parameters

call_id string Required

The identifier for the call provided in the [realtime.call.incoming](#) webhook.

Request body

status_code integer Optional

SIP response code to send back to the caller. Defaults to [603](#) (Decline) when omitted.

Returns

Returns [200 OK](#) after OpenAI sends the SIP status code to the caller.

Example request

curl ⚡

```
1 curl -X POST https://api.openai.com/v1/realtimercalls/${CALL_ID}/reject \
2   -H "Authorization: Bearer ${OPENAI_API_KEY}" \
```

```
3 -H "Content-Type: application/json" \
4 -d '{"status_code": 486}'
```

Refer call

```
POST https://api.openai.com/v1/realtime/calls/{call_id}/refer
```

Transfer an active SIP call to a new destination using the SIP REFER verb.

Path parameters

call_id string Required

The identifier for the call provided in the [realtime.call.incoming](#) webhook.

Request body

target_uri string Required

URI that should appear in the SIP Refer-To header. Supports values like `tel:+14155550123` or `sip:agent@example.com`.

Returns

Returns `200 OK` once the REFER is handed off to your SIP provider.

Example request

curl ⚡

```
1 curl -X POST https://api.openai.com/v1/realtime/calls/$CALL_ID/refer \
2   -H "Authorization: Bearer $OPENAI_API_KEY" \
3   -H "Content-Type: application/json" \
4   -d '{"target_uri": "tel:+14155550123"}'
```

Hang up call

POST `https://api.openai.com/v1/realtime/calls/{call_id}/hangup`

End an active Realtime API call, whether it was initiated over SIP or WebRTC.

Path parameters

call_id string **Required**

The identifier for the call. For SIP calls, use the value provided in the [realtime.call.incoming](#) webhook.

For WebRTC sessions, reuse the call ID returned in the [Location](#) header when creating the call with

[POST /v1/realtime/calls](#).

Returns

Returns [200 OK](#) when OpenAI begins terminating the realtime call.

Example request

curl ⚡

```
curl -X POST https://api.openai.com/v1/realtime/calls/$CALL_ID/hangup \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

PREVIOUS

< Client secrets

NEXT

Client events >

Calls



REST endpoints for controlling WebRTC or SIP calls with the Realtime API. Accept or reject an incoming call, transfer it to another destination, or hang up the call once you are finished.

Accept call



```
POST https://api.openai.com/v1/realtime/calls/{call_id}/accept
```

Accept an incoming SIP call and configure the realtime session that will handle it.

Path parameters

call_id string Required

The identifier for the call provided in the `realtime.call.incoming` webhook.

Request body

type string Required

The type of session to create. Always `realtime` for the Realtime API.

audio object Optional

Configuration for input and output audio.

› Show properties

include array Optional

Additional fields to include in server outputs.

`item.input_audio_transcription.logprobs` : Include logprobs for input audio transcription.

instructions string Optional

The default system instructions (i.e. system message) prepended to model calls. This field allows the client to guide the model on desired responses. The model can be instructed on response content and format, (e.g. "be extremely succinct", "act friendly", "here are examples of good responses") and on audio behavior (e.g. "talk quickly", "inject emotion into your voice", "laugh frequently"). The instructions are not guaranteed to be followed by the model, but they provide guidance to the model on the desired behavior.

Note that the server sets default instructions which will be used if this field is not set and are visible in the `session.created` event at the start of the session.

max_output_tokens integer or "inf" Optional

Maximum number of output tokens for a single assistant response, inclusive of tool calls. Provide an integer between 1 and 4096 to limit output tokens, or `inf` for the maximum available tokens for a given model. Defaults to `inf`.

model string Optional

The Realtime model used for this session.

output_modalities array Optional Defaults to audio

The set of modalities the model can respond with. It defaults to `["audio"]`, indicating that the model will respond with audio plus a transcript. `["text"]` can be used to make the model respond with text only. It is not possible to request both `text` and `audio` at the same time.

prompt object Optional

Reference to a prompt template and its variables. [Learn more](#).

› Show properties

tool_choice string or object Optional Defaults to auto

How the model chooses tools. Provide one of the string modes or force a specific function/MCP tool.

› Show possible types

tools array Optional

Tools available to the model.

› Show possible types

tracing "auto" or object Optional Defaults to null

Realtime API can write session traces to the [Traces Dashboard](#). Set to null to disable tracing. Once tracing is enabled for a session, the configuration cannot be modified.

`auto` will create a trace for the session with default values for the workflow name, group id, and metadata.

› Show possible types

truncation string or object Optional

When the number of tokens in a conversation exceeds the model's input token limit, the conversation be truncated, meaning messages (starting from the oldest) will not be included in the model's context. A 32k context model with 4,096 max output tokens can only include 28,224 tokens in the context before truncation occurs.

Clients can configure truncation behavior to truncate with a lower max token limit, which is an effective way to control token usage and cost.

Truncation will reduce the number of cached tokens on the next turn (busting the cache), since messages are dropped from the beginning of the context. However, clients can also configure truncation to retain messages up to a fraction of the maximum context size, which will reduce the need for future truncations and thus improve the cache rate.

Truncation can be disabled entirely, which means the server will never truncate but would instead return an error if the conversation exceeds the model's input token limit.

› Show possible types

Returns

Returns **200 OK** once OpenAI starts ringing the SIP leg with the supplied session configuration.

Example request

```
curl -X POST https://api.openai.com/v1/realtime/calls/$CALL_ID/accept \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
    "type": "realtime",
```

```
"model": "gpt-realtime",
"instructions": "You are Alex, a friendly concierge for Example Corp.",
}'
```

Reject call



POST https://api.openai.com/v1/realtime/calls/{call_id}/reject

Decline an incoming SIP call by returning a SIP status code to the caller.

Path parameters

call_id string Required

The identifier for the call provided in the [realtime.call.incoming](#) webhook.

Request body

status_code integer Optional

SIP response code to send back to the caller. Defaults to [603](#) (Decline) when omitted.

Returns

Returns `200 OK` after OpenAI sends the SIP status code to the caller.

Example request

```
curl -X POST https://api.openai.com/v1/realtime/calls/$CALL_ID/reject \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{"status_code": 486}'
```

Refer call



```
POST https://api.openai.com/v1/realtime/calls/{call_id}/refer
```

Transfer an active SIP call to a new destination using the SIP REFER verb.

Path parameters

call_id string **Required**

The identifier for the call provided in the `realtime.call.incoming` webhook.

Request body

`target_uri` string Required

URI that should appear in the SIP Refer-To header. Supports values like `tel:+14155550123` or `sip:agent@example.com`.

Returns

Returns `200 OK` once the REFER is handed off to your SIP provider.

Example request

```
curl -X POST https://api.openai.com/v1/realtime/calls/$CALL_ID/refer \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{"target_uri": "tel:+14155550123"}'
```

Hang up call



```
POST https://api.openai.com/v1/realtime/calls/{call_id}/hangup
```

End an active Realtime API call, whether it was initiated over SIP or WebRTC.

Path parameters

call_id string Required

The identifier for the call. For SIP calls, use the value provided in the `realtime.call.incoming` webhook. For WebRTC sessions, reuse the call ID returned in the `Location` header when creating the call with `POST /v1/realtime/calls`.

Returns

Returns `200 OK` when OpenAI begins terminating the realtime call.

Example request

```
curl -X POST https://api.openai.com/v1/realtime/calls/$CALL_ID/hangup \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

< PREVIOUS
Client secrets

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

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 configuration. The client may send this event at any time to update any field except for `voice` and `model`. `voice` can be updated only if there have been no other audio outputs yet.

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 in the `session.update`

are updated. To clear a field like `instructions`, pass an empty string. To clear a field like `tools`, pass an empty array. To clear a field like `turn_detection`, pass `null`.

event_id string

Optional client-generated ID used to identify this event. This is an arbitrary string that a client may assign. It will be passed back if there is an error with the event, but the corresponding `session.updated` event will not include it.

session object

Update the Realtime session. Choose either a realtime session or a transcription session.

› Show possible types

type string

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

OBJECT `session.update`

```
1  {
2    "type": "session.update",
3    "session": {
4      "type": "realtime",
5      "instructions": "You are a creative assistant that helps with design tasks.",
6      "tools": [
7        {
8          "type": "function",
9          "name": "display_color_palette",
```

```
10     "description": "Call this function when a user asks for a color palette.",  
11     "parameters": {  
12         "type": "object",  
13         "properties": {  
14             "theme": {  
15                 "type": "string",  
16                 "description": "Description of the theme for the color scheme."  
17             },  
18             "colors": {  
19                 "type": "array",  
20                 "description": "Array of five hex color codes based on the theme.",  
21                 "items": {  
22                     "type": "string",  
23                     "description": "Hex color code"  
24                 }  
25             }  
26         },  
27         "required": [  
28             "theme",  
29             "colors"  
30         ]  
31     }  
32 },  
33 ],  
34     "tool_choice": "auto"  
35 }  
36 }
```

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. A "commit" will create a new user message item in the conversation history from the buffer content and clear the buffer. Input audio transcription (if enabled) will be generated when the buffer is committed.

If VAD is enabled 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. Input audio noise reduction operates on writes to the audio buffer.

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 most 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.

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`.

OBJECT `conversation.item.create`



```
1  {
2      "type": "conversation.item.create",
3      "item": {
4          "type": "message",
5          "role": "user",
6          "content": [
7              {
8                  "type": "input_text",
9                  "text": "hi"
10             }
11         ]
12     },
13     "event_id": "b904fba0-0ec4-40af-8bbb-f908a9b26793",
14 }
```

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": "item_003"
5 }
```

conversation.item.truncate

Send this event to truncate a previous assistant message's 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 no 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 message items can be truncated.

type string

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

OBJECT `conversation.item.truncate`



```
1 {
2   "event_id": "event_678",
3   "type": "conversation.item.truncate",
4   "item_id": "item_002",
5   "content_index": 0,
6   "audio_end_ms": 1500
7 }
```

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": "item_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 by default.

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 `tools`. If these are set, they will override the Session's configuration for this Response only.

Responses can be created out-of-band of the default Conversation, meaning that they can have arbitrary input, and it's possible to disable writing the output to the Conversation. Only one Response can write to the default Conversation at a time, but otherwise multiple Responses can be created in parallel. The `metadata` field is a good way to disambiguate multiple simultaneous Responses.

Clients can set `conversation` to `none` to create a Response that does not write to the default Conversation. Arbitrary input can be provided with the `input` field, which is an array accepting raw Items and references to existing Items.

event_id string

Optional client-generated ID used to identify this event.

response object

Create a new Realtime response with these parameters

› Show properties

type string

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

OBJECT `response.create`



```
1 // Trigger a response with the default Conversation and no special parameters
2 {
3     "type": "response.create",
4 }
5
6 // Trigger an out-of-band response that does not write to the default Conversation
7 {
8     "type": "response.create",
9     "response": {
10         "instructions": "Provide a concise answer.",
11         "tools": [], // clear any session tools
12         "conversation": "none",
13         "output_modalities": ["text"],
14         "metadata": {
15             "response_purpose": "summarization"
16         },
17         "input": [
18             {
```

```
19      "type": "item_reference",
20      "id": "item_12345",
21    },
22  {
23    "type": "message",
24    "role": "user",
25    "content": [
26      {
27        "type": "input_text",
28        "text": "Summarize the above message in one sentence."
29      }
30    ]
31  }
32 ],
33 }
34 }
```

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. It's safe to call `response.cancel` even if no response is in progress, an error will be returned the session will remain unaffected.

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   "type": "response.cancel"
3   "response_id": "resp_12345",
4 }
```

output_audio_buffer.clear

WebRTC/SIP 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 }
```

PREVIOUS

Calls

NEXT

Server events

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

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 missing.",
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

The session configuration.

› Show possible types

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-realtime-2025-08-28",
9     "output_modalities": [
10       "audio"
11     ],
12     "instructions": "Your knowledge cutoff is 2023-10. You are a helpful, witty, and
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
35         }
36       },
37     }
```

```
37     "output": {  
38         "format": {  
39             "type": "audio/pcm",  
40             "rate": 24000  
41         },  
42         "voice": "marin",  
43         "speed": 1  
44     }  
45 },  
46     "include": null  
47 },  
48 }
```

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

The session configuration.

- › Show possible types

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-realtime-2025-08-28",
9      "output_modalities": [
10        "audio"
11      ],
12      "instructions": "Your knowledge cutoff is 2023-10. You are a helpful, witty, and
13      "tools": [
14        {
15          "type": "function",
16          "name": "display_color_palette",
17          "description": "\nCall this function when a user asks for a color palette.\r
18          "parameters": {
19            "type": "object",
20            "strict": true,
```

```
21     "properties": {
22       "theme": {
23         "type": "string",
24         "description": "Description of the theme for the color scheme."
25       },
26       "colors": {
27         "type": "array",
28         "description": "Array of five hex color codes based on the theme.",
29         "items": {
30           "type": "string",
31           "description": "Hex color code"
32         }
33       }
34     },
35     "required": [
36       "theme",
37       "colors"
38     ]
39   }
40 }
41 ],
42 "tool_choice": "auto",
43 "max_output_tokens": "inf",
44 "tracing": null,
45 "prompt": null,
46 "expires_at": 1756324832,
47 "audio": {
48   "input": {
49     "format": {
```

```
50      "type": "audio/pcm",
51      "rate": 24000
52    },
53    "transcription": null,
54    "noise_reduction": null,
55    "turn_detection": {
56      "type": "server_vad",
57      "threshold": 0.5,
58      "prefix_padding_ms": 300,
59      "silence_duration_ms": 200,
60      "idle_timeout_ms": null,
61      "create_response": true,
62      "interrupt_response": true
63    }
64  },
65  "output": {
66    "format": {
67      "type": "audio/pcm",
68      "rate": 24000
69    },
70    "voice": "marin",
71    "speed": 1
72  }
73},
74  "include": null
75},
76 }
```

conversation.item.added

Sent by the server when an Item is added to the default Conversation. This can happen in several cases:

- When the client sends a `conversation.item.create` event.
- When the input audio buffer is committed. In this case the item will be a user message containing the audio from the buffer.
- When the model is generating a Response. In this case the `conversation.item.added` event will be sent when the model starts generating a specific Item, and thus it will not yet have any content (and `status` will be `in_progress`).

The event will include the full content of the Item (except when model is generating a Response) except for audio data, which can be retrieved separately with a `conversation.item.retrieve` event if necessary.

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

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 }
```

conversation.item.done

Returned when a conversation item is finalized.

The event will include the full content of the Item except for audio data, which can be retrieved separately with a `conversation.item.retrieve` event if needed.

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

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_CCXLgMZPo3qioWCeQa4WH",
4      "previous_item_id": "item_CCXLecNJIVR2HuY3ABLj",
5      "item": {
6          "id": "item_CCXLfxmM5sXVJVz4mCa2S",
7          "type": "message",
8          "status": "completed",
9          "role": "assistant",
10         "content": [
11             {
12                 "type": "output_audio",
13                 "transcript": "Oh, I can hear you loud and clear! Sounds like we're connected."
14             }
15         ]
16     }
17 }
```

conversation.item.retrieved

Returned when a conversation item is retrieved with `conversation.item.retrieve`. This is provided as a way to fetch the server's representation of an item, for example to get access to the post-processed audio data after noise cancellation and VAD. It includes the full content of the Item, including audio data.

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    "type": "conversation.item.retrieved",
3    "event_id": "event_CCXGSizgEppa2d4XbKA7K",
```

```
4  "item": {  
5      "id": "item_CCXGRxbY0n6WE4EszhF5w",  
6      "object": "realtime.item",  
7      "type": "message",  
8      "status": "completed",  
9      "role": "assistant",  
10     "content": [  
11         {  
12             "type": "audio",  
13             "transcript": "Yes, I can hear you loud and clear. How can I help you today?",  
14             "audio": "8//2//v/9//q/+//+P/s...",  
15             "format": "pcm16"  
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 (when VAD is enabled). 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 item containing the audio that is being transcribed.

logprobs array

The log probabilities of the transcription.

› Show properties

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, this is billed according to the ASR model's pricing rather than the realtime model's pricing.

› Show possible types

OBJECT `conversation.item.input_audio_transcription.completed`



```
1  {
2    "type": "conversation.item.input_audio_transcription.completed",
3    "event_id": "event_CCXGRvtUVraxy5SJAnNOWZ",
4    "item_id": "item_CCXGQ4e1ht4cOraEYcuR2",
5    "content_index": 0,
6    "transcript": "Hey, can you hear me?",
7    "usage": {
8      "type": "tokens",
9      "total_tokens": 22,
10     "input_tokens": 13,
11     "input_token_details": {
12       "text_tokens": 0,
13       "audio_tokens": 13
14     },
15     "output_tokens": 9
}
```

```
16    }
17 }
```

conversation.item.input_audio_transcription.delta

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

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 containing the audio that is being transcribed.

logprobs array

The log probabilities of the transcription. These can be enabled by configuring the session with

"include": ["item.input_audio_transcription.logprobs"] . Each entry in the array corresponds a log probability of which token would be selected for this chunk of transcription. This can help to identify if it was possible there were multiple valid options for a given chunk of transcription.

› Show properties

type string

The event type, must be conversation.item.input_audio_transcription.delta .

OBJECT conversation.item.input_audio_transcription.delta



```
1 {
2   "type": "conversation.item.input_audio_transcription.delta",
3   "event_id": "event_CCXGRxsAimPAs8kS2Wc7Z",
4   "item_id": "item_CCXGQ4e1ht4cOraEYcuR2",
5   "content_index": 0,
6   "delta": "Hey",
7   "obfuscation": "aLxx0jTEci0Ge"
8 }
```

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.

text string

The text for this segment.

type string

The event type, must be `conversation.item.input_audio_transcription.segment`.

OBJECT `conversation.item.input_audio_transcription.segment`



```
1  {
2      "event_id": "event_6501",
3      "type": "conversation.item.input_audio_transcription.segment",
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 }
```

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_transcription.failed`



```
1  {
2      "event_id": "event_2324",
3      "type": "conversation.item.input_audio_transcription.failed",
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 transcribed.",
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

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.dtmf_event_received

SIP Only: Returned when an DTMF event is received. A DTMF event is a message that represents a telephone keypad press (0–9, *, #, A–D). The `event` property is the keypad that the user press. The `received_at` is the UTC Unix Timestamp that the server received the event.

event string

The telephone keypad that was pressed by the user.

received_at integer

UTC Unix Timestamp when DTMF Event was received by server.

type string

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

OBJECT `input_audio_buffer.dtmf_event_received`



```
1 {
2   "type": "input_audio_buffer.dtmf_event_received",
3   "event": "9",
4   "received_at": 1763605109,
5 }
```

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 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`.

OBJECT `input_audio_buffer.speech_started`



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

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_stopped",
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. This is configured with `idle_timeout_ms` in the `turn_detection` settings of the session, and it indicates that there hasn't been any speech detected for the configured duration.

The `audio_start_ms` and `audio_end_ms` fields indicate the segment of audio after the last model response up to the triggering time, as an offset from the beginning of audio written to the input audio buffer. This means it demarcates the segment of audio that was silent and the difference between the start and end values will roughly match the configured timeout.

The empty audio will be committed to the conversation as an `input_audio` item (there will be a `input_audio_buffer.committed` event) and a model response will be generated. There may be speech that didn't trigger VAD but is still detected by the model, so the model may respond with something relevant to the conversation or a prompt to continue speaking.

audio_end_ms integer

Millisecond offset of audio written to the input audio buffer at the time the timeout was triggered.

audio_start_ms integer

Millisecond offset of audio written to the input audio buffer that was after the playback time of the last model response.

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   "type": "input_audio_buffer.timeout_triggered",
3   "event_id": "event_CEKKrf1KTGvemCPyiJTJ2",
4   "audio_start_ms": 13216,
5   "audio_end_ms": 19232,
6   "item_id": "item_CEKKrWH0GiwN0ET97NUZc"
7 }
```

output_audio_buffer.started

WebRTC/SIP Only: Emitted when the server begins streaming audio to the client. This event is emitted after an audio content part has been added (`response.content_part.added`) to the response. [Learn more.](#)

event_id string

The unique ID of the server event.

response_id string

The unique ID of the response that produced the audio.

type string

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

OBJECT `output_audio_buffer.started`

```
1 {
2   "event_id": "event_abc123",
3   "type": "output_audio_buffer.started",
4   "response_id": "resp_abc123"
5 }
```

output_audio_buffer.stopped

WebRTC/SIP Only: Emitted when the output audio buffer has been completely drained on the server, and no more audio is forthcoming. This event is emitted after the full response data has been sent to the client (`response.done`). [Learn more.](#)

event_id string

The unique ID of the server event.

response_id string

The unique ID of the response that produced the audio.

type string

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

OBJECT `output_audio_buffer.stopped`



```
1 {
2   "event_id": "event_abc123",
3   "type": "output_audio_buffer.stopped",
4   "response_id": "resp_abc123"
5 }
```

output_audio_buffer.cleared

WebRTC/SIP Only: Emitted when the output audio buffer is cleared. This happens either in VAD mode when the user has interrupted (`input_audio_buffer.speech_started`), or when the client has emitted the `output_audio_buffer.clear` event to manually cut off the current audio response. [Learn more.](#)

event_id string

The unique ID of the server event.

response_id string

The unique ID of the response that produced the audio.

type string

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

OBJECT `output_audio_buffer.cleared`



```
1 {
2     "event_id": "event_abc123",
3     "type": "output_audio_buffer.cleared",
```

```
4     "response_id": "resp_abc123"  
5 }
```

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_C9G8pqbTEddBSIxBN6Os",
4      "response": {
5          "object": "realtime.response",
6          "id": "resp_C9G8p7IH2WxLbkgPNouYL",
7          "status": "in_progress",
8          "status_details": null,
9          "output": [],
10         "conversation_id": "conv_C9G8mmBkLhQJwCon3hoJN",
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 }
```

```
},  
}
```

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.

Clients should check the `status` field of the Response to determine if it was successful (`completed`) or if there was another outcome: `cancelled`, `failed`, or `incomplete`.

A response will contain all output items that were generated during the response, excluding any audio content.

`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      "type": "response.done",
3      "event_id": "event_CCXHxcMy86rrKhBLDdqCh",
4      "response": {
5          "object": "realtime.response",
6          "id": "resp_CCXHw0UJld10EzIUXQCNh",
7          "status": "completed",
8          "status_details": null,
9          "output": [
10             {
11                 "id": "item_CCXHwGjjDUfOXbiyS1K7i",
12                 "type": "message",
13                 "status": "completed",
14                 "role": "assistant",
15                 "content": [
16                     {
17                         "type": "output_audio",
18                         "transcript": "Loud and clear! I can hear you perfectly. How can I help"
19                     }
20                 ]
21             }
22         ],
23         "conversation_id": "conv_CCXHsurMKcaVxIZvaCI5m",
```

```
24     "output_modalities": [
25         "audio"
26     ],
27     "max_output_tokens": "inf",
28     "audio": {
29         "output": {
30             "format": {
31                 "type": "audio/pcm",
32                 "rate": 24000
33             },
34             "voice": "alloy"
35         }
36     },
37     "usage": {
38         "total_tokens": 253,
39         "input_tokens": 132,
40         "output_tokens": 121,
41         "input_token_details": {
42             "text_tokens": 119,
43             "audio_tokens": 13,
44             "image_tokens": 0,
45             "cached_tokens": 64,
46             "cached_tokens_details": {
47                 "text_tokens": 64,
48                 "audio_tokens": 0,
49                 "image_tokens": 0
50             }
51         },
52         "output_token_details": {
```

```
53     "text_tokens": 30,  
54     "audio_tokens": 91  
55   }  
56 },  
57   "metadata": null  
58 }  
59 }
```

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 with that."
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

The ID of the response.

type string

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

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 that."
11     }
12 }
```

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.delta`



```
1 {
2   "event_id": "event_4546",
3   "type": "response.output_audio_transcript.delta",
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_transcript.done",
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 you today?"
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.delta`



```
1 {
2   "event_id": "event_5354",
3   "type": "response.function_call_arguments.delta",
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_arguments.done",
```

```
4   "response_id": "resp_002",
5   "item_id": "fc_001",
6   "output_index": 0,
7   "call_id": "call_001",
8   "arguments": "{\"location\": \"San Francisco\"}"
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

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.delta",
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.done",
4     "response_id": "resp_001",
5     "item_id": "mcp_call_001",
6     "output_index": 0,
7     "arguments": "{\"q\": \"docs\"}"
8 }
```

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 }
```

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 }
```

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 }
```

PREVIOUS

< **Client events**

NEXT

Chat Completions >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Chat Completions

The Chat Completions API endpoint will generate a model response from a list of messages comprising a conversation.

Related guides:

- [Quickstart](#)
- [Text inputs and outputs](#)
- [Image inputs](#)
- [Audio inputs and outputs](#)
- [Structured Outputs](#)
- [Function calling](#)
- [Conversation state](#)

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Create chat completion

POST <https://api.openai.com/v1/chat/completions>

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Creates a model response for the given chat conversation. Learn more in the [text generation](#), [vision](#), and [audio](#) guides.

Parameter support can differ depending on the model used to generate the response, particularly for newer reasoning models. Parameters that are only supported for reasoning models are noted below. For the current state of unsupported parameters in reasoning models, refer to the [reasoning guide](#).

Request body

messages array Required

A list of messages comprising the conversation so far. Depending on the [model](#) you use, different message types (modalities) are supported, like [text](#), [images](#), and [audio](#).

› Show possible types

model string Required

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

audio object or null Optional

Parameters for audio output. Required when audio output is requested with `modalities: ["audio"]`.

[Learn more.](#)

› Show properties

frequency_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

function_call Deprecated string or object Optional

Deprecated in favor of `tool_choice`.

Controls which (if any) function is called by the model.

`none` means the model will not call a function and instead generates a message.

`auto` means the model can pick between generating a message or calling a function.

Specifying a particular function via `{"name": "my_function"}` forces the model to call that function.

`none` is the default when no functions are present. `auto` is the default if functions are present.

› Show possible types

functions Deprecated array Optional

Deprecated in favor of `tools`.

A list of functions the model may generate JSON inputs for.

› Show properties

logit_bias map Optional Defaults to null

Modify the likelihood of specified tokens appearing in the completion.

Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

logprobs boolean or null Optional Defaults to false

Whether to return log probabilities of the output tokens or not. If true, returns the log probabilities of each output token returned in the `content` of `message`.

max_completion_tokens integer or null Optional

An upper bound for the number of tokens that can be generated for a completion, including visible output tokens and reasoning tokens.

max_tokens Deprecated integer or null Optional

The maximum number of tokens that can be generated in the chat completion. This value can be used to control costs for text generated via API.

This value is now deprecated in favor of `max_completion_tokens`, and is not compatible with o-series models.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

modalities array Optional

Output types that you would like the model to generate. Most models are capable of generating text, which is the default:

`["text"]`

The `gpt-4o-audio-preview` model can also be used to generate audio. To request that this model generate both text and audio responses, you can use:

`["text", "audio"]`

n integer or null Optional Defaults to 1

How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep `n` as `1` to minimize costs.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable parallel function calling during tool use.

prediction object Optional

Configuration for a Predicted Output, which can greatly improve response times when large parts of the model response are known ahead of time. This is most common when you are regenerating a file with only minor changes to most of the content.

> Show possible types

presence_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. Learn more.

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format object Optional

An object specifying the format that the model must output.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables the older JSON mode, which ensures the message the model generates is valid JSON. Using `json_schema` is preferred for models that support it.

› Show possible types

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

seed Deprecated integer or null Optional

This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same `seed` and parameters should return the same result. Determinism is not guaranteed, and you should refer to the `system_fingerprint` response parameter to monitor changes in the backend.

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

stop string / array / null Optional Defaults to null

Not supported with latest reasoning models `o3` and `o4-mini`.

Up to 4 sequences where the API will stop generating further tokens. The returned text will not contain the stop sequence.

store boolean or null Optional Defaults to false

Whether or not to store the output of this chat completion request for use in our [model distillation](#) or [evals](#) products.

Supports text and image inputs. Note: image inputs over 8MB will be dropped.

stream boolean or null Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information, along with the [streaming responses](#) guide for more information on how to handle the streaming events.

stream_options object Optional Defaults to null

Options for streaming response. Only set this when you set `stream: true`.

› Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tool and instead generates a message. `auto` means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools. Specifying a particular tool via `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

`none` is the default when no tools are present. `auto` is the default if tools are present.

› Show possible types

tools array Optional

A list of tools the model may call. You can provide either [custom tools](#) or [function tools](#).

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more.](#)

verbosity string Optional Defaults to medium

Constrains the verbosity of the model's response. Lower values will result in more concise responses, while higher values will result in more verbose responses. Currently supported values are `low`, `medium`, and `high`.

web_search_options object Optional

This tool searches the web for relevant results to use in a response. Learn more about the [web search tool](#).

› Show properties

Returns

Returns a [chat completion](#) object, or a streamed sequence of [chat completion chunk](#) objects if the request is streamed.

Default [Image input](#) [Streaming](#) [Functions](#) [Logprobs](#)

Example request

python 

```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3
4 completion = client.chat.completions.create(
5     model="gpt-5.2",
6     messages=[
7         {"role": "developer", "content": "You are a helpful assistant."},
8         {"role": "user", "content": "Hello!"}
9     ]
10 )
11
12 print(completion.choices[0].message)
```

Response



```
1 {
2     "id": "chatcmpl-B9MBs8CjcvOU2jLn4n570S5qMJKcT",
3     "object": "chat.completion",
4     "created": 1741569952,
5     "model": "gpt-4.1-2025-04-14",
6     "choices": [
7         {
8             "index": 0,
9             "message": {
10                 "role": "assistant",
11                 "content": "Hello! How can I assist you today?",
12                 "refusal": null,
13                 "annotations": []
14             },
15             "logprobs": null,
```

```
16     "finish_reason": "stop"
17   }
18 ],
19 "usage": {
20   "prompt_tokens": 19,
21   "completion_tokens": 10,
22   "total_tokens": 29,
23   "prompt_tokens_details": {
24     "cached_tokens": 0,
25     "audio_tokens": 0
26   },
27   "completion_tokens_details": {
28     "reasoning_tokens": 0,
29     "audio_tokens": 0,
30     "accepted_prediction_tokens": 0,
31     "rejected_prediction_tokens": 0
32   }
33 },
34 "service_tier": "default"
35 }
```

Get chat completion

```
GET https://api.openai.com/v1/chat/completions/{completion_id}
```

Get a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

`completion_id` string Required

The ID of the chat completion to retrieve.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 print(first_completion)
```



Response

```
1  {
2      "object": "chat.completion",
3      "id": "chatcmpl-abc123",
4      "model": "gpt-4o-2024-08-06",
5      "created": 1738960610,
6      "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7      "tool_choice": null,
8      "usage": {
9          "total_tokens": 31,
10         "completion_tokens": 18,
11         "prompt_tokens": 13
12     },
13     "seed": 4944116822809979520,
14     "top_p": 1.0,
15     "temperature": 1.0,
16     "presence_penalty": 0.0,
17     "frequency_penalty": 0.0,
18     "system_fingerprint": "fp_50cad350e4",
19     "input_user": null,
20     "service_tier": "default",
21     "tools": null,
22     "metadata": {},
23     "choices": [
24         {
25             "index": 0,
26             "message": {
27                 "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture"
28             }
29         }
30     ]
31 }
```

```
28     "role": "assistant",
29     "tool_calls": null,
30     "function_call": null
31   },
32   "finish_reason": "stop",
33   "logprobs": null
34 }
35 ],
36 "response_format": null
37 }
```

Get chat messages

```
GET https://api.openai.com/v1/chat/completions/{completion_id}/messages
```

Get the messages in a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve messages from.

Query parameters

after string Optional

Identifier for the last message from the previous pagination request.

limit integer Optional Defaults to 20

Number of messages to retrieve.

order string Optional Defaults to asc

Sort order for messages by timestamp. Use `asc` for ascending order or `desc` for descending order.

Defaults to `asc`.

Returns

A list of [messages](#) for the specified chat completion.

Example request

python

```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 messages = client.chat.completions.messages.list(completion_id=first_id)
8 print(messages)
```

Response



```
1 {
2     "object": "list",
3     "data": [
4         {
5             "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6             "role": "user",
7             "content": "write a haiku about ai",
8             "name": null,
9             "content_parts": null
10        }
11    ],
12    "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
13    "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
14    "has_more": false
15 }
```

List Chat Completions

```
GET https://api.openai.com/v1/chat/completions
```

List stored Chat Completions. Only Chat Completions that have been stored with the `store` parameter set to `true` will be returned.

Query parameters

after string Optional

Identifier for the last chat completion from the previous pagination request.

limit integer Optional Defaults to 20

Number of Chat Completions to retrieve.

metadata object or null Optional

A list of metadata keys to filter the Chat Completions by. Example:

```
metadata[key1]=value1&metadata[key2]=value2
```

model string Optional

The model used to generate the Chat Completions.

order string Optional Defaults to asc

Sort order for Chat Completions by timestamp. Use `asc` for ascending order or `desc` for descending order. Defaults to `asc`.

Returns

A list of [Chat Completions](#) matching the specified filters.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 print(completions)
```

Response

🔗

```
1 {
2     "object": "list",
3     "data": [
4         {
5             "object": "chat.completion",
6             "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
```

```
7     "model": "gpt-4.1-2025-04-14",
8     "created": 1738960610,
9     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10    "tool_choice": null,
11    "usage": {
12        "total_tokens": 31,
13        "completion_tokens": 18,
14        "prompt_tokens": 13
15    },
16    "seed": 4944116822809979520,
17    "top_p": 1.0,
18    "temperature": 1.0,
19    "presence_penalty": 0.0,
20    "frequency_penalty": 0.0,
21    "system_fingerprint": "fp_50cad350e4",
22    "input_user": null,
23    "service_tier": "default",
24    "tools": null,
25    "metadata": {},
26    "choices": [
27        {
28            "index": 0,
29            "message": {
30                "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31                "role": "assistant",
32                "tool_calls": null,
33                "function_call": null
34            },
35            "finish_reason": "stop",

```

```
36         "logprobs": null
37     }
38 ],
39     "response_format": null
40 }
41 ],
42 "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
43 "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
44 "has_more": false
45 }
```

Update chat completion

```
POST https://api.openai.com/v1/chat/completions/{completion_id}
```

Modify a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be modified. Currently, the only supported modification is to update the `metadata` field.

Path parameters

`completion_id` string **Required**

The ID of the chat completion to update.

Request body

metadata map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python 

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 updated_completion = client.chat.completions.update(completion_id=first_id, request_
7 print(updated_completion)
```



Response

```
1  {
2      "object": "chat.completion",
3      "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
4      "model": "gpt-4o-2024-08-06",
5      "created": 1738960610,
6      "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7      "tool_choice": null,
8      "usage": {
9          "total_tokens": 31,
10         "completion_tokens": 18,
11         "prompt_tokens": 13
12     },
13     "seed": 4944116822809979520,
14     "top_p": 1.0,
15     "temperature": 1.0,
16     "presence_penalty": 0.0,
17     "frequency_penalty": 0.0,
18     "system_fingerprint": "fp_50cad350e4",
19     "input_user": null,
20     "service_tier": "default",
21     "tools": null,
22     "metadata": {
23         "foo": "bar"
24     },
25     "choices": [
26         {
27             "index": 0,
```

```
28     "message": {
29         "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture",
30         "role": "assistant",
31         "tool_calls": null,
32         "function_call": null
33     },
34     "finish_reason": "stop",
35     "logprobs": null
36 }
37 ],
38 "response_format": null
39 }
```

Delete chat completion

```
DELETE https://api.openai.com/v1/chat/completions/{completion_id}
```

Delete a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be deleted.

Path parameters

completion_id string Required

The ID of the chat completion to delete.

Returns

A deletion confirmation object.

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 delete_response = client.chat.completions.delete(completion_id=first_id)
7 print(delete_response)
```

Response

Copy

```
1 {
2   "object": "chat.completion.deleted",
3   "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
```

```
4  "deleted": true  
5 }
```

The chat completion object

Represents a chat completion response returned by model, based on the provided input.

choices array

A list of chat completion choices. Can be more than one if **n** is greater than 1.

› Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created.

id string

A unique identifier for the chat completion.

model string

The model used for the chat completion.

object string

The object type, which is always `chat.completion`.

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to 'flex' or 'priority', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

system_fingerprint Deprecated string

This fingerprint represents the backend configuration that the model runs with.

Can be used in conjunction with the `seed` request parameter to understand when backend changes have been made that might impact determinism.

usage object

Usage statistics for the completion request.

› Show properties

OBJECT The chat completion object



```
1  {
2      "id": "chatcmpl-B9MHDbs1fkBeAs8l4bebGdFOJ6PeG",
3      "object": "chat.completion",
4      "created": 1741570283,
5      "model": "gpt-4o-2024-08-06",
6      "choices": [
7          {
8              "index": 0,
9              "message": {
10                  "role": "assistant",
11                  "content": "The image shows a wooden boardwalk path running through a lush green forest. The path is made of light-colored wood planks and leads into the distance, surrounded by tall trees and dense foliage. The lighting suggests it's daytime, with sunlight filtering through the leaves. There are no people or animals visible on the path.", "refusal": null,
13                  "annotations": []
14              },
15              "logprobs": null,
16              "finish_reason": "stop"
17          }
18      ],
19      "usage": {
20          "prompt_tokens": 1117,
21          "completion_tokens": 46,
22          "total_tokens": 1163,
23          "prompt_tokens_details": {
24              "cached_tokens": 0,
25              "audio_tokens": 0
26          },
27          "completion_tokens_details": {
28          }
29      }
30  }
```

```
28     "reasoning_tokens": 0,  
29     "audio_tokens": 0,  
30     "accepted_prediction_tokens": 0,  
31     "rejected_prediction_tokens": 0  
32   }  
33 },  
34 "service_tier": "default",  
35 "system_fingerprint": "fp_fc9f1d7035"  
36 }
```

The chat completion list object

An object representing a list of Chat Completions.

data array

An array of chat completion objects.

[› Show properties](#)

first_id string

The identifier of the first chat completion in the data array.

has_more boolean

Indicates whether there are more Chat Completions available.

last_id string

The identifier of the last chat completion in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion list object



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "object": "chat.completion",
6              "id": "chatcmp1-AyPNinnUqUDYo9SAdA52NobMflmj2",
7              "model": "gpt-4o-2024-08-06",
8              "created": 1738960610,
9              "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10             "tool_choice": null,
11             "usage": {
12                 "total_tokens": 31,
13                 "completion_tokens": 18,
14                 "prompt_tokens": 13
15             },
16             "seed": 4944116822809979520,
```

```
17     "top_p": 1.0,
18     "temperature": 1.0,
19     "presence_penalty": 0.0,
20     "frequency_penalty": 0.0,
21     "system_fingerprint": "fp_50cad350e4",
22     "input_user": null,
23     "service_tier": "default",
24     "tools": null,
25     "metadata": {},
26     "choices": [
27       {
28         "index": 0,
29         "message": {
30           "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31           "role": "assistant",
32           "tool_calls": null,
33           "function_call": null
34         },
35         "finish_reason": "stop",
36         "logprobs": null
37       }
38     ],
39     "response_format": null
40   }
41 ],
42   "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
43   "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
44
45
```

```
"has_more": false  
}
```

The chat completion message list object

An object representing a list of chat completion messages.

data array

An array of chat completion message objects.

› Show properties

first_id string

The identifier of the first chat message in the data array.

has_more boolean

Indicates whether there are more chat messages available.

last_id string

The identifier of the last chat message in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion message list object



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6              "role": "user",
7              "content": "write a haiku about ai",
8              "name": null,
9              "content_parts": null
10         }
11     ],
12     "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
13     "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
14     "has_more": false
15 }
```

PREVIOUS

< **Server events**

NEXT

Streaming >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Chat Completions

The Chat Completions API endpoint will generate a model response from a list of messages comprising a conversation.

Related guides:

- [Quickstart](#)
- [Text inputs and outputs](#)
- [Image inputs](#)
- [Audio inputs and outputs](#)
- [Structured Outputs](#)
- [Function calling](#)
- [Conversation state](#)

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Create chat completion

POST <https://api.openai.com/v1/chat/completions>

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Creates a model response for the given chat conversation. Learn more in the [text generation](#), [vision](#), and [audio](#) guides.

Parameter support can differ depending on the model used to generate the response, particularly for newer reasoning models. Parameters that are only supported for reasoning models are noted below. For the current state of unsupported parameters in reasoning models, refer to the [reasoning guide](#).

Request body

messages array Required

A list of messages comprising the conversation so far. Depending on the [model](#) you use, different message types (modalities) are supported, like [text](#), [images](#), and [audio](#).

› Show possible types

model string Required

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

audio object or null Optional

Parameters for audio output. Required when audio output is requested with `modalities: ["audio"]`.
[Learn more.](#)

› Show properties

frequency_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

function_call Deprecated string or object Optional

Deprecated in favor of `tool_choice`.

Controls which (if any) function is called by the model.

`none` means the model will not call a function and instead generates a message.

`auto` means the model can pick between generating a message or calling a function.

Specifying a particular function via `{"name": "my_function"}` forces the model to call that function.

`none` is the default when no functions are present. `auto` is the default if functions are present.

› Show possible types

functions Deprecated array Optional

Deprecated in favor of `tools`.

A list of functions the model may generate JSON inputs for.

› Show properties

logit_bias map Optional Defaults to null

Modify the likelihood of specified tokens appearing in the completion.

Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

logprobs boolean or null Optional Defaults to false

Whether to return log probabilities of the output tokens or not. If true, returns the log probabilities of each output token returned in the `content` of `message`.

max_completion_tokens integer or null Optional

An upper bound for the number of tokens that can be generated for a completion, including visible output tokens and reasoning tokens.

max_tokens Deprecated integer or null Optional

The maximum number of tokens that can be generated in the chat completion. This value can be used to control costs for text generated via API.

This value is now deprecated in favor of `max_completion_tokens`, and is not compatible with o-series models.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

modalities array Optional

Output types that you would like the model to generate. Most models are capable of generating text, which is the default:

`["text"]`

The `gpt-4o-audio-preview` model can also be used to generate audio. To request that this model generate both text and audio responses, you can use:

`["text", "audio"]`

n integer or null Optional Defaults to 1

How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep `n` as `1` to minimize costs.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable parallel function calling during tool use.

prediction object Optional

Configuration for a Predicted Output, which can greatly improve response times when large parts of the model response are known ahead of time. This is most common when you are regenerating a file with only minor changes to most of the content.

> Show possible types

presence_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. Learn more.

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format object Optional

An object specifying the format that the model must output.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables the older JSON mode, which ensures the message the model generates is valid JSON. Using `json_schema` is preferred for models that support it.

› Show possible types

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

seed Deprecated integer or null Optional

This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same `seed` and parameters should return the same result. Determinism is not guaranteed, and you should refer to the `system_fingerprint` response parameter to monitor changes in the backend.

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

stop string / array / null Optional Defaults to null

Not supported with latest reasoning models `o3` and `o4-mini`.

Up to 4 sequences where the API will stop generating further tokens. The returned text will not contain the stop sequence.

store boolean or null Optional Defaults to false

Whether or not to store the output of this chat completion request for use in our [model distillation](#) or [evals](#) products.

Supports text and image inputs. Note: image inputs over 8MB will be dropped.

stream boolean or null Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information, along with the [streaming responses](#) guide for more information on how to handle the streaming events.

stream_options object Optional Defaults to null

Options for streaming response. Only set this when you set `stream: true`.

› Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tool and instead generates a message. `auto` means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools. Specifying a particular tool via `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

`none` is the default when no tools are present. `auto` is the default if tools are present.

› Show possible types

tools array Optional

A list of tools the model may call. You can provide either [custom tools](#) or [function tools](#).

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more.](#)

verbosity string Optional Defaults to medium

Constrains the verbosity of the model's response. Lower values will result in more concise responses, while higher values will result in more verbose responses. Currently supported values are `low`, `medium`, and `high`.

web_search_options object Optional

This tool searches the web for relevant results to use in a response. Learn more about the [web search tool](#).

› Show properties

Returns

Returns a [chat completion](#) object, or a streamed sequence of [chat completion chunk](#) objects if the request is streamed.

Default **Image input** Streaming Functions Logprobs

Example request

python ▾ 

```
1 from openai import OpenAI  
2
```

```
3 client = OpenAI()
4
5 response = client.chat.completions.create(
6     model="gpt-4.1",
7     messages=[
8         {
9             "role": "user",
10            "content": [
11                {"type": "text", "text": "What's in this image?"},
12                {
13                    "type": "image_url",
14                    "image_url": {
15                        "url": "https://upload.wikimedia.org/wikipedia/commons/thumb/0/0d/Pythagorean_triple_3-4-5_v2.svg/1200px-Pythagorean_triple_3-4-5_v2.svg.png"
16                    }
17                },
18            ],
19        }
20    ],
21    max_tokens=300,
22 )
23
24 print(response.choices[0])
```

Response



```
1 {
2     "id": "chatcmpl-B9MHDbs1fkBeAs8l4bebGdF0J6PeG",
3     "object": "chat.completion",
```

```
4     "created": 1741570283,
5     "model": "gpt-4.1-2025-04-14",
6     "choices": [
7         {
8             "index": 0,
9             "message": {
10                 "role": "assistant",
11                 "content": "The image shows a wooden boardwalk path running through a lush green forest. The path is made of light-colored wood planks and is surrounded by dense trees and foliage. Sunlight filters through the leaves, creating dappled shadows on the ground. The overall atmosphere is peaceful and natural.", "refusal": null,
13                 "annotations": []
14             },
15             "logprobs": null,
16             "finish_reason": "stop"
17         }
18     ],
19     "usage": {
20         "prompt_tokens": 1117,
21         "completion_tokens": 46,
22         "total_tokens": 1163,
23         "prompt_tokens_details": {
24             "cached_tokens": 0,
25             "audio_tokens": 0
26         },
27         "completion_tokens_details": {
28             "reasoning_tokens": 0,
29             "audio_tokens": 0,
30             "accepted_prediction_tokens": 0,
31             "rejected_prediction_tokens": 0
32     }
```

```
33  },
34  "service_tier": "default"
35 }
```

Get chat completion

```
GET https://api.openai.com/v1/chat/completions/{completion_id}
```

Get a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 print(first_completion)
```

Response

🔗

```
1 {
2     "object": "chat.completion",
3     "id": "chatcmpl-abc123",
4     "model": "gpt-4o-2024-08-06",
5     "created": 1738960610,
6     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7     "tool_choice": null,
8     "usage": {
9         "total_tokens": 31,
10        "completion_tokens": 18,
11        "prompt_tokens": 13
12    },
```

```
13 "seed": 4944116822809979520,
14 "top_p": 1.0,
15 "temperature": 1.0,
16 "presence_penalty": 0.0,
17 "frequency_penalty": 0.0,
18 "system_fingerprint": "fp_50cad350e4",
19 "input_user": null,
20 "service_tier": "default",
21 "tools": null,
22 "metadata": {},
23 "choices": [
24   {
25     "index": 0,
26     "message": {
27       "content": "Mind of circuits hum, \nLearning patterns in silence- \nFuture",
28       "role": "assistant",
29       "tool_calls": null,
30       "function_call": null
31     },
32     "finish_reason": "stop",
33     "logprobs": null
34   }
35 ],
36 "response_format": null
37 }
```

Get chat messages

```
GET https://api.openai.com/v1/chat/completions/{completion_id}/messages
```

Get the messages in a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve messages from.

Query parameters

after string Optional

Identifier for the last message from the previous pagination request.

limit integer Optional Defaults to 20

Number of messages to retrieve.

order string Optional Defaults to asc

Sort order for messages by timestamp. Use `asc` for ascending order or `desc` for descending order.

Defaults to `asc`.

Returns

A list of messages for the specified chat completion.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 messages = client.chat.completions.messages.list(completion_id=first_id)
8 print(messages)
```

Response

🔗

```
1 {
2     "object": "list",
3     "data": [
4         {
5             "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
```

```
6     "role": "user",
7     "content": "write a haiku about ai",
8     "name": null,
9     "content_parts": null
10    }
11  ],
12  "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",
13  "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",
14  "has_more": false
15 }
```

List Chat Completions

GET <https://api.openai.com/v1/chat/completions>

List stored Chat Completions. Only Chat Completions that have been stored with the `store` parameter set to `true` will be returned.

Query parameters

after string Optional

Identifier for the last chat completion from the previous pagination request.

limit integer Optional Defaults to 20

Number of Chat Completions to retrieve.

metadata object or null Optional

A list of metadata keys to filter the Chat Completions by. Example:

```
metadata[key1]=value1&metadata[key2]=value2
```

model string Optional

The model used to generate the Chat Completions.

order string Optional Defaults to asc

Sort order for Chat Completions by timestamp. Use `asc` for ascending order or `desc` for descending order. Defaults to `asc`.

Returns

A list of [Chat Completions](#) matching the specified filters.

Example request

python 

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 print(completions)
```

Response



```
1 {
2     "object": "list",
3     "data": [
4         {
5             "object": "chat.completion",
6             "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
7             "model": "gpt-4.1-2025-04-14",
8             "created": 1738960610,
9             "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10            "tool_choice": null,
11            "usage": {
12                "total_tokens": 31,
13                "completion_tokens": 18,
14                "prompt_tokens": 13
15            },
16            "seed": 4944116822809979520,
17            "top_p": 1.0,
18            "temperature": 1.0,
19            "presence_penalty": 0.0,
20            "frequency_penalty": 0.0,
```

```
21     "system_fingerprint": "fp_50cad350e4",
22     "input_user": null,
23     "service_tier": "default",
24     "tools": null,
25     "metadata": {},
26     "choices": [
27       {
28         "index": 0,
29         "message": {
30           "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31           "role": "assistant",
32           "tool_calls": null,
33           "function_call": null
34         },
35         "finish_reason": "stop",
36         "logprobs": null
37       }
38     ],
39     "response_format": null
40   }
41 ],
42   "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
43   "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
44   "has_more": false
45 }
```

Update chat completion

```
POST https://api.openai.com/v1/chat/completions/{completion_id}
```

Modify a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be modified. Currently, the only supported modification is to update the `metadata` field.

Path parameters

completion_id string Required

The ID of the chat completion to update.

Request body

metadata map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 updated_completion = client.chat.completions.update(completion_id=first_id, request_t
7 print(updated_completion)
```

Response

🔗

```
1 {
2     "object": "chat.completion",
3     "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
4     "model": "gpt-4o-2024-08-06",
5     "created": 1738960610,
6     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7     "tool_choice": null,
8     "usage": {
9         "total_tokens": 31,
10        "completion_tokens": 18,
```

```
11     "prompt_tokens": 13
12 },
13 "seed": 4944116822809979520,
14 "top_p": 1.0,
15 "temperature": 1.0,
16 "presence_penalty": 0.0,
17 "frequency_penalty": 0.0,
18 "system_fingerprint": "fp_50cad350e4",
19 "input_user": null,
20 "service_tier": "default",
21 "tools": null,
22 "metadata": {
23     "foo": "bar"
24 },
25 "choices": [
26     {
27         "index": 0,
28         "message": {
29             "content": "Mind of circuits hum, \nLearning patterns in silence- \nFuture",
30             "role": "assistant",
31             "tool_calls": null,
32             "function_call": null
33         },
34         "finish_reason": "stop",
35         "logprobs": null
36     }
37 ],
38
39
```

```
"response_format": null  
}
```

Delete chat completion

```
DELETE https://api.openai.com/v1/chat/completions/{completion_id}
```

Delete a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be deleted.

Path parameters

`completion_id` string Required

The ID of the chat completion to delete.

Returns

A deletion confirmation object.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 delete_response = client.chat.completions.delete(completion_id=first_id)
7 print(delete_response)
```

Response



```
1 {
2   "object": "chat.completion.deleted",
3   "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
4   "deleted": true
5 }
```

The chat completion object

Represents a chat completion response returned by model, based on the provided input.

choices array

A list of chat completion choices. Can be more than one if `n` is greater than 1.

> Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created.

id string

A unique identifier for the chat completion.

model string

The model used for the chat completion.

object string

The object type, which is always `chat.completion`.

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to 'flex' or 'priority', then the request will be processed with the corresponding service tier.

- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

system_fingerprint Deprecated string

This fingerprint represents the backend configuration that the model runs with.

Can be used in conjunction with the `seed` request parameter to understand when backend changes have been made that might impact determinism.

usage object

Usage statistics for the completion request.

› Show properties

OBJECT The chat completion object

```
1  {
2    "id": "chatcmpl-B9MHDbs1fkBeAs8l4bebGdFOJ6PeG",
3    "object": "chat.completion",
4    "created": 1741570283,
5    "model": "gpt-4o-2024-08-06",
6    "choices": [
7      {
8        "index": 0,
9        "message": {
```

```
10     "role": "assistant",
11     "content": "The image shows a wooden boardwalk path running through a lush green forest. The path is made of light-colored wood planks and leads into the distance, surrounded by tall trees and dense foliage. The lighting suggests it's daytime, with sunlight filtering through the leaves. The overall atmosphere is peaceful and natural.",  
12     "refusal": null,  
13     "annotations": []  
14   },  
15   "logprobs": null,  
16   "finish_reason": "stop"  
17 }  
18 ],  
19 "usage": {  
20   "prompt_tokens": 1117,  
21   "completion_tokens": 46,  
22   "total_tokens": 1163,  
23   "prompt_tokens_details": {  
24     "cached_tokens": 0,  
25     "audio_tokens": 0  
26   },  
27   "completion_tokens_details": {  
28     "reasoning_tokens": 0,  
29     "audio_tokens": 0,  
30     "accepted_prediction_tokens": 0,  
31     "rejected_prediction_tokens": 0  
32   }  
33 },  
34   "service_tier": "default",  
35   "system_fingerprint": "fp_fc9f1d7035"  
36 }
```

The chat completion list object

An object representing a list of Chat Completions.

data array

An array of chat completion objects.

› Show properties

first_id string

The identifier of the first chat completion in the data array.

has_more boolean

Indicates whether there are more Chat Completions available.

last_id string

The identifier of the last chat completion in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion list object



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "object": "chat.completion",
6              "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
7              "model": "gpt-4o-2024-08-06",
8              "created": 1738960610,
9              "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10             "tool_choice": null,
11             "usage": {
12                 "total_tokens": 31,
13                 "completion_tokens": 18,
14                 "prompt_tokens": 13
15             },
16             "seed": 4944116822809979520,
17             "top_p": 1.0,
18             "temperature": 1.0,
19             "presence_penalty": 0.0,
20             "frequency_penalty": 0.0,
21             "system_fingerprint": "fp_50cad350e4",
22             "input_user": null,
23             "service_tier": "default",
24             "tools": null,
25             "metadata": {},
26             "choices": [
27                 {
```

```
28     "index": 0,
29     "message": {
30         "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31         "role": "assistant",
32         "tool_calls": null,
33         "function_call": null
34     },
35     "finish_reason": "stop",
36     "logprobs": null
37 }
38 ],
39 "response_format": null
40 }
41 ],
42 "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
43 "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
44 "has_more": false
45 }
```

The chat completion message list object

An object representing a list of chat completion messages.

data array

An array of chat completion message objects.

> Show properties

first_id string

The identifier of the first chat message in the data array.

has_more boolean

Indicates whether there are more chat messages available.

last_id string

The identifier of the last chat message in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion message list object



```
1  {
2    "object": "list",
3    "data": [
4      {
5        "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6        "role": "user",
7        "content": "write a haiku about ai",
```

```
8     "name": null,  
9     "content_parts": null  
10    }  
11  ],  
12  "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",  
13  "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",  
14  "has_more": false  
15 }
```

PREVIOUS
< **Server events**

NEXT
Streaming >

Chat Completions



The Chat Completions API endpoint will generate a model response from a list of messages comprising a conversation.

Related guides:

[Quickstart](#)

[Text inputs and outputs](#)

[Image inputs](#)

[Audio inputs and outputs](#)

[Structured Outputs](#)

[Function calling](#)

[Conversation state](#)

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions with Responses](#).

Create chat completion



POST <https://api.openai.com/v1/chat/completions>

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features.

Compare [Chat Completions with Responses](#).

Creates a model response for the given chat conversation. Learn more in the [text generation](#), [vision](#), and [audio](#) guides.

Parameter support can differ depending on the model used to generate the response, particularly for newer reasoning models. Parameters that are only supported for reasoning models are noted below. For the current state of unsupported parameters in reasoning models, [refer to the reasoning guide](#).

Request body

messages array Required

A list of messages comprising the conversation so far. Depending on the [model](#) you use, different message types (modalities) are supported, like [text](#), [images](#), and [audio](#).

› Show possible types

model string Required

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

audio object or null Optional

Parameters for audio output. Required when audio output is requested with `modalities: ["audio"]`. [Learn more.](#)

› Show properties

frequency_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

function_call Deprecated string or object Optional

Deprecated in favor of `tool_choice`.

Controls which (if any) function is called by the model.

`none` means the model will not call a function and instead generates a message.

`auto` means the model can pick between generating a message or calling a function.

Specifying a particular function via `{"name": "my_function"}` forces the model to call that function.

`none` is the default when no functions are present. `auto` is the default if functions are present.

› Show possible types

functions Deprecated array Optional

Deprecated in favor of `tools`.

A list of functions the model may generate JSON inputs for.

› Show properties

logit_bias map Optional Defaults to null

Modify the likelihood of specified tokens appearing in the completion.

Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100.

Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

logprobs boolean or null Optional Defaults to false

Whether to return log probabilities of the output tokens or not. If true, returns the log probabilities of each output token returned in the `content` of `message`.

max_completion_tokens integer or null Optional

An upper bound for the number of tokens that can be generated for a completion, including visible output tokens and reasoning tokens.

max_tokens Deprecated integer or null Optional

The maximum number of tokens that can be generated in the chat completion. This value can be used to control costs for text generated via API.

This value is now deprecated in favor of `max_completion_tokens`, and is not compatible with o-series models.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

modalities array Optional

Output types that you would like the model to generate. Most models are capable of generating text, which is the default:

```
["text"]
```

The `gpt-4o-audio-preview` model can also be used to [generate audio](#). To request that this model generate both text and audio responses, you can use:

```
["text", "audio"]
```

n integer or null Optional Defaults to 1

How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep `n` as `1` to minimize costs.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

prediction object Optional

Configuration for a [Predicted Output](#), which can greatly improve response times when large parts of the model response are known ahead of time. This is most common when you are regenerating a file with only minor changes to most of the content.

> Show possible types

presence_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. [Learn more](#).

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more](#).

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

`gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in gpt-5.1.

All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.

The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.

`xhigh` is supported for all models after `gpt-5.1-codex-max`.

response_format object Optional

An object specifying the format that the model must output.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables the older JSON mode, which ensures the message the model generates is valid JSON. Using `json_schema` is preferred for models that support it.

› Show possible types

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

seed Deprecated integer or null Optional

This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same `seed` and parameters should return the same result. Determinism is not guaranteed, and you should refer to the `system_fingerprint` response parameter to monitor changes in the backend.

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

stop string / array / null Optional Defaults to null

Not supported with latest reasoning models `o3` and `o4-mini`.

Up to 4 sequences where the API will stop generating further tokens. The returned text will not contain the stop sequence.

store boolean or null Optional Defaults to false

Whether or not to store the output of this chat completion request for use in our [model distillation](#) or [evals](#) products.

Supports text and image inputs. Note: image inputs over 8MB will be dropped.

stream boolean or null Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information, along with the [streaming responses](#) guide for more information on how to handle the streaming events.

stream_options object Optional Defaults to null

Options for streaming response. Only set this when you set `stream: true`.

> Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tool and instead generates a message.

`auto` means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools. Specifying a particular tool via `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

`none` is the default when no tools are present. `auto` is the default if tools are present.

› Show possible types

tools array Optional

A list of tools the model may call. You can provide either custom tools or function tools.

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more](#).

verbosity string Optional Defaults to medium

Constrains the verbosity of the model's response. Lower values will result in more concise responses, while higher values will result in more verbose responses. Currently supported values are `low`, `medium`, and `high`.

web_search_options object Optional

This tool searches the web for relevant results to use in a response. Learn more about the [web search tool](#).

› Show properties

Returns

Returns a [chat completion](#) object, or a streamed sequence of [chat completion chunk](#) objects if the request is streamed.

Default Image input **Streaming** Functions Logprobs

Example request

```
from openai import OpenAI
client = OpenAI()

completion = client.chat.completions.create(
    model="gpt-5.2",
    messages=[
        {"role": "developer", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Hello!"}
    ],
    stream=True
)

for chunk in completion:
    print(chunk.choices[0].delta)
```

Response

```
{"id": "chatcmpl-123", "object": "chat.completion.chunk", "created": 1694268190, "model": "gpt-4o-mini",  
 {"id": "chatcmpl-123", "object": "chat.completion.chunk", "created": 1694268190, "model": "gpt-4o-mini",  
 ...  
 {"id": "chatcmpl-123", "object": "chat.completion.chunk", "created": 1694268190, "model": "gpt-4o-mini",
```

Get chat completion



```
GET https://api.openai.com/v1/chat/completions/{completion_id}
```

Get a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

`completion_id` string Required

The ID of the chat completion to retrieve.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

completions = client.chat.completions.list()
first_id = completions[0].id
first_completion = client.chat.completions.retrieve(completion_id=first_id)
print(first_completion)
```

Response

```
{
  "object": "chat.completion",
  "id": "chatcmpl-abc123",
  "model": "gpt-4o-2024-08-06",
  "created": 1738960610,
  "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
  "tool_choice": null,
  "usage": {
```

```
"total_tokens": 31,  
"completion_tokens": 18,  
"prompt_tokens": 13  
},  
"seed": 4944116822809979520,  
"top_p": 1.0,  
"temperature": 1.0,  
"presence_penalty": 0.0,  
"frequency_penalty": 0.0,  
"system_fingerprint": "fp_50cad350e4",  
"input_user": null,  
"service_tier": "default",  
"tools": null,  
"metadata": {},  
"choices": [  
  {  
    "index": 0,  
    "message": {  
      "content": "Mind of circuits hum, \nLearning patterns in silence- \nFuture's quiet spark",  
      "role": "assistant",  
      "tool_calls": null,  
      "function_call": null  
    },  
    "finish_reason": "stop",  
    "logprobs": null  
  }  
,  
  {"  
    "response_format": null  
  }  
]
```

Get chat messages



```
GET https://api.openai.com/v1/chat/completions/{completion_id}/messages
```

Get the messages in a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve messages from.

Query parameters

after string Optional

Identifier for the last message from the previous pagination request.

limit integer Optional Defaults to 20

Number of messages to retrieve.

order string Optional Defaults to asc

Sort order for messages by timestamp. Use `asc` for ascending order or `desc` for descending order. Defaults to `asc`.

Returns

A list of messages for the specified chat completion.

Example request

```
from openai import OpenAI
client = OpenAI()

completions = client.chat.completions.list()
first_id = completions[0].id
first_completion = client.chat.completions.retrieve(completion_id=first_id)
messages = client.chat.completions.messages.list(completion_id=first_id)
print(messages)
```

Response

```
{
  "object": "list",
  "data": [
    {
```

```
"id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
"role": "user",
"content": "write a haiku about ai",
"name": null,
"content_parts": null
},
],
"first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
"last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
"has_more": false
}
```

List Chat Completions



GET <https://api.openai.com/v1/chat/completions>

List stored Chat Completions. Only Chat Completions that have been stored with the `store` parameter set to `true` will be returned.

Query parameters

after string Optional

Identifier for the last chat completion from the previous pagination request.

limit integer Optional Defaults to 20

Number of Chat Completions to retrieve.

metadata object or null Optional

A list of metadata keys to filter the Chat Completions by. Example:

```
metadata[key1]=value1&metadata[key2]=value2
```

model string Optional

The model used to generate the Chat Completions.

order string Optional Defaults to asc

Sort order for Chat Completions by timestamp. Use `asc` for ascending order or `desc` for descending order. Defaults to `asc`.

Returns

A list of [Chat Completions](#) matching the specified filters.

Example request

```
from openai import OpenAI
client = OpenAI()
```

```
completions = client.chat.completions.list()  
print(completions)
```

Response

```
{  
  "object": "list",  
  "data": [  
    {  
      "object": "chat.completion",  
      "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
      "model": "gpt-4.1-2025-04-14",  
      "created": 1738960610,  
      "request_id": "req_ded8ab984ec4bf840f37566c1011c417",  
      "tool_choice": null,  
      "usage": {  
        "total_tokens": 31,  
        "completion_tokens": 18,  
        "prompt_tokens": 13  
      },  
      "seed": 4944116822809979520,  
      "top_p": 1.0,  
      "temperature": 1.0,  
      "presence_penalty": 0.0,  
      "frequency_penalty": 0.0,  
      "system_fingerprint": "fp_50cad350e4",  
      "input_user": null,  
      "service_tier": "default",  
      "tools": null,  
      "content": "Hello, how can I assist you today?"  
    }  
  ]  
}
```

```
"metadata": {},  
"choices": [  
    {  
        "index": 0,  
        "message": {  
            "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture's quiet s  
            "role": "assistant",  
            "tool_calls": null,  
            "function_call": null  
        },  
        "finish_reason": "stop",  
        "logprobs": null  
    },  
    ],  
    "response_format": null  
},  
],  
"first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
"last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
"has_more": false  
}
```

Update chat completion



```
POST https://api.openai.com/v1/chat/completions/{completion_id}
```

Modify a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be modified. Currently, the only supported modification is to update the `metadata` field.

Path parameters

completion_id string Required

The ID of the chat completion to update.

Request body

metadata map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

completions = client.chat.completions.list()
first_id = completions[0].id
updated_completion = client.chat.completions.update(completion_id=first_id, request_body={"metadata": {"key": "value"}, "tool_choice": "OPTIONAL"})
print(updated_completion)
```

Response

```
{
  "object": "chat.completion",
  "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
  "model": "gpt-4o-2024-08-06",
  "created": 1738960610,
  "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
  "tool_choice": null,
  "usage": {
    "total_tokens": 31,
    "completion_tokens": 18,
    "prompt_tokens": 13
  },
}
```

```
"seed": 4944116822809979520,  
"top_p": 1.0,  
"temperature": 1.0,  
"presence_penalty": 0.0,  
"frequency_penalty": 0.0,  
"system_fingerprint": "fp_50cad350e4",  
"input_user": null,  
"service_tier": "default",  
"tools": null,  
"metadata": {  
    "foo": "bar"  
},  
"choices": [  
    {  
        "index": 0,  
        "message": {  
            "content": "Mind of circuits hum, \nLearning patterns in silence- \nFuture's quiet spa  
            "role": "assistant",  
            "tool_calls": null,  
            "function_call": null  
        },  
        "finish_reason": "stop",  
        "logprobs": null  
    }  
],  
"response_format": null  
}
```

Delete chat completion



```
DELETE https://api.openai.com/v1/chat/completions/{completion_id}
```

Delete a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be deleted.

Path parameters

completion_id string Required

The ID of the chat completion to delete.

Returns

A deletion confirmation object.

Example request

```
from openai import OpenAI
client = OpenAI()

completions = client.chat.completions.list()
first_id = completions[0].id
```

```
delete_response = client.chat.completions.delete(completion_id=first_id)
print(delete_response)
```

Response

```
{
  "object": "chat.completion.deleted",
  "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
  "deleted": true
}
```

The chat completion object



Represents a chat completion response returned by model, based on the provided input.

choices array

A list of chat completion choices. Can be more than one if `n` is greater than 1.

> Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created.

id string

A unique identifier for the chat completion.

model string

The model used for the chat completion.

object string

The object type, which is always `chat.completion`.

service_tier string

Specifies the processing type used for serving the request.

If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.

If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.

If set to 'flex' or 'priority', then the request will be processed with the corresponding service tier.

When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

system_fingerprint Deprecated string

This fingerprint represents the backend configuration that the model runs with.

Can be used in conjunction with the `seed` request parameter to understand when backend changes have been made that might impact determinism.

usage object

Usage statistics for the completion request.

➤ Show properties

OBJECT The chat completion object

```
{  
  "id": "chatcmpl-B9MHDbslfkBeAs8l4bebGdFOJ6PeG",  
  "object": "chat.completion",  
  "created": 1741570283,  
  "model": "gpt-4o-2024-08-06",  
  "choices": [  
    {  
      "index": 0,  
      "message": {  
        "role": "assistant",  
        "content": "The image shows a wooden boardwalk path running through a lush green field or  
        "refusal": null,  
        "annotations": []  
      },  
      "logprobs": null,  
      "finish_reason": "stop"  
    },  
  ],  
  "usage": {  
    "prompt_tokens": 1117,  
    "completion_tokens": 1117,  
    "total_tokens": 2234  
  }  
}
```

```
"completion_tokens": 46,  
"total_tokens": 1163,  
"prompt_tokens_details": {  
    "cached_tokens": 0,  
    "audio_tokens": 0  
},  
"completion_tokens_details": {  
    "reasoning_tokens": 0,  
    "audio_tokens": 0,  
    "accepted_prediction_tokens": 0,  
    "rejected_prediction_tokens": 0  
}  
},  
"service_tier": "default",  
"system_fingerprint": "fp_fc9f1d7035"  
}
```

The chat completion list object



An object representing a list of Chat Completions.

data array

An array of chat completion objects.

› Show properties

first_id string

The identifier of the first chat completion in the data array.

has_more boolean

Indicates whether there are more Chat Completions available.

last_id string

The identifier of the last chat completion in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion list object

```
{  
  "object": "list",  
  "data": [  
    {  
      "object": "chat.completion",  
      "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
      "model": "gpt-4o-2024-08-06",  
      "created": 1738960610,  
      "request_id": "req_ded8ab984ec4bf840f37566c1011c417",  
      "tool_choice": null,  
    }]
```

```
"usage": {  
    "total_tokens": 31,  
    "completion_tokens": 18,  
    "prompt_tokens": 13  
},  
"seed": 4944116822809979520,  
"top_p": 1.0,  
"temperature": 1.0,  
"presence_penalty": 0.0,  
"frequency_penalty": 0.0,  
"system_fingerprint": "fp_50cad350e4",  
"input_user": null,  
"service_tier": "default",  
"tools": null,  
"metadata": {},  
"choices": [  
    {  
        "index": 0,  
        "message": {  
            "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture's quiet s  
            "role": "assistant",  
            "tool_calls": null,  
            "function_call": null  
        },  
        "finish_reason": "stop",  
        "logprobs": null  
    }  
],  
"response_format": null
```

```
    },
    ],
    "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
    "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
    "has_more": false
}
```

The chat completion message list object



An object representing a list of chat completion messages.

data array

An array of chat completion message objects.

› Show properties

first_id string

The identifier of the first chat message in the data array.

has_more boolean

Indicates whether there are more chat messages available.

last_id string

The identifier of the last chat message in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion message list object

```
{  
  "object": "list",  
  "data": [  
    {  
      "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",  
      "role": "user",  
      "content": "write a haiku about ai",  
      "name": null,  
      "content_parts": null  
    }  
,  
  "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",  
  "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",  
  "has_more": false  
}
```

PREVIOUS



NEXT

Server events

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Chat Completions

The Chat Completions API endpoint will generate a model response from a list of messages comprising a conversation.

Related guides:

- [Quickstart](#)
- [Text inputs and outputs](#)
- [Image inputs](#)
- [Audio inputs and outputs](#)
- [Structured Outputs](#)
- [Function calling](#)
- [Conversation state](#)

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Create chat completion

POST <https://api.openai.com/v1/chat/completions>

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Creates a model response for the given chat conversation. Learn more in the [text generation](#), [vision](#), and [audio](#) guides.

Parameter support can differ depending on the model used to generate the response, particularly for newer reasoning models. Parameters that are only supported for reasoning models are noted below. For the current state of unsupported parameters in reasoning models, refer to the [reasoning guide](#).

Request body

messages array Required

A list of messages comprising the conversation so far. Depending on the [model](#) you use, different message types (modalities) are supported, like [text](#), [images](#), and [audio](#).

› Show possible types

model string Required

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

audio object or null Optional

Parameters for audio output. Required when audio output is requested with `modalities: ["audio"]`.
[Learn more.](#)

› Show properties

frequency_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

function_call Deprecated string or object Optional

Deprecated in favor of `tool_choice`.

Controls which (if any) function is called by the model.

`none` means the model will not call a function and instead generates a message.

`auto` means the model can pick between generating a message or calling a function.

Specifying a particular function via `{"name": "my_function"}` forces the model to call that function.

`none` is the default when no functions are present. `auto` is the default if functions are present.

› Show possible types

functions Deprecated array Optional

Deprecated in favor of `tools`.

A list of functions the model may generate JSON inputs for.

› Show properties

logit_bias map Optional Defaults to null

Modify the likelihood of specified tokens appearing in the completion.

Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

logprobs boolean or null Optional Defaults to false

Whether to return log probabilities of the output tokens or not. If true, returns the log probabilities of each output token returned in the `content` of `message`.

max_completion_tokens integer or null Optional

An upper bound for the number of tokens that can be generated for a completion, including visible output tokens and reasoning tokens.

max_tokens Deprecated integer or null Optional

The maximum number of tokens that can be generated in the chat completion. This value can be used to control costs for text generated via API.

This value is now deprecated in favor of `max_completion_tokens`, and is not compatible with o-series models.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

modalities array Optional

Output types that you would like the model to generate. Most models are capable of generating text, which is the default:

`["text"]`

The `gpt-4o-audio-preview` model can also be used to generate audio. To request that this model generate both text and audio responses, you can use:

`["text", "audio"]`

n integer or null Optional Defaults to 1

How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep `n` as `1` to minimize costs.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable parallel function calling during tool use.

prediction object Optional

Configuration for a Predicted Output, which can greatly improve response times when large parts of the model response are known ahead of time. This is most common when you are regenerating a file with only minor changes to most of the content.

> Show possible types

presence_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. Learn more.

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format object Optional

An object specifying the format that the model must output.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables the older JSON mode, which ensures the message the model generates is valid JSON. Using `json_schema` is preferred for models that support it.

› Show possible types

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

seed Deprecated integer or null Optional

This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same `seed` and parameters should return the same result. Determinism is not guaranteed, and you should refer to the `system_fingerprint` response parameter to monitor changes in the backend.

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

stop string / array / null Optional Defaults to null

Not supported with latest reasoning models `o3` and `o4-mini`.

Up to 4 sequences where the API will stop generating further tokens. The returned text will not contain the stop sequence.

store boolean or null Optional Defaults to false

Whether or not to store the output of this chat completion request for use in our [model distillation](#) or [evals](#) products.

Supports text and image inputs. Note: image inputs over 8MB will be dropped.

stream boolean or null Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information, along with the [streaming responses](#) guide for more information on how to handle the streaming events.

stream_options object Optional Defaults to null

Options for streaming response. Only set this when you set `stream: true`.

› Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tool and instead generates a message. `auto` means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools. Specifying a particular tool via `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

`none` is the default when no tools are present. `auto` is the default if tools are present.

› Show possible types

tools array Optional

A list of tools the model may call. You can provide either [custom tools](#) or [function tools](#).

› Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more.](#)

verbosity string Optional Defaults to medium

Constrains the verbosity of the model's response. Lower values will result in more concise responses, while higher values will result in more verbose responses. Currently supported values are `low`, `medium`, and `high`.

web_search_options object Optional

This tool searches the web for relevant results to use in a response. Learn more about the [web search tool](#).

› Show properties

Returns

Returns a [chat completion](#) object, or a streamed sequence of [chat completion chunk](#) objects if the request is streamed.

Default Image input Streaming **Functions** Logprobs

Example request

python ⚙️

```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3
4 tools = [
5   {
6     "type": "function",
7     "function": {
8       "name": "get_current_weather",
9       "description": "Get the current weather in a given location",
10      "parameters": {
11        "type": "object",
12        "properties": {
13          "location": {
14            "type": "string",
15            "description": "The city and state, e.g. San Francisco, CA",
16          },
17          "unit": {"type": "string", "enum": ["celsius", "fahrenheit"]},
18        },
19        "required": ["location"],
20      },
21    }
22  ]
23 ]
24 messages = [{"role": "user", "content": "What's the weather like in Boston today?"}]
25 completion = client.chat.completions.create(
26   model="gpt-5.2",
27   messages=messages,
28   tools=tools,
29   tool_choice="auto"
30 )
```

```
31  
32 print(completion)
```

Response



```
1  {  
2      "id": "chatcmpl-abc123",  
3      "object": "chat.completion",  
4      "created": 1699896916,  
5      "model": "gpt-4o-mini",  
6      "choices": [  
7          {  
8              "index": 0,  
9              "message": {  
10                  "role": "assistant",  
11                  "content": null,  
12                  "tool_calls": [  
13                      {  
14                          "id": "call_abc123",  
15                          "type": "function",  
16                          "function": {  
17                              "name": "get_current_weather",  
18                              "arguments": "{\n\"location\": \"Boston, MA\"\n}"  
19                          }  
20                      }  
21                  ]  
22          },  
23          {"logprobs": null,
```

```
24     "finish_reason": "tool_calls"
25   }
26 ],
27 "usage": {
28   "prompt_tokens": 82,
29   "completion_tokens": 17,
30   "total_tokens": 99,
31   "completion_tokens_details": {
32     "reasoning_tokens": 0,
33     "accepted_prediction_tokens": 0,
34     "rejected_prediction_tokens": 0
35   }
36 }
37 }
```

Get chat completion

```
GET https://api.openai.com/v1/chat/completions/{completion_id}
```

Get a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 print(first_completion)
```

Response

Copy

```
1 {
2     "object": "chat.completion",
3     "id": "chatcmpl-abc123",
4     "model": "gpt-4o-2024-08-06",
```

```
5     "created": 1738960610,
6     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7     "tool_choice": null,
8     "usage": {
9         "total_tokens": 31,
10        "completion_tokens": 18,
11        "prompt_tokens": 13
12    },
13    "seed": 4944116822809979520,
14    "top_p": 1.0,
15    "temperature": 1.0,
16    "presence_penalty": 0.0,
17    "frequency_penalty": 0.0,
18    "system_fingerprint": "fp_50cad350e4",
19    "input_user": null,
20    "service_tier": "default",
21    "tools": null,
22    "metadata": {},
23    "choices": [
24        {
25            "index": 0,
26            "message": {
27                "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture",
28                "role": "assistant",
29                "tool_calls": null,
30                "function_call": null
31            },
32            "finish_reason": "stop",
33            "logprobs": null

```

```
34      }
35    ],
36    "response_format": null
37 }
```

Get chat messages

```
GET https://api.openai.com/v1/chat/completions/{completion_id}/messages
```

Get the messages in a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve messages from.

Query parameters

after string Optional

Identifier for the last message from the previous pagination request.

limit integer Optional Defaults to 20

Number of messages to retrieve.

order string Optional Defaults to asc

Sort order for messages by timestamp. Use `asc` for ascending order or `desc` for descending order.
Defaults to `asc`.

Returns

A list of messages for the specified chat completion.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 messages = client.chat.completions.messages.list(completion_id=first_id)
8 print(messages)
```

Response



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6              "role": "user",
7              "content": "write a haiku about ai",
8              "name": null,
9              "content_parts": null
10         },
11     ],
12     "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
13     "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
14     "has_more": false
15 }
```

List Chat Completions

GET <https://api.openai.com/v1/chat/completions>

List stored Chat Completions. Only Chat Completions that have been stored with the `store` parameter set to `true` will be returned.

Query parameters

after string Optional

Identifier for the last chat completion from the previous pagination request.

limit integer Optional Defaults to 20

Number of Chat Completions to retrieve.

metadata object or null Optional

A list of metadata keys to filter the Chat Completions by. Example:

```
metadata[key1]=value1&metadata[key2]=value2
```

model string Optional

The model used to generate the Chat Completions.

order string Optional Defaults to asc

Sort order for Chat Completions by timestamp. Use `asc` for ascending order or `desc` for descending order. Defaults to `asc`.

Returns

A list of [Chat Completions](#) matching the specified filters.

Example request

python ↗

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 print(completions)
```

Response

↗

```
1 {
2     "object": "list",
3     "data": [
4         {
5             "object": "chat.completion",
6             "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
7             "model": "gpt-4.1-2025-04-14",
8             "created": 1738960610,
9             "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10            "tool_choice": null,
11            "usage": {
12                "total_tokens": 31,
13                "completion_tokens": 18,
14                "prompt_tokens": 13
15            }
16        }
17    ]
18}
```

```
15 },
16 "seed": 4944116822809979520,
17 "top_p": 1.0,
18 "temperature": 1.0,
19 "presence_penalty": 0.0,
20 "frequency_penalty": 0.0,
21 "system_fingerprint": "fp_50cad350e4",
22 "input_user": null,
23 "service_tier": "default",
24 "tools": null,
25 "metadata": {},
26 "choices": [
27 {
28     "index": 0,
29     "message": {
30         "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31         "role": "assistant",
32         "tool_calls": null,
33         "function_call": null
34     },
35     "finish_reason": "stop",
36     "logprobs": null
37 }
38 ],
39 "response_format": null
40 }
41 ],
42 "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
43 "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
```

```
44  "has_more": false  
45 }
```

Update chat completion

POST https://api.openai.com/v1/chat/completions/{completion_id}

Modify a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be modified. Currently, the only supported modification is to update the `metadata` field.

Path parameters

completion_id string Required

The ID of the chat completion to update.

Request body

metadata map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python 

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 updated_completion = client.chat.completions.update(completion_id=first_id, request_t
7 print(updated_completion)
```

Response



```
1 {
2     "object": "chat.completion",
3     "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
```

```
4     "model": "gpt-4o-2024-08-06",
5     "created": 1738960610,
6     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7     "tool_choice": null,
8     "usage": {
9         "total_tokens": 31,
10        "completion_tokens": 18,
11        "prompt_tokens": 13
12    },
13    "seed": 4944116822809979520,
14    "top_p": 1.0,
15    "temperature": 1.0,
16    "presence_penalty": 0.0,
17    "frequency_penalty": 0.0,
18    "system_fingerprint": "fp_50cad350e4",
19    "input_user": null,
20    "service_tier": "default",
21    "tools": null,
22    "metadata": {
23        "foo": "bar"
24    },
25    "choices": [
26        {
27            "index": 0,
28            "message": {
29                "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture",
30                "role": "assistant",
31                "tool_calls": null,
32                "function_call": null
33            }
34        }
35    ]
36 }
```

```
33     },
34     "finish_reason": "stop",
35     "logprobs": null
36   }
37 ],
38 "response_format": null
39 }
```

Delete chat completion

```
DELETE https://api.openai.com/v1/chat/completions/{completion_id}
```

Delete a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be deleted.

Path parameters

completion_id string **Required**

The ID of the chat completion to delete.

Returns

A deletion confirmation object.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 delete_response = client.chat.completions.delete(completion_id=first_id)
7 print(delete_response)
```

Response

⌚

```
1 {
2   "object": "chat.completion.deleted",
3   "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
4   "deleted": true
5 }
```

The chat completion object

Represents a chat completion response returned by model, based on the provided input.

choices array

A list of chat completion choices. Can be more than one if `n` is greater than 1.

> Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created.

id string

A unique identifier for the chat completion.

model string

The model used for the chat completion.

object string

The object type, which is always `chat.completion`.

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

`system_fingerprint` Deprecated string

This fingerprint represents the backend configuration that the model runs with.

Can be used in conjunction with the `seed` request parameter to understand when backend changes have been made that might impact determinism.

`usage` object

Usage statistics for the completion request.

› Show properties

OBJECT The chat completion object



```
1  {
2    "id": "chatcmpl-B9MHDbs1fkBeAs8l4bebGdFOJ6PeG",
3    "object": "chat.completion",
```

```
4     "created": 1741570283,
5     "model": "gpt-4o-2024-08-06",
6     "choices": [
7         {
8             "index": 0,
9             "message": {
10                 "role": "assistant",
11                 "content": "The image shows a wooden boardwalk path running through a lush green forest. The path is made of light-colored wooden planks and is surrounded by dense trees and foliage. The lighting suggests it's daytime, and the overall atmosphere is peaceful and natural.",
12                 "refusal": null,
13                 "annotations": []
14             },
15             "logprobs": null,
16             "finish_reason": "stop"
17         }
18     ],
19     "usage": {
20         "prompt_tokens": 1117,
21         "completion_tokens": 46,
22         "total_tokens": 1163,
23         "prompt_tokens_details": {
24             "cached_tokens": 0,
25             "audio_tokens": 0
26         },
27         "completion_tokens_details": {
28             "reasoning_tokens": 0,
29             "audio_tokens": 0,
30             "accepted_prediction_tokens": 0,
31             "rejected_prediction_tokens": 0
32     }
```

```
33 },
34 "service_tier": "default",
35 "system_fingerprint": "fp_fc9f1d7035"
36 }
```

The chat completion list object

An object representing a list of Chat Completions.

data array

An array of chat completion objects.

› Show properties

first_id string

The identifier of the first chat completion in the data array.

has_more boolean

Indicates whether there are more Chat Completions available.

last_id string

The identifier of the last chat completion in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion list object

```
1  {
2      "object": "list",
3      "data": [
4          {
5              "object": "chat.completion",
6              "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
7              "model": "gpt-4o-2024-08-06",
8              "created": 1738960610,
9              "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10             "tool_choice": null,
11             "usage": {
12                 "total_tokens": 31,
13                 "completion_tokens": 18,
14                 "prompt_tokens": 13
15             },
16             "seed": 4944116822809979520,
17             "top_p": 1.0,
18             "temperature": 1.0,
19             "presence_penalty": 0.0,
20             "frequency_penalty": 0.0,
21             "system_fingerprint": "fp_50cad350e4",
```



```
22     "input_user": null,  
23     "service_tier": "default",  
24     "tools": null,  
25     "metadata": {},  
26     "choices": [  
27         {  
28             "index": 0,  
29             "message": {  
30                 "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu  
31                 "role": "assistant",  
32                 "tool_calls": null,  
33                 "function_call": null  
34             },  
35             "finish_reason": "stop",  
36             "logprobs": null  
37         }  
38     ],  
39     "response_format": null  
40 },  
41 ],  
42 "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
43 "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
44 "has_more": false  
45 }
```

The chat completion message list object

An object representing a list of chat completion messages.

data array

An array of chat completion message objects.

[› Show properties](#)

first_id string

The identifier of the first chat message in the data array.

has_more boolean

Indicates whether there are more chat messages available.

last_id string

The identifier of the last chat message in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion message list object



```
1  {
2      "object": "list",
3      "data": [
4          {
5              "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6              "role": "user",
7              "content": "write a haiku about ai",
8              "name": null,
9              "content_parts": null
10         }
11     ],
12     "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
13     "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
14     "has_more": false
15 }
```

PREVIOUS
< **Server events**

NEXT
Streaming >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Chat Completions

The Chat Completions API endpoint will generate a model response from a list of messages comprising a conversation.

Related guides:

- [Quickstart](#)
- [Text inputs and outputs](#)
- [Image inputs](#)
- [Audio inputs and outputs](#)
- [Structured Outputs](#)
- [Function calling](#)
- [Conversation state](#)

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Create chat completion

POST <https://api.openai.com/v1/chat/completions>

Starting a new project? We recommend trying [Responses](#) to take advantage of the latest OpenAI platform features. Compare [Chat Completions](#) with [Responses](#).

Creates a model response for the given chat conversation. Learn more in the [text generation](#), [vision](#), and [audio](#) guides.

Parameter support can differ depending on the model used to generate the response, particularly for newer reasoning models. Parameters that are only supported for reasoning models are noted below. For the current state of unsupported parameters in reasoning models, refer to the [reasoning guide](#).

Request body

messages array Required

A list of messages comprising the conversation so far. Depending on the [model](#) you use, different message types (modalities) are supported, like [text](#), [images](#), and [audio](#).

› Show possible types

model string Required

Model ID used to generate the response, like `gpt-4o` or `o3`. OpenAI offers a wide range of models with different capabilities, performance characteristics, and price points. Refer to the [model guide](#) to browse and compare available models.

audio object or null Optional

Parameters for audio output. Required when audio output is requested with `modalities: ["audio"]`.

[Learn more.](#)

› Show properties

frequency_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

function_call Deprecated string or object Optional

Deprecated in favor of `tool_choice`.

Controls which (if any) function is called by the model.

`none` means the model will not call a function and instead generates a message.

`auto` means the model can pick between generating a message or calling a function.

Specifying a particular function via `{"name": "my_function"}` forces the model to call that function.

`none` is the default when no functions are present. `auto` is the default if functions are present.

› Show possible types

functions Deprecated array Optional

Deprecated in favor of `tools`.

A list of functions the model may generate JSON inputs for.

› Show properties

logit_bias map Optional Defaults to null

Modify the likelihood of specified tokens appearing in the completion.

Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

logprobs boolean or null Optional Defaults to false

Whether to return log probabilities of the output tokens or not. If true, returns the log probabilities of each output token returned in the `content` of `message`.

max_completion_tokens integer or null Optional

An upper bound for the number of tokens that can be generated for a completion, including visible output tokens and reasoning tokens.

max_tokens Deprecated integer or null Optional

The maximum number of tokens that can be generated in the chat completion. This value can be used to control costs for text generated via API.

This value is now deprecated in favor of `max_completion_tokens`, and is not compatible with o-series models.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

modalities array Optional

Output types that you would like the model to generate. Most models are capable of generating text, which is the default:

`["text"]`

The `gpt-4o-audio-preview` model can also be used to generate audio. To request that this model generate both text and audio responses, you can use:

`["text", "audio"]`

n integer or null Optional Defaults to 1

How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep `n` as `1` to minimize costs.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable parallel function calling during tool use.

prediction object Optional

Configuration for a Predicted Output, which can greatly improve response times when large parts of the model response are known ahead of time. This is most common when you are regenerating a file with only minor changes to most of the content.

> Show possible types

presence_penalty number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.

prompt_cache_key string Optional

Used by OpenAI to cache responses for similar requests to optimize your cache hit rates. Replaces the `user` field. Learn more.

prompt_cache_retention string Optional

The retention policy for the prompt cache. Set to `24h` to enable extended prompt caching, which keeps cached prefixes active for longer, up to a maximum of 24 hours. [Learn more.](#)

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format object Optional

An object specifying the format that the model must output.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables the older JSON mode, which ensures the message the model generates is valid JSON. Using `json_schema` is preferred for models that support it.

› Show possible types

safety_identifier string Optional

A stable identifier used to help detect users of your application that may be violating OpenAI's usage policies. The IDs should be a string that uniquely identifies each user. We recommend hashing their username or email address, in order to avoid sending us any identifying information. [Learn more.](#)

seed Deprecated integer or null Optional

This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same `seed` and parameters should return the same result. Determinism is not guaranteed, and you should refer to the `system_fingerprint` response parameter to monitor changes in the backend.

service_tier string Optional Defaults to auto

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to '[flex](#)' or '[priority](#)', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

stop string / array / null Optional Defaults to null

Not supported with latest reasoning models `o3` and `o4-mini`.

Up to 4 sequences where the API will stop generating further tokens. The returned text will not contain the stop sequence.

store boolean or null Optional Defaults to false

Whether or not to store the output of this chat completion request for use in our [model distillation](#) or [evals](#) products.

Supports text and image inputs. Note: image inputs over 8MB will be dropped.

stream boolean or null Optional Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using [server-sent events](#). See the [Streaming section below](#) for more information, along with the [streaming responses](#) guide for more information on how to handle the streaming events.

stream_options object Optional Defaults to null

Options for streaming response. Only set this when you set `stream: true`.

› Show properties

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. We generally recommend altering this or `top_p` but not both.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tool and instead generates a message. `auto` means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools. Specifying a particular tool via `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

`none` is the default when no tools are present. `auto` is the default if tools are present.

> Show possible types

tools array Optional

A list of tools the model may call. You can provide either [custom tools](#) or [function tools](#).

> Show possible types

top_logprobs integer Optional

An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability.

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or `temperature` but not both.

user Deprecated string Optional

This field is being replaced by `safety_identifier` and `prompt_cache_key`. Use `prompt_cache_key` instead to maintain caching optimizations. A stable identifier for your end-users. Used to boost cache hit rates by better bucketing similar requests and to help OpenAI detect and prevent abuse. [Learn more.](#)

verbosity string Optional Defaults to medium

Constrains the verbosity of the model's response. Lower values will result in more concise responses, while higher values will result in more verbose responses. Currently supported values are `low`, `medium`, and `high`.

web_search_options object Optional

This tool searches the web for relevant results to use in a response. Learn more about the [web search tool](#).

› Show properties

Returns

Returns a [chat completion](#) object, or a streamed sequence of [chat completion chunk](#) objects if the request is streamed.

Default Image input Streaming Functions Logprobs

Example request

python ⚙️

```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3
4 completion = client.chat.completions.create(
5     model="gpt-5.2",
6     messages=[
7         {"role": "user", "content": "Hello!"}
8     ],
9     logprobs=True,
10    top_logprobs=2
11 )
12
13 print(completion.choices[0].message)
14 print(completion.choices[0].logprobs)
```

Response



```
1 {
2     "id": "chatcmpl-123",
3     "object": "chat.completion",
4     "created": 1702685778,
5     "model": "gpt-4o-mini",
6     "choices": [
7         {
8             "index": 0,
9             "message": {
10                 "role": "assistant",
11                 "content": "Hello! How can I assist you today?"
12             },
13             "logprobs": {
```

```
14 "content": [
15   {
16     "token": "Hello",
17     "logprob": -0.31725305,
18     "bytes": [72, 101, 108, 108, 111],
19     "top_logprobs": [
20       {
21         "token": "Hello",
22         "logprob": -0.31725305,
23         "bytes": [72, 101, 108, 108, 111]
24       },
25       {
26         "token": "Hi",
27         "logprob": -1.3190403,
28         "bytes": [72, 105]
29       }
30     ]
31   },
32   {
33     "token": "!",
34     "logprob": -0.02380986,
35     "bytes": [
36       33
37     ],
38     "top_logprobs": [
39       {
40         "token": "!",
41         "logprob": -0.02380986,
42         "bytes": [33]
```

```
43 },
44 {
45     "token": " there",
46     "logprob": -3.787621,
47     "bytes": [32, 116, 104, 101, 114, 101]
48 }
49 ]
50 },
51 {
52     "token": " How",
53     "logprob": -0.000054669687,
54     "bytes": [32, 72, 111, 119],
55     "top_logprobs": [
56         {
57             "token": " How",
58             "logprob": -0.000054669687,
59             "bytes": [32, 72, 111, 119]
60         },
61         {
62             "token": "<|end|>",
63             "logprob": -10.953937,
64             "bytes": null
65         }
66     ]
67 },
68 {
69     "token": " can",
70     "logprob": -0.015801601,
71     "bytes": [32, 99, 97, 110],
```

```
72     "top_logprobs": [
73         {
74             "token": " can",
75             "logprob": -0.015801601,
76             "bytes": [32, 99, 97, 110]
77         },
78         {
79             "token": " may",
80             "logprob": -4.161023,
81             "bytes": [32, 109, 97, 121]
82         }
83     ],
84 },
85 {
86     "token": " I",
87     "logprob": -3.7697225e-6,
88     "bytes": [
89         32,
90         73
91     ],
92     "top_logprobs": [
93         {
94             "token": " I",
95             "logprob": -3.7697225e-6,
96             "bytes": [32, 73]
97         },
98         {
99             "token": " assist",
100            "logprob": -13.596657,
```

```
101             "bytes": [32, 97, 115, 115, 105, 115, 116]
102         }
103     ]
104 },
105 {
106     "token": " assist",
107     "logprob": -0.04571125,
108     "bytes": [32, 97, 115, 115, 105, 115, 116],
109     "top_logprobs": [
110         {
111             "token": " assist",
112             "logprob": -0.04571125,
113             "bytes": [32, 97, 115, 115, 105, 115, 116]
114         },
115         {
116             "token": " help",
117             "logprob": -3.1089056,
118             "bytes": [32, 104, 101, 108, 112]
119         }
120     ]
121 },
122 {
123     "token": " you",
124     "logprob": -5.4385737e-6,
125     "bytes": [32, 121, 111, 117],
126     "top_logprobs": [
127         {
128             "token": " you",
129             "logprob": -5.4385737e-6,
```

```
130         "bytes": [32, 121, 111, 117]
131     },
132     {
133         "token": " today",
134         "logprob": -12.807695,
135         "bytes": [32, 116, 111, 100, 97, 121]
136     }
137 ]
138 },
139 {
140     "token": " today",
141     "logprob": -0.0040071653,
142     "bytes": [32, 116, 111, 100, 97, 121],
143     "top_logprobs": [
144     {
145         "token": " today",
146         "logprob": -0.0040071653,
147         "bytes": [32, 116, 111, 100, 97, 121]
148     },
149     {
150         "token": "?",
151         "logprob": -5.5247097,
152         "bytes": [63]
153     }
154 ]
155 },
156 {
157     "token": "?",
158     "logprob": -0.0008108172,
```

```
159         "bytes": [63],  
160         "top_logprobs": [  
161             {  
162                 "token": "?",  
163                 "logprob": -0.0008108172,  
164                 "bytes": [63]  
165             },  
166             {  
167                 "token": "?\n",  
168                 "logprob": -7.184561,  
169                 "bytes": [63, 10]  
170             }  
171         ]  
172     }  
173 ]  
174 },  
175     "finish_reason": "stop"  
176 }  
177 ],  
178 "usage": {  
179     "prompt_tokens": 9,  
180     "completion_tokens": 9,  
181     "total_tokens": 18,  
182     "completion_tokens_details": {  
183         "reasoning_tokens": 0,  
184         "accepted_prediction_tokens": 0,  
185         "rejected_prediction_tokens": 0  
186     }  
187 }
```

Get chat completion

188 "system_fingerprint": null
189 }
Get a stored chat completion. Only Chat Completions that have been created with the [store](#) parameter set to true will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python ▾ [Copy](#)

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 print(first_completion)
```

Response

[Copy](#)

```
1  {
2      "object": "chat.completion",
3      "id": "chatcmpl-abc123",
4      "model": "gpt-4o-2024-08-06",
5      "created": 1738960610,
6      "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7      "tool_choice": null,
8      "usage": {
9          "total_tokens": 31,
10         "completion_tokens": 18,
11         "prompt_tokens": 13
12     },
13     "seed": 4944116822809979520,
14     "top_p": 1.0,
15     "temperature": 1.0,
16     "presence_penalty": 0.0,
17     "frequency_penalty": 0.0,
18     "system_fingerprint": "fp_50cad350e4",
19     "input_user": null,
20     "service_tier": "default",
21     "tools": null,
22     "metadata": {},
23     "choices": [
24         {
25             "index": 0,
26             "message": {
27                 "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture",
28                 "role": "assistant",
29                 "tool_calls": null,
30             }
31         }
32     ]
33 }
```

```
30     "function_call": null
31   },
32   "finish_reason": "stop",
33   "logprobs": null
34 }
35 ],
36 "response_format": null
37 }
```

Get chat messages

```
GET https://api.openai.com/v1/chat/completions/{completion_id}/messages
```

Get the messages in a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` will be returned.

Path parameters

completion_id string Required

The ID of the chat completion to retrieve messages from.

Query parameters

after string Optional

Identifier for the last message from the previous pagination request.

limit integer Optional Defaults to 20

Number of messages to retrieve.

order string Optional Defaults to `asc`

Sort order for messages by timestamp. Use `asc` for ascending order or `desc` for descending order.

Defaults to `asc`.

Returns

A list of messages for the specified chat completion.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 first_completion = client.chat.completions.retrieve(completion_id=first_id)
7 messages = client.chat.completions.messages.list(completion_id=first_id)
8 print(messages)
```

Response

🔗

```
1 {
2     "object": "list",
3     "data": [
4         {
5             "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6             "role": "user",
7             "content": "write a haiku about ai",
8             "name": null,
9             "content_parts": null
10        }
11    ],
```

```
12  "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",
13  "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2-0",
14  "has_more": false
15 }
```

List Chat Completions

GET <https://api.openai.com/v1/chat/completions>

List stored Chat Completions. Only Chat Completions that have been stored with the `store` parameter set to `true` will be returned.

Query parameters

after string Optional

Identifier for the last chat completion from the previous pagination request.

limit integer Optional Defaults to 20

Number of Chat Completions to retrieve.

metadata object or null Optional

A list of metadata keys to filter the Chat Completions by. Example:

```
metadata[key1]=value1&metadata[key2]=value2
```

model string Optional

The model used to generate the Chat Completions.

order string Optional Defaults to asc

Sort order for Chat Completions by timestamp. Use `asc` for ascending order or `desc` for descending order. Defaults to `asc`.

Returns

A list of [Chat Completions](#) matching the specified filters.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 print(completions)
```

Response

⌚

```
1  {
2      "object": "list",
3      "data": [
4          {
5              "object": "chat.completion",
6              "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
7              "model": "gpt-4.1-2025-04-14",
8              "created": 1738960610,
9              "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10             "tool_choice": null,
11             "usage": {
12                 "total_tokens": 31,
13                 "completion_tokens": 18,
14                 "prompt_tokens": 13
15             },
16             "seed": 4944116822809979520,
17             "top_p": 1.0,
18             "temperature": 1.0,
19             "presence_penalty": 0.0,
20             "frequency_penalty": 0.0,
21             "system_fingerprint": "fp_50cad350e4",
22             "input_user": null,
23             "service_tier": "default",
24             "tools": null,
25             "metadata": {},
26             "choices": [
27                 {
28                     "index": 0,
29                     "message": {
```

```
30         "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31         "role": "assistant",
32         "tool_calls": null,
33         "function_call": null
34     },
35     "finish_reason": "stop",
36     "logprobs": null
37 }
38 ],
39 "response_format": null
40 }
41 ],
42 "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
43 "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
44 "has_more": false
45 }
```

Update chat completion

POST https://api.openai.com/v1/chat/completions/{completion_id}

Modify a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be modified. Currently, the only supported modification is to update the `metadata` field.

Path parameters

completion_id string Required

The ID of the chat completion to update.

Request body

metadata map Required

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The [ChatCompletion](#) object matching the specified ID.

Example request

python 

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
6 updated_completion = client.chat.completions.update(completion_id=first_id, request_t
7 print(updated_completion)
```

Response



```
1 {
2     "object": "chat.completion",
3     "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMf1mj2",
4     "model": "gpt-4o-2024-08-06",
5     "created": 1738960610,
6     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
7     "tool_choice": null,
8     "usage": {
9         "total_tokens": 31,
10        "completion_tokens": 18,
11        "prompt_tokens": 13
12    },
13    "seed": 4944116822809979520,
14    "top_p": 1.0,
15    "temperature": 1.0,
16    "presence_penalty": 0.0,
17    "frequency_penalty": 0.0,
```

```
18 "system_fingerprint": "fp_50cad350e4",
19 "input_user": null,
20 "service_tier": "default",
21 "tools": null,
22 "metadata": {
23     "foo": "bar"
24 },
25 "choices": [
26     {
27         "index": 0,
28         "message": {
29             "content": "Mind of circuits hum, \nLearning patterns in silence— \nFuture",
30             "role": "assistant",
31             "tool_calls": null,
32             "function_call": null
33         },
34         "finish_reason": "stop",
35         "logprobs": null
36     }
37 ],
38 "response_format": null
39 }
```

Delete chat completion

```
DELETE https://api.openai.com/v1/chat/completions/{completion_id}
```

Delete a stored chat completion. Only Chat Completions that have been created with the `store` parameter set to `true` can be deleted.

Path parameters

completion_id string Required

The ID of the chat completion to delete.

Returns

A deletion confirmation object.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 completions = client.chat.completions.list()
5 first_id = completions[0].id
```

```
6 delete_response = client.chat.completions.delete(completion_id=first_id)
7 print(delete_response)
```

Response



```
1 {
2   "object": "chat.completion.deleted",
3   "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
4   "deleted": true
5 }
```

The chat completion object

Represents a chat completion response returned by model, based on the provided input.

choices array

A list of chat completion choices. Can be more than one if `n` is greater than 1.

› Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created.

id string

A unique identifier for the chat completion.

model string

The model used for the chat completion.

object string

The object type, which is always `chat.completion`.

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to 'flex' or 'priority', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

system_fingerprint Deprecated string

This fingerprint represents the backend configuration that the model runs with.

Can be used in conjunction with the `seed` request parameter to understand when backend changes have been made that might impact determinism.

usage object

Usage statistics for the completion request.

› Show properties

OBJECT The chat completion object



```
1  {
2      "id": "chatcmpl-B9MHDbs1fkBeAs8l4bebGdFOJ6PeG",
3      "object": "chat.completion",
4      "created": 1741570283,
5      "model": "gpt-4o-2024-08-06",
6      "choices": [
7          {
8              "index": 0,
9              "message": {
10                  "role": "assistant",
11                  "content": "The image shows a wooden boardwalk path running through a lush green forest. The path is made of light-colored wood planks and leads towards a bright opening in the trees. The surrounding area is filled with dense foliage and tall trees. The lighting suggests it's daytime, with sunlight filtering through the canopy above.",
12                  "refusal": null,
13                  "annotations": []
14              },
15              "logprobs": null,
```

```
16     "finish_reason": "stop"
17   }
18 ],
19 "usage": {
20   "prompt_tokens": 1117,
21   "completion_tokens": 46,
22   "total_tokens": 1163,
23   "prompt_tokens_details": {
24     "cached_tokens": 0,
25     "audio_tokens": 0
26   },
27   "completion_tokens_details": {
28     "reasoning_tokens": 0,
29     "audio_tokens": 0,
30     "accepted_prediction_tokens": 0,
31     "rejected_prediction_tokens": 0
32   }
33 },
34 "service_tier": "default",
35 "system_fingerprint": "fp_fc9f1d7035"
36 }
```

The chat completion list object

An object representing a list of Chat Completions.

data array

An array of chat completion objects.

› Show properties

first_id string

The identifier of the first chat completion in the data array.

has_more boolean

Indicates whether there are more Chat Completions available.

last_id string

The identifier of the last chat completion in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion list object



```
1  {
2    "object": "list",
```

```
3 "data": [
4 {
5     "object": "chat.completion",
6     "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",
7     "model": "gpt-4o-2024-08-06",
8     "created": 1738960610,
9     "request_id": "req_ded8ab984ec4bf840f37566c1011c417",
10    "tool_choice": null,
11    "usage": {
12        "total_tokens": 31,
13        "completion_tokens": 18,
14        "prompt_tokens": 13
15    },
16    "seed": 4944116822809979520,
17    "top_p": 1.0,
18    "temperature": 1.0,
19    "presence_penalty": 0.0,
20    "frequency_penalty": 0.0,
21    "system_fingerprint": "fp_50cad350e4",
22    "input_user": null,
23    "service_tier": "default",
24    "tools": null,
25    "metadata": {},
26    "choices": [
27        {
28            "index": 0,
29            "message": {
30                "content": "Mind of circuits hum, \nLearning patterns in silence— \nFu
31                "role": "assistant",
```

```
32         "tool_calls": null,  
33         "function_call": null  
34     },  
35     "finish_reason": "stop",  
36     "logprobs": null  
37   }  
38 ],  
39   "response_format": null  
40 }  
41 ],  
42 "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
43 "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2",  
44 "has_more": false  
45 }
```

The chat completion message list object

An object representing a list of chat completion messages.

data array

An array of chat completion message objects.

› Show properties

first_id string

The identifier of the first chat message in the data array.

has_more boolean

Indicates whether there are more chat messages available.

last_id string

The identifier of the last chat message in the data array.

object string

The type of this object. It is always set to "list".

OBJECT The chat completion message list object



```
1  {
2    "object": "list",
3    "data": [
4      {
5        "id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
6        "role": "user",
7        "content": "write a haiku about ai",
8        "name": null,
9        "content_parts": null
10      }
```

```
11  ],
12  "first_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
13  "last_id": "chatcmpl-AyPNinnUqUDYo9SAdA52NobMflmj2-0",
14  "has_more": false
15 }
```

PREVIOUS
< **Server events**

NEXT
Streaming >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Streaming

Stream Chat Completions in real time. Receive chunks of completions returned from the model using server-sent events. [Learn more.](#)

The chat completion chunk object

Represents a streamed chunk of a chat completion response returned by the model, based on the provided input. [Learn more.](#)

choices array

A list of chat completion choices. Can contain more than one elements if `n` is greater than 1. Can also be empty for the last chunk if you set `stream_options: {"include_usage": true}`.

› Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created. Each chunk has the same timestamp.

id string

A unique identifier for the chat completion. Each chunk has the same ID.

model string

The model to generate the completion.

object string

The object type, which is always `chat.completion.chunk`.

service_tier string

Specifies the processing type used for serving the request.

- If set to 'auto', then the request will be processed with the service tier configured in the Project settings. Unless otherwise configured, the Project will use 'default'.
- If set to 'default', then the request will be processed with the standard pricing and performance for the selected model.
- If set to 'flex' or 'priority', then the request will be processed with the corresponding service tier.
- When not set, the default behavior is 'auto'.

When the `service_tier` parameter is set, the response body will include the `service_tier` value based on the processing mode actually used to serve the request. This response value may be different from the value set in the parameter.

system_fingerprint Deprecated string

This fingerprint represents the backend configuration that the model runs with. Can be used in conjunction with the `seed` request parameter to understand when backend changes have been made that might impact determinism.

usage object or null

Usage statistics for the completion request.

› Show properties

OBJECT The chat completion chunk object



```
1 {"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-3.5-turbo", "content": "Hello, I am a large language model created by OpenAI. I can answer your questions and provide information on a wide range of topics. How can I help you today?"}, {"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-3.5-turbo", "content": "I am trained on a massive amount of text data and can generate human-like responses. I can answer questions, write stories, and even code programs. I am here to assist you with any inquiries you may have."}, {"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-3.5-turbo", "content": "Please let me know if you have any specific questions or topics you would like me to address. I am always learning and improving, so your feedback is valuable to me."}, {"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-3.5-turbo", "content": "Thank you for using my services! If you have any further questions or need additional assistance, please don't hesitate to ask. I am here to help."}
```



PREVIOUS

< Chat Completions

NEXT

Assistants >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Assistants Beta

Build assistants that can call models and use tools to perform tasks.

[Get started with the Assistants API](#)

Create assistant Beta

```
POST https://api.openai.com/v1/assistants
```

Create an assistant with a model and instructions.

Request body

model string Required

ID of the model to use. You can use the [List models](#) API to see all of your available models, or see our [Model overview](#) for descriptions of them.

description string Optional

The description of the assistant. The maximum length is 512 characters.

instructions string Optional

The system instructions that the assistant uses. The maximum length is 256,000 characters.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

name string Optional

The name of the assistant. The maximum length is 256 characters.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": {...} }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_resources object Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array Optional Defaults to []

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types `code_interpreter`, `file_search`, or `function`.

› Show possible types

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

Returns

An assistant object.

Code Interpreter

Files

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_assistant = client.beta.assistants.create(
5     instructions="You are a personal math tutor. When asked a question, write and ru",
6     name="Math Tutor",
7     tools=[{"type": "code_interpreter"}],
8     model="gpt-4o",
9 )
10 print(my_assistant)
```

Response

🔗

```
1 {
2     "id": "asst_abc123",
3     "object": "assistant",
4     "created_at": 1698984975,
5     "name": "Math Tutor",
6     "description": null,
7     "model": "gpt-4o",
```

```
8  "instructions": "You are a personal math tutor. When asked a question, write and r
9  "tools": [
10   {
11     "type": "code_interpreter"
12   }
13 ],
14 "metadata": {},
15 "top_p": 1.0,
16 "temperature": 1.0,
17 "response_format": "auto"
18 }
```

List assistants

Beta

GET https://api.openai.com/v1/assistants

Returns a list of assistants.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of [assistant](#) objects.

Example request

python ▾ 

```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3
4 my_assistants = client.beta.assistants.list(
5     order="desc",
6     limit="20",
7 )
8 print(my_assistants.data)
```

Response



```
1 {
2     "object": "list",
3     "data": [
4         {
5             "id": "asst_abc123",
6             "object": "assistant",
7             "created_at": 1698982736,
8             "name": "Coding Tutor",
9             "description": null,
10            "model": "gpt-4o",
11            "instructions": "You are a helpful assistant designed to make me better at codin",
12            "tools": [],
13            "tool_resources": {},
14            "metadata": {},
15            "top_p": 1.0,
16            "temperature": 1.0,
17            "response_format": "auto"
18        },
19        {
```

```
20     "id": "asst_abc456",
21     "object": "assistant",
22     "created_at": 1698982718,
23     "name": "My Assistant",
24     "description": null,
25     "model": "gpt-4o",
26     "instructions": "You are a helpful assistant designed to make me better at cod
27     "tools": [],
28     "tool_resources": {},
29     "metadata": {},
30     "top_p": 1.0,
31     "temperature": 1.0,
32     "response_format": "auto"
33 },
34 {
35     "id": "asst_abc789",
36     "object": "assistant",
37     "created_at": 1698982643,
38     "name": null,
39     "description": null,
40     "model": "gpt-4o",
41     "instructions": null,
42     "tools": [],
43     "tool_resources": {},
44     "metadata": {},
45     "top_p": 1.0,
46     "temperature": 1.0,
47     "response_format": "auto"
48 }
```

```
49  ],
50  "first_id": "asst_abc123",
51  "last_id": "asst_abc789",
52  "has_more": false
53 }
```

Retrieve assistant Beta

```
GET https://api.openai.com/v1/assistants/{assistant_id}
```

Retrieves an assistant.

Path parameters

assistant_id string Required

The ID of the assistant to retrieve.

Returns

The assistant object matching the specified ID.

Example request

python ↗

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_assistant = client.beta.assistants.retrieve("asst_abc123")
5 print(my_assistant)
```

Response

↗

```
1 {
2     "id": "asst_abc123",
3     "object": "assistant",
4     "created_at": 1699009709,
5     "name": "HR Helper",
6     "description": null,
7     "model": "gpt-4o",
8     "instructions": "You are an HR bot, and you have access to files to answer employee questions.",
9     "tools": [
10         {
11             "type": "file_search"
12         }
13     ],
14     "metadata": {}
```

```
15 "top_p": 1.0,  
16 "temperature": 1.0,  
17 "response_format": "auto"  
18 }
```

Modify assistant Beta

POST https://api.openai.com/v1/assistants/{assistant_id}

Modifies an assistant.

Path parameters

assistant_id string Required

The ID of the assistant to modify.

Request body

description string Optional

The description of the assistant. The maximum length is 512 characters.

instructions string Optional

The system instructions that the assistant uses. The maximum length is 256,000 characters.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

ID of the model to use. You can use the [List models](#) API to see all of your available models, or see our [Model overview](#) for descriptions of them.

name string Optional

The name of the assistant. The maximum length is 256 characters.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values

in gpt-5.1.

- All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_resources object Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array Optional Defaults to []

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types `code_interpreter`, `file_search`, or `function`.

› Show possible types

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

Returns

The modified `assistant` object.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_updated_assistant = client.beta.assistants.update(
5     "asst_abc123",
6     instructions="You are an HR bot, and you have access to files to answer employee c
7     name="HR Helper",
8     tools=[{"type": "file_search"}],
9     model="gpt-4o"
10 )
11
12 print(my_updated_assistant)
```

Response



```
1 {
2     "id": "asst_123",
3     "object": "assistant",
4     "created_at": 1699009709,
5     "name": "HR Helper",
6     "description": null,
7     "model": "gpt-4o",
8     "instructions": "You are an HR bot, and you have access to files to answer employee c
9     "tools": [
10         {
11             "type": "file_search"
12         }
13     ]
14 }
```

```
13 ],
14 "tool_resources": {
15   "file_search": {
16     "vector_store_ids": []
17   }
18 },
19 "metadata": {},
20 "top_p": 1.0,
21 "temperature": 1.0,
22 "response_format": "auto"
23 }
```

Delete assistant Beta

```
DELETE https://api.openai.com/v1/assistants/{assistant_id}
```

Delete an assistant.



Path parameters

assistant_id string Required

The ID of the assistant to delete.

Returns

Deletion status

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.beta.assistants.delete("asst_abc123")
5 print(response)
```

Response

⌚

```
1 {
2   "id": "asst_abc123",
3   "object": "assistant.deleted",
4   "deleted": true
5 }
```

The assistant object Beta

Represents an [assistant](#) that can call the model and use tools.

created_at integer

The Unix timestamp (in seconds) for when the assistant was created.

description string

The description of the assistant. The maximum length is 512 characters.

id string

The identifier, which can be referenced in API endpoints.

instructions string

The system instructions that the assistant uses. The maximum length is 256,000 characters.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

ID of the model to use. You can use the [List models](#) API to see all of your available models, or see our [Model overview](#) for descriptions of them.

name string

The name of the assistant. The maximum length is 256 characters.

object string

The object type, which is always `assistant`.

response_format "auto" or object

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_resources object

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types `code_interpreter`, `file_search`, or `function`.

› Show possible types

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

OBJECT The `assistant` object



PREVIOUS

< Streaming

NEXT

Threads

Assistants

Beta



Build assistants that can call models and use tools to perform tasks.

[Get started with the Assistants API](#)

Create assistant

Beta



```
POST https://api.openai.com/v1/assistants
```

Create an assistant with a model and instructions.

Request body

model string Required

ID of the model to use. You can use the [List models API](#) to see all of your available models, or see our [Model overview](#) for descriptions of them.

description string Optional

The description of the assistant. The maximum length is 512 characters.

instructions string Optional

The system instructions that the assistant uses. The maximum length is 256,000 characters.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

name string Optional

The name of the assistant. The maximum length is 256 characters.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for reasoning_models. Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

`gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in gpt-5.1.

All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.

The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.

`xhigh` is supported for all models after `gpt-5.1-codex-max`.

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since [gpt-3.5-turbo-1106](#).

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_resources object Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array Optional Defaults to []

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types `code_interpreter`, `file_search`, or `function`.

> Show possible types

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

Returns

An assistant object.

Code Interpreter Files

Example request

```
from openai import OpenAI
client = OpenAI()

my_assistant = client.beta.assistants.create(
    instructions="You are an HR bot, and you have access to files to answer employee questions abc",
    name="HR Helper",
    tools=[{"type": "file_search"}],
    tool_resources={"file_search": {"vector_store_ids": ["vs_123"]}},
    model="gpt-4o"
)
print(my_assistant)
```

Response

```
{  
  "id": "asst_abc123",  
  "object": "assistant",  
  "created_at": 1699009403,  
  "name": "HR Helper",  
  "description": null,  
  "model": "gpt-4o",  
  "instructions": "You are an HR bot, and you have access to files to answer employee questions about company policies and benefits.",  
  "tools": [  
    {  
      "type": "file_search"  
    }  
  ],  
  "tool_resources": {  
    "file_search": {  
      "vector_store_ids": ["vs_123"]  
    }  
  },  
  "metadata": {},  
  "top_p": 1.0,  
  "temperature": 1.0,  
  "response_format": "auto"  
}
```

List assistants

Beta



```
GET https://api.openai.com/v1/assistants
```

Returns a list of assistants.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include after=obj_foo in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with obj_foo, your subsequent call can include before=obj_foo in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of [assistant](#) objects.

Example request

```
from openai import OpenAI
client = OpenAI()

my_assistants = client.beta.assistants.list(
    order="desc",
    limit="20",
)
print(my_assistants.data)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "asst_abc123",
      "object": "assistant",
      "created_at": 1698982736,
```

```
"name": "Coding Tutor",
"description": null,
"model": "gpt-4o",
"instructions": "You are a helpful assistant designed to make me better at coding!",
"tools": [],
"tool_resources": {},
"metadata": {},
"top_p": 1.0,
"temperature": 1.0,
"response_format": "auto"
},
{
"id": "asst_abc456",
"object": "assistant",
"created_at": 1698982718,
"name": "My Assistant",
"description": null,
"model": "gpt-4o",
"instructions": "You are a helpful assistant designed to make me better at coding!",
"tools": [],
"tool_resources": {},
"metadata": {},
"top_p": 1.0,
"temperature": 1.0,
"response_format": "auto"
},
{
"id": "asst_abc789",
"object": "assistant",
```

```
"created_at": 1698982643,  
"name": null,  
"description": null,  
"model": "gpt-4o",  
"instructions": null,  
"tools": [],  
"tool_resources": {},  
"metadata": {},  
"top_p": 1.0,  
"temperature": 1.0,  
"response_format": "auto"  
}  
,  
"first_id": "asst_abc123",  
"last_id": "asst_abc789",  
"has_more": false  
}
```

Retrieve assistant Beta



GET https://api.openai.com/v1/assistants/{assistant_id}

Retrieves an assistant.

Path parameters

assistant_id string Required

The ID of the assistant to retrieve.

Returns

The assistant object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

my_assistant = client.beta.assistants.retrieve("asst_abc123")
print(my_assistant)
```

Response

```
{
  "id": "asst_abc123",
  "object": "assistant",
  "created_at": 1699009709,
```

```
"name": "HR Helper",
"description": null,
"model": "gpt-4o",
"instructions": "You are an HR bot, and you have access to files to answer employee questions about company policies and benefits.",
"tools": [
  {
    "type": "file_search"
  }
],
"metadata": {},
"top_p": 1.0,
"temperature": 1.0,
"response_format": "auto"
}
```

Modify assistant Beta



POST https://api.openai.com/v1/assistants/{assistant_id}

Modifies an assistant.

Path parameters

assistant_id string Required

The ID of the assistant to modify.

Request body

description string Optional

The description of the assistant. The maximum length is 512 characters.

instructions string Optional

The system instructions that the assistant uses. The maximum length is 256,000 characters.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

ID of the model to use. You can use the [List models](#) API to see all of your available models, or see our [Model overview](#) for descriptions of them.

name string Optional

The name of the assistant. The maximum length is 256 characters.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

`gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.

All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.

The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.

`xhigh` is supported for all models after `gpt-5.1-codex-max`.

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

temperature number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_resources object Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array Optional Defaults to []

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types `code_interpreter` , `file_search` , or `function` .

› Show possible types

top_p number Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

Returns

The modified `assistant` object.

Example request

```
from openai import OpenAI
client = OpenAI()

my_updated_assistant = client.beta.assistants.update(
    "asst_abc123",
    instructions="You are an HR bot, and you have access to files to answer employee questions about",
    name="HR Helper",
    tools=[{"type": "file_search"}],
    model="gpt-4o"
)

print(my_updated_assistant)
```

Response

```
{
  "id": "asst_123",
  "object": "assistant",
  "created_at": 1699009709,
  "name": "HR Helper",
  "description": null,
  "model": "gpt-4o",
  "instructions": "You are an HR bot, and you have access to files to answer employee questions about",
  "tools": [
    {
      "type": "file_search"
    }
  ]
}
```

```
],
  "tool_resources": {
    "file_search": {
      "vector_store_ids": []
    }
  },
  "metadata": {},
  "top_p": 1.0,
  "temperature": 1.0,
  "response_format": "auto"
}
```

Delete assistant Beta

🔗

```
DELETE https://api.openai.com/v1/assistants/{assistant_id}
```

Delete an assistant.

Path parameters

assistant_id string Required

The ID of the assistant to delete.

Returns

Deletion status

Example request

```
from openai import OpenAI
client = OpenAI()

response = client.beta.assistants.delete("asst_abc123")
print(response)
```

Response

```
{
  "id": "asst_abc123",
  "object": "assistant.deleted",
  "deleted": true
}
```

The assistant object

Beta



Represents an `assistant` that can call the model and use tools.

created_at integer

The Unix timestamp (in seconds) for when the assistant was created.

description string

The description of the assistant. The maximum length is 512 characters.

id string

The identifier, which can be referenced in API endpoints.

instructions string

The system instructions that the assistant uses. The maximum length is 256,000 characters.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

ID of the model to use. You can use the [List models](#) API to see all of your available models, or see our [Model overview](#) for descriptions of them.

name string

The name of the assistant. The maximum length is 256 characters.

object string

The object type, which is always `assistant`.

response_format "auto" or object

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

> Show possible types

temperature number

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_resources object

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the

`code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

> Show properties

tools array

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types `code_interpreter`, `file_search`, or `function`.

> Show possible types

top_p number

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

OBJECT The assistant object

```
{  
  "id": "asst_abc123",  
  "object": "assistant",  
  "created_at": 1698984975,  
  "name": "Math Tutor",  
  "description": null,  
  "model": "gpt-4o",  
  "instructions": "You are a personal math tutor. When asked a question, write and run Python code.",  
  "tools": [  
    {"name": "Calculator", "url": "https://www.wolframalpha.com"},  
    {"name": "Python Interpreter", "url": "https://repl.it/languages/python3"},  
    {"name": "Symbolab", "url": "https://www.symbolab.com"},  
    {"name": "Wolfram Alpha", "url": "https://www.wolframalpha.com"}]  
}
```

```
{  
    "type": "code_interpreter"  
}  
,  
"metadata": {},  
"top_p": 1.0,  
"temperature": 1.0,  
"response_format": "auto"  
}
```

< PREVIOUS
Streaming

NEXT >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Threads

Beta

Create threads that assistants can interact with.

Related guide: [Assistants](#)

Create thread

Beta

```
POST https://api.openai.com/v1/threads
```

Create a thread.

Request body

messages array Optional

A list of [messages](#) to start the thread with.

› Show properties

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

tool_resources object Optional

A set of resources that are made available to the assistant's tools in this thread. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

Returns

A [thread](#) object.

Empty

Messages

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 empty_thread = client.beta.threads.create()
5 print(empty_thread)
```

Response



```
1 {
2   "id": "thread_abc123",
3   "object": "thread",
4   "created_at": 1699012949,
5   "metadata": {},
6   "tool_resources": {}
7 }
```

Retrieve thread Beta

```
GET https://api.openai.com/v1/threads/{thread_id}
```

Retrieves a thread.

Path parameters

thread_id string Required

The ID of the thread to retrieve.

Returns

The thread object matching the specified ID.

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_thread = client.beta.threads.retrieve("thread_abc123")
5 print(my_thread)
```

Response

Copy

```
1 {
2     "id": "thread_abc123",
3     "object": "thread",
```

```
4 "created_at": 1699014083,  
5 "metadata": {},  
6 "tool_resources": {  
7     "code_interpreter": {  
8         "file_ids": []  
9     }  
10 }  
11 }
```

Modify thread Beta

POST https://api.openai.com/v1/threads/{thread_id}

Modifies a thread.

Path parameters

thread_id string Required

The ID of the thread to modify. Only the `metadata` can be modified.

Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

tool_resources object Optional

A set of resources that are made available to the assistant's tools in this thread. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

Returns

The modified `thread` object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_updated_thread = client.beta.threads.update()
```

```
5     "thread_abc123",
6     metadata={
7         "modified": "true",
8         "user": "abc123"
9     }
10 )
11 print(my_updated_thread)
```

Response



```
1  {
2      "id": "thread_abc123",
3      "object": "thread",
4      "created_at": 1699014083,
5      "metadata": {
6          "modified": "true",
7          "user": "abc123"
8      },
9      "tool_resources": {}
10 }
```

Delete thread

Beta

```
DELETE https://api.openai.com/v1/threads/{thread_id}
```

Delete a thread.

Path parameters

thread_id string Required

The ID of the thread to delete.

Returns

Deletion status

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.beta.threads.delete("thread_abc123")
5 print(response)
```

Response



```
1 {
2   "id": "thread_abc123",
3   "object": "thread.deleted",
4   "deleted": true
5 }
```

The thread object Beta

Represents a thread that contains [messages](#).

created_at integer

The Unix timestamp (in seconds) for when the thread was created.

id string

The identifier, which can be referenced in API endpoints.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

object string

The object type, which is always `thread`.

tool_resources object

A set of resources that are made available to the assistant's tools in this thread. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

OBJECT The `thread` object

```
1 {
2   "id": "thread_abc123",
3   "object": "thread",
4   "created_at": 1698107661,
5   "metadata": {}
6 }
```

PREVIOUS

< Assistants

NEXT

Messages >

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Threads

Beta

Create threads that assistants can interact with.

Related guide: [Assistants](#)

Create thread

Beta

```
POST https://api.openai.com/v1/threads
```

Create a thread.

Request body

messages array Optional

A list of messages to start the thread with.

› Show properties

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

tool_resources object Optional

A set of resources that are made available to the assistant's tools in this thread. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

Returns

A thread object.

Empty

Messages

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 message_thread = client.beta.threads.create(
5     messages=[
6         {
7             "role": "user",
8             "content": "Hello, what is AI?"
9         },
10        {
11            "role": "user",
12            "content": "How does AI work? Explain it in simple terms."
13        },
14    ]
15 )
16
17 print(message_thread)
```

Response



```
1 {
2     "id": "thread_abc123",
3     "object": "thread",
4     "created_at": 1699014083,
5     "metadata": {},
6     "tool_resources": {}
7 }
```

Retrieve thread Beta

```
GET https://api.openai.com/v1/threads/{thread_id}
```

Retrieves a thread.

Path parameters

thread_id string Required

The ID of the thread to retrieve.

Returns

The thread object matching the specified ID.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_thread = client.beta.threads.retrieve("thread_abc123")
5 print(my_thread)
```

Response



```
1 {
2     "id": "thread_abc123",
3     "object": "thread",
4     "created_at": 1699014083,
5     "metadata": {},
6     "tool_resources": {
7         "code_interpreter": {
8             "file_ids": []
9         }
10    }
11 }
```

Modify thread

Beta

```
POST https://api.openai.com/v1/threads/{thread_id}
```

Modifies a thread.

Path parameters

thread_id string Required

The ID of the thread to modify. Only the `metadata` can be modified.

Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

tool_resources object Optional

A set of resources that are made available to the assistant's tools in this thread. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

Returns

The modified thread object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 my_updated_thread = client.beta.threads.update(
5     "thread_abc123",
6     metadata={
7         "modified": "true",
8         "user": "abc123"
9     }
10 )
11 print(my_updated_thread)
```

Response

🔗

```
1 {
2     "id": "thread_abc123",
3     "object": "thread",
4     "created_at": 1699014083,
5     "metadata": {
6         "modified": "true",
7         "user": "abc123"
```

```
8  },
9  "tool_resources": {}
10 }
```

Delete thread Beta

```
DELETE https://api.openai.com/v1/threads/{thread_id}
```

Delete a thread.

Path parameters

thread_id string Required

The ID of the thread to delete.

Returns

Deletion status

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.beta.threads.delete("thread_abc123")
5 print(response)
```

Response

🔗

```
1 {
2   "id": "thread_abc123",
3   "object": "thread.deleted",
4   "deleted": true
5 }
```

The thread object Beta

Represents a thread that contains [messages](#).

created_at integer

The Unix timestamp (in seconds) for when the thread was created.

id string

The identifier, which can be referenced in API endpoints.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

object string

The object type, which is always `thread`.

tool_resources object

A set of resources that are made available to the assistant's tools in this thread. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

> Show properties

OBJECT The thread object



```
1 {
2   "id": "thread_abc123",
3   "object": "thread",
```

```
4  "created_at": 1698107661,  
5  "metadata": {}  
6 }
```

PREVIOUS
< **Assistants**

NEXT
Messages >

Messages Beta



Create messages within threads

Related guide: [Assistants](#)

Create message Beta



```
POST https://api.openai.com/v1/threads/{thread_id}/messages
```

Create a message.

Path parameters

thread_id string Required

The ID of the [thread](#) to create a message for.

Request body

content string or array Required

› Show possible types

role string Required

The role of the entity that is creating the message. Allowed values include:

user : Indicates the message is sent by an actual user and should be used in most cases to represent user-generated messages.

assistant : Indicates the message is generated by the assistant. Use this value to insert messages from the assistant into the conversation.

attachments array Optional

A list of files attached to the message, and the tools they should be added to.

› Show properties

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

A [message](#) object.

Example request

```
from openai import OpenAI
client = OpenAI()

thread_message = client.beta.threads.messages.create(
    "thread_abc123",
    role="user",
    content="How does AI work? Explain it in simple terms.",
)
print(thread_message)
```

Response

```
{
  "id": "msg_abc123",
  "object": "thread.message",
  "created_at": 1713226573,
  "assistant_id": null,
  "thread_id": "thread_abc123",
  "run_id": null,
  "role": "user",
  "content": [
    {
      "type": "text",
      "text": {
        "value": "How does AI work? Explain it in simple terms.",
        "annotations": []
      }
    }
  ]
}
```

```
    },
  ],
  "attachments": [],
  "metadata": {}
}
```

List messages Beta



GET `https://api.openai.com/v1/threads/{thread_id}/messages`

Returns a list of messages for a given thread.

Path parameters

thread_id string Required

The ID of the thread the messages belong to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with `obj_foo`, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with `obj_foo`, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

run_id string Optional

Filter messages by the run ID that generated them.

Returns

A list of `message` objects.

Example request

```
from openai import OpenAI  
client = OpenAI()
```

```
thread_messages = client.beta.threads.messages.list("thread_abc123")
print(thread_messages.data)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "msg_abc123",
      "object": "thread.message",
      "created_at": 1699016383,
      "assistant_id": null,
      "thread_id": "thread_abc123",
      "run_id": null,
      "role": "user",
      "content": [
        {
          "type": "text",
          "text": {
            "value": "How does AI work? Explain it in simple terms.",
            "annotations": []
          }
        }
      ],
      "attachments": [],
      "metadata": {}
    },
  ]
```

```
{  
    "id": "msg_abc456",  
    "object": "thread.message",  
    "created_at": 1699016383,  
    "assistant_id": null,  
    "thread_id": "thread_abc123",  
    "run_id": null,  
    "role": "user",  
    "content": [  
        {  
            "type": "text",  
            "text": {  
                "value": "Hello, what is AI?",  
                "annotations": []  
            }  
        }  
    ],  
    "attachments": [],  
    "metadata": {}  
},  
    "first_id": "msg_abc123",  
    "last_id": "msg_abc456",  
    "has_more": false  
}
```

Retrieve message

Beta



```
GET https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}
```

Retrieve a message.

Path parameters

message_id string Required

The ID of the message to retrieve.

thread_id string Required

The ID of the thread to which this message belongs.

Returns

The message object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()
```

```
message = client.beta.threads.messages.retrieve(  
    message_id="msg_abc123",  
    thread_id="thread_abc123",  
)  
print(message)
```

Response

```
{  
    "id": "msg_abc123",  
    "object": "thread.message",  
    "created_at": 1699017614,  
    "assistant_id": null,  
    "thread_id": "thread_abc123",  
    "run_id": null,  
    "role": "user",  
    "content": [  
        {  
            "type": "text",  
            "text": {  
                "value": "How does AI work? Explain it in simple terms.",  
                "annotations": []  
            }  
        }  
    ],  
    "attachments": [],  
    "metadata": {}  
}
```

Modify message Beta

ⓘ

```
POST https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}
```

Modifies a message.

Path parameters

message_id string Required

The ID of the message to modify.

thread_id string Required

The ID of the thread to which this message belongs.

Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The modified message object.

Example request

```
from openai import OpenAI
client = OpenAI()

message = client.beta.threads.messages.update(
    message_id="msg_abc12",
    thread_id="thread_abc123",
    metadata={
        "modified": "true",
        "user": "abc123",
    },
)
print(message)
```

Response

```
{
    "id": "msg_abc123",
    "object": "thread.message",
```

```
"created_at": 1699017614,  
"assistant_id": null,  
"thread_id": "thread_abc123",  
"run_id": null,  
"role": "user",  
"content": [  
  {  
    "type": "text",  
    "text": {  
      "value": "How does AI work? Explain it in simple terms.",  
      "annotations": []  
    }  
  }  
],  
"file_ids": [],  
"metadata": {  
  "modified": "true",  
  "user": "abc123"  
}  
}
```

Delete message

Beta



DELETE https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}

Deletes a message.

Path parameters

message_id string Required

The ID of the message to delete.

thread_id string Required

The ID of the thread to which this message belongs.

Returns

Deletion status

Example request

```
from openai import OpenAI
client = OpenAI()

deleted_message = client.beta.threads.messages.delete(
    message_id="msg_abc12",
    thread_id="thread_abc123",
)
print(deleted_message)
```

Response

```
{  
  "id": "msg_abc123",  
  "object": "thread.message.deleted",  
  "deleted": true  
}
```

The message object

Beta



Represents a message within a [thread](#).

assistant_id string

If applicable, the ID of the [assistant](#) that authored this message.

attachments array

A list of files attached to the message, and the tools they were added to.

› Show properties

completed_at integer

The Unix timestamp (in seconds) for when the message was completed.

content array

The content of the message in array of text and/or images.

› Show possible types

created_at integer

The Unix timestamp (in seconds) for when the message was created.

id string

The identifier, which can be referenced in API endpoints.

incomplete_at integer

The Unix timestamp (in seconds) for when the message was marked as incomplete.

incomplete_details object

On an incomplete message, details about why the message is incomplete.

› Show properties

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

object string

The object type, which is always `thread.message`.

role string

The entity that produced the message. One of `user` or `assistant`.

run_id string

The ID of the `run` associated with the creation of this message. Value is `null` when messages are created manually using the create message or create thread endpoints.

status string

The status of the message, which can be either `in_progress`, `incomplete`, or `completed`.

thread_id string

The `thread` ID that this message belongs to.

OBJECT The message object

```
{  
  "id": "msg_abc123",  
  "object": "thread.message",  
  "created_at": 1698983503,  
  "thread_id": "thread_abc123",  
  "role": "assistant",  
  "content": [  
    {  
      "type": "text",  
      "text": {  
        "value": "Hi! How can I help you today?",  
        "annotations": []  
      }  
    }  
  ]  
}
```

```
        }
    }
],
"assistant_id": "asst_abc123",
"run_id": "run_abc123",
"attachments": [],
"metadata": {}
}
```

< PREVIOUS
Threads

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Runs Beta

Represents an execution run on a thread.

Related guide: [Assistants](#)

Create run Beta

```
POST https://api.openai.com/v1/threads/{thread_id}/runs
```

Create a run.

Path parameters

thread_id string Required

The ID of the thread to run.

Query parameters

include[] array Optional

A list of additional fields to include in the response. Currently the only supported value is

`step_details.tool_calls[*].file_search.results[*].content` to fetch the file search result content.

See the [file search tool documentation](#) for more information.

Request body

assistant_id string Required

The ID of the [assistant](#) to use to execute this run.

additional_instructions string or null Optional

Appends additional instructions at the end of the instructions for the run. This is useful for modifying the behavior on a per-run basis without overriding other instructions.

additional_messages array or null Optional

Adds additional messages to the thread before creating the run.

› Show properties

instructions string or null Optional

Overrides the [instructions](#) of the assistant. This is useful for modifying the behavior on a per-run basis.

max_completion_tokens integer or null Optional

The maximum number of completion tokens that may be used over the course of the run. The run will make a best effort to use only the number of completion tokens specified, across multiple turns of the run. If the run exceeds the number of completion tokens specified, the run will end with status [incomplete](#). See [incomplete_details](#) for more info.

max_prompt_tokens integer or null Optional

The maximum number of prompt tokens that may be used over the course of the run. The run will make a best effort to use only the number of prompt tokens specified, across multiple turns of the run. If the run exceeds the number of prompt tokens specified, the run will end with status [incomplete](#). See [incomplete_details](#) for more info.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

The ID of the [Model](#) to be used to execute this run. If a value is provided here, it will override the model associated with the assistant. If not, the model associated with the assistant will be used.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": {...} }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

stream boolean or null Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

temperature number or null Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message. `auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools before responding to the user. Specifying a particular tool like `{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tools array or null Optional

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

› Show possible types

top_p number or null Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

truncation_strategy object or null Optional

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

› Show properties

Returns

A [run](#) object.

Default

Streaming

Streaming with Functions

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.create(
5     thread_id="thread_abc123",
6     assistant_id="asst_abc123"
7 )
8
9 print(run)
```

Response

🔗

```
1 {
2     "id": "run_abc123",
3     "object": "thread.run",
4     "created_at": 1699063290,
5     "assistant_id": "asst_abc123",
6     "thread_id": "thread_abc123",
7     "status": "queued",
8     "started_at": 1699063290,
9     "expires_at": null,
10    "cancelled_at": null,
11    "failed_at": null,
12    "completed_at": 1699063291,
13    "last_error": null,
14    "model": "gpt-4o",
```

```
15 "instructions": null,  
16 "incomplete_details": null,  
17 "tools": [  
18     {  
19         "type": "code_interpreter"  
20     }  
21 ],  
22 "metadata": {},  
23 "usage": null,  
24 "temperature": 1.0,  
25 "top_p": 1.0,  
26 "max_prompt_tokens": 1000,  
27 "max_completion_tokens": 1000,  
28 "truncation_strategy": {  
29     "type": "auto",  
30     "last_messages": null  
31 },  
32 "response_format": "auto",  
33 "tool_choice": "auto",  
34 "parallel_tool_calls": true  
35 }
```

Create thread and run

Beta

POST <https://api.openai.com/v1/threads/runs>

Create a thread and run it in one request.

Request body

assistant_id string Required

The ID of the [assistant](#) to use to execute this run.

instructions string or null Optional

Override the default system message of the assistant. This is useful for modifying the behavior on a per-run basis.

max_completion_tokens integer or null Optional

The maximum number of completion tokens that may be used over the course of the run. The run will make a best effort to use only the number of completion tokens specified, across multiple turns of the run. If the run exceeds the number of completion tokens specified, the run will end with status [incomplete](#). See [incomplete_details](#) for more info.

max_prompt_tokens integer or null Optional

The maximum number of prompt tokens that may be used over the course of the run. The run will make a best effort to use only the number of prompt tokens specified, across multiple turns of the run. If the run

exceeds the number of prompt tokens specified, the run will end with status `incomplete`. See `incomplete_details` for more info.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

The ID of the [Model](#) to be used to execute this run. If a value is provided here, it will override the model associated with the assistant. If not, the model associated with the assistant will be used.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

stream boolean or null Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

temperature number or null Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

thread object Optional

Options to create a new thread. If no thread is provided when running a request, an empty thread will be created.

› Show properties

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message. `auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools before

responding to the user. Specifying a particular tool like `{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tool_resources object or null Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array or null Optional

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

› Show possible types

top_p number or null Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with `top_p` probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

truncation_strategy object or null Optional

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

› Show properties

Returns

A [run](#) object.

Default

Streaming

Streaming with Functions

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.create_and_run(
5     assistant_id="asst_abc123",
6     thread={
7         "messages": [
8             {"role": "user", "content": "Explain deep learning to a 5 year old."}
9         ]
10    }
11 )
12
13 print(run)
```

Response



```
1  {
2      "id": "run_abc123",
3      "object": "thread.run",
4      "created_at": 1699076792,
5      "assistant_id": "asst_abc123",
6      "thread_id": "thread_abc123",
7      "status": "queued",
8      "started_at": null,
9      "expires_at": 1699077392,
10     "cancelled_at": null,
11     "failed_at": null,
12     "completed_at": null,
13     "required_action": null,
14     "last_error": null,
15     "model": "gpt-4o",
16     "instructions": "You are a helpful assistant.",
17     "tools": [],
18     "tool_resources": {},
19     "metadata": {},
20     "temperature": 1.0,
21     "top_p": 1.0,
22     "max_completion_tokens": null,
23     "max_prompt_tokens": null,
24     "truncation_strategy": {
25         "type": "auto",
26         "last_messages": null
27     },
28     "incomplete_details": null,
29     "usage": null,
```

```
30     "response_format": "auto",
31     "tool_choice": "auto",
32     "parallel_tool_calls": true
33 }
```

List runs Beta

```
GET https://api.openai.com/v1/threads/{thread_id}/runs
```

Returns a list of runs belonging to a thread.

Path parameters

thread_id string Required

The ID of the thread the run belongs to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj_foo, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of [run](#) objects.

Example request

python ▾



```
1 from openai import OpenAI  
2 client = OpenAI()
```

```
3
4 runs = client.beta.threads.runs.list(
5   "thread_abc123"
6 )
7
8 print(runs)
```

Response



```
1 {
2   "object": "list",
3   "data": [
4     {
5       "id": "run_abc123",
6       "object": "thread.run",
7       "created_at": 1699075072,
8       "assistant_id": "asst_abc123",
9       "thread_id": "thread_abc123",
10      "status": "completed",
11      "started_at": 1699075072,
12      "expires_at": null,
13      "cancelled_at": null,
14      "failed_at": null,
15      "completed_at": 1699075073,
16      "last_error": null,
17      "model": "gpt-4o",
18      "instructions": null,
19      "incomplete_details": null,
```

```
20     "tools": [
21         {
22             "type": "code_interpreter"
23         }
24     ],
25     "tool_resources": {
26         "code_interpreter": {
27             "file_ids": [
28                 "file-abc123",
29                 "file-abc456"
30             ]
31         }
32     },
33     "metadata": {},
34     "usage": {
35         "prompt_tokens": 123,
36         "completion_tokens": 456,
37         "total_tokens": 579
38     },
39     "temperature": 1.0,
40     "top_p": 1.0,
41     "max_prompt_tokens": 1000,
42     "max_completion_tokens": 1000,
43     "truncation_strategy": {
44         "type": "auto",
45         "last_messages": null
46     },
47     "response_format": "auto",
48     "tool_choice": "auto",
```

```
49     "parallel_tool_calls": true
50   },
51   {
52     "id": "run_abc456",
53     "object": "thread.run",
54     "created_at": 1699063290,
55     "assistant_id": "asst_abc123",
56     "thread_id": "thread_abc123",
57     "status": "completed",
58     "started_at": 1699063290,
59     "expires_at": null,
60     "cancelled_at": null,
61     "failed_at": null,
62     "completed_at": 1699063291,
63     "last_error": null,
64     "model": "gpt-4o",
65     "instructions": null,
66     "incomplete_details": null,
67     "tools": [
68       {
69         "type": "code_interpreter"
70       }
71     ],
72     "tool_resources": {
73       "code_interpreter": {
74         "file_ids": [
75           "file-abc123",
76           "file-abc456"
77         ]
78       }
79     }
80   }
81 }
```

```
78     }
79   },
80   "metadata": {},
81   "usage": {
82     "prompt_tokens": 123,
83     "completion_tokens": 456,
84     "total_tokens": 579
85   },
86   "temperature": 1.0,
87   "top_p": 1.0,
88   "max_prompt_tokens": 1000,
89   "max_completion_tokens": 1000,
90   "truncation_strategy": {
91     "type": "auto",
92     "last_messages": null
93   },
94   "response_format": "auto",
95   "tool_choice": "auto",
96   "parallel_tool_calls": true
97 }
98 ],
99 "first_id": "run_abc123",
100 "last_id": "run_abc456",
101 "has_more": false
102 }
```

Retrieve run

Beta

```
GET https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}
```

Retrieves a run.

Path parameters

run_id string Required

The ID of the run to retrieve.

thread_id string Required

The ID of the thread that was run.

Returns

The run object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
```

```
4 run = client.beta.threads.runs.retrieve(  
5     thread_id="thread_abc123",  
6     run_id="run_abc123"  
7 )  
8  
9 print(run)
```

Response



```
1 {  
2     "id": "run_abc123",  
3     "object": "thread.run",  
4     "created_at": 1699075072,  
5     "assistant_id": "asst_abc123",  
6     "thread_id": "thread_abc123",  
7     "status": "completed",  
8     "started_at": 1699075072,  
9     "expires_at": null,  
10    "cancelled_at": null,  
11    "failed_at": null,  
12    "completed_at": 1699075073,  
13    "last_error": null,  
14    "model": "gpt-4o",  
15    "instructions": null,  
16    "incomplete_details": null,  
17    "tools": [  
18        {  
19            "type": "code_interpreter"
```

```
20      }
21    ],
22    "metadata": {},
23    "usage": {
24      "prompt_tokens": 123,
25      "completion_tokens": 456,
26      "total_tokens": 579
27    },
28    "temperature": 1.0,
29    "top_p": 1.0,
30    "max_prompt_tokens": 1000,
31    "max_completion_tokens": 1000,
32    "truncation_strategy": {
33      "type": "auto",
34      "last_messages": null
35    },
36    "response_format": "auto",
37    "tool_choice": "auto",
38    "parallel_tool_calls": true
39 }
```

Modify run

Beta

```
POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}
```

Modifies a run.

Path parameters

run_id string Required

The ID of the run to modify.

thread_id string Required

The ID of the thread that was run.

Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The modified run object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.update(
5     thread_id="thread_abc123",
6     run_id="run_abc123",
7     metadata={"user_id": "user_abc123"},
8 )
9
10 print(run)
```

Response

🔗

```
1 {
2     "id": "run_abc123",
3     "object": "thread.run",
4     "created_at": 1699075072,
5     "assistant_id": "asst_abc123",
6     "thread_id": "thread_abc123",
7     "status": "completed",
8     "started_at": 1699075072,
9     "expires_at": null,
```

```
10  "cancelled_at": null,  
11  "failed_at": null,  
12  "completed_at": 1699075073,  
13  "last_error": null,  
14  "model": "gpt-4o",  
15  "instructions": null,  
16  "incomplete_details": null,  
17  "tools": [  
18      {  
19          "type": "code_interpreter"  
20      }  
21  ],  
22  "tool_resources": {  
23      "code_interpreter": {  
24          "file_ids": [  
25              "file-abc123",  
26              "file-abc456"  
27          ]  
28      }  
29  },  
30  "metadata": {  
31      "user_id": "user_abc123"  
32  },  
33  "usage": {  
34      "prompt_tokens": 123,  
35      "completion_tokens": 456,  
36      "total_tokens": 579  
37  },  
38  "temperature": 1.0,
```

```
39 "top_p": 1.0,  
40 "max_prompt_tokens": 1000,  
41 "max_completion_tokens": 1000,  
42 "truncation_strategy": {  
43     "type": "auto",  
44     "last_messages": null  
45 },  
46 "response_format": "auto",  
47 "tool_choice": "auto",  
48 "parallel_tool_calls": true  
49 }
```

Submit tool outputs to run Beta

POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/submit_tool_outputs

When a run has the `status: "requires_action"` and `required_action.type` is `submit_tool_outputs`, this endpoint can be used to submit the outputs from the tool calls once they're all completed. All outputs must be submitted in a single request.

Path parameters

run_id string Required

The ID of the run that requires the tool output submission.

thread_id string Required

The ID of the thread to which this run belongs.

Request body

tool_outputs array Required

A list of tools for which the outputs are being submitted.

> Show properties

stream boolean Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

Returns

The modified run object matching the specified ID.

Default

Streaming

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.submit_tool_outputs(
5     thread_id="thread_123",
6     run_id="run_123",
7     tool_outputs=[
8         {
9             "tool_call_id": "call_001",
10            "output": "70 degrees and sunny."
11        }
12    ]
13 )
14
15 print(run)
```

Response

🔗

```
1 {
2     "id": "run_123",
3     "object": "thread.run",
4     "created_at": 1699075592,
5     "assistant_id": "asst_123",
6     "thread_id": "thread_123",
7     "status": "queued",
8     "started_at": 1699075592,
```

```
9     "expires_at": 1699076192,  
10    "cancelled_at": null,  
11    "failed_at": null,  
12    "completed_at": null,  
13    "last_error": null,  
14    "model": "gpt-4o",  
15    "instructions": null,  
16    "tools": [  
17      {  
18        "type": "function",  
19        "function": {  
20          "name": "get_current_weather",  
21          "description": "Get the current weather in a given location",  
22          "parameters": {  
23            "type": "object",  
24            "properties": {  
25              "location": {  
26                "type": "string",  
27                "description": "The city and state, e.g. San Francisco, CA"  
28              },  
29              "unit": {  
30                "type": "string",  
31                "enum": ["celsius", "fahrenheit"]  
32              }  
33            },  
34            "required": ["location"]  
35          }  
36        }  
37      }
```

```
38 ],
39 "metadata": {},
40 "usage": null,
41 "temperature": 1.0,
42 "top_p": 1.0,
43 "max_prompt_tokens": 1000,
44 "max_completion_tokens": 1000,
45 "truncation_strategy": {
46     "type": "auto",
47     "last_messages": null
48 },
49 "response_format": "auto",
50 "tool_choice": "auto",
51 "parallel_tool_calls": true
52 }
```

Cancel a run Beta

POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/cancel

Cancels a run that is `in_progress`.

Path parameters

run_id string Required

The ID of the run to cancel.

thread_id string Required

The ID of the thread to which this run belongs.

Returns

The modified [run](#) object matching the specified ID.

Example request

python ▼ Copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.cancel(
5     thread_id="thread_abc123",
6     run_id="run_abc123"
7 )
8
9 print(run)
```

Response



```
1  {
2      "id": "run_abc123",
3      "object": "thread.run",
4      "created_at": 1699076126,
5      "assistant_id": "asst_abc123",
6      "thread_id": "thread_abc123",
7      "status": "cancelling",
8      "started_at": 1699076126,
9      "expires_at": 1699076726,
10     "cancelled_at": null,
11     "failed_at": null,
12     "completed_at": null,
13     "last_error": null,
14     "model": "gpt-4o",
15     "instructions": "You summarize books.",
16     "tools": [
17         {
18             "type": "file_search"
19         }
20     ],
21     "tool_resources": {
22         "file_search": {
23             "vector_store_ids": ["vs_123"]
24         }
25     },
26     "metadata": {},
27     "usage": null,
```

```
28     "temperature": 1.0,  
29     "top_p": 1.0,  
30     "response_format": "auto",  
31     "tool_choice": "auto",  
32     "parallel_tool_calls": true  
33 }
```

The run object Beta

Represents an execution run on a [thread](#).

assistant_id string

The ID of the [assistant](#) used for execution of this run.

cancelled_at integer or null

The Unix timestamp (in seconds) for when the run was cancelled.

completed_at integer or null

The Unix timestamp (in seconds) for when the run was completed.

created_at integer

The Unix timestamp (in seconds) for when the run was created.

expires_at integer or null

The Unix timestamp (in seconds) for when the run will expire.

failed_at integer or null

The Unix timestamp (in seconds) for when the run failed.

id string

The identifier, which can be referenced in API endpoints.

incomplete_details object or null

Details on why the run is incomplete. Will be `null` if the run is not incomplete.

› Show properties

instructions string

The instructions that the assistant used for this run.

last_error object or null

The last error associated with this run. Will be `null` if there are no errors.

› Show properties

max_completion_tokens integer or null

The maximum number of completion tokens specified to have been used over the course of the run.

max_prompt_tokens integer or null

The maximum number of prompt tokens specified to have been used over the course of the run.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

The model that the assistant used for this run.

object string

The object type, which is always `thread.run`.

parallel_tool_calls boolean

Whether to enable parallel function calling during tool use.

required_action object or null

Details on the action required to continue the run. Will be `null` if no action is required.

› Show properties

response_format "auto" or object

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

started_at integer or null

The Unix timestamp (in seconds) for when the run was started.

status string

The status of the run, which can be either `queued`, `in_progress`, `requires_action`, `cancelling`, `cancelled`, `failed`, `completed`, `incomplete`, or `expired`.

temperature number or null

The sampling temperature used for this run. If not set, defaults to 1.

thread_id string

The ID of the thread that was executed on as a part of this run.

tool_choice string or object

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message. `auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools before responding to the user. Specifying a particular tool like `{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tools array

The list of tools that the assistant used for this run.

› Show possible types

top_p number or null

The nucleus sampling value used for this run. If not set, defaults to 1.

truncation_strategy object or null

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

› Show properties

usage object

Usage statistics related to the run. This value will be `null` if the run is not in a terminal state (i.e. `in_progress`, `queued`, etc.).

› Show properties

OBJECT The run object



```
1  {
2    "id": "run_abc123",
3    "object": "thread.run",
4    "created_at": 1698107661,
5    "assistant_id": "asst_abc123",
6    "thread_id": "thread_abc123",
7    "status": "completed",
8    "started_at": 1699073476,
9    "expires_at": null,
10   "cancelled_at": null,
11   "failed_at": null,
12   "completed_at": 1699073498,
13   "last_error": null,
14   "model": "gpt-4o",
15   "instructions": null,
16   "tools": [{"type": "file_search"}, {"type": "code_interpreter"}],
17   "metadata": {},
18   "incomplete_details": null,
19   "usage": {
```

```
20     "prompt_tokens": 123,  
21     "completion_tokens": 456,  
22     "total_tokens": 579  
23 },  
24     "temperature": 1.0,  
25     "top_p": 1.0,  
26     "max_prompt_tokens": 1000,  
27     "max_completion_tokens": 1000,  
28     "truncation_strategy": {  
29         "type": "auto",  
30         "last_messages": null  
31     },  
32     "response_format": "auto",  
33     "tool_choice": "auto",  
34     "parallel_tool_calls": true  
35 }
```

PREVIOUS
< **Messages**

NEXT
Run steps >

Runs

Beta



Represents an execution run on a thread.

Related guide: [Assistants](#)

Create run

Beta



```
POST https://api.openai.com/v1/threads/{thread_id}/runs
```

Create a run.

Path parameters

thread_id string Required

The ID of the thread to run.

Query parameters

include[] array Optional

A list of additional fields to include in the response. Currently the only supported value is

`step_details.tool_calls[*].file_search.results[*].content` to fetch the file search result content.

See the [file search tool documentation](#) for more information.

Request body

assistant_id string Required

The ID of the [assistant](#) to use to execute this run.

additional_instructions string or null Optional

Appends additional instructions at the end of the instructions for the run. This is useful for modifying the behavior on a per-run basis without overriding other instructions.

additional_messages array or null Optional

Adds additional messages to the thread before creating the run.

› Show properties

instructions string or null Optional

Overrides the [instructions](#) of the assistant. This is useful for modifying the behavior on a per-run basis.

max_completion_tokens integer or null Optional

The maximum number of completion tokens that may be used over the course of the run. The run will make a best effort to use only the number of completion tokens specified, across multiple turns of the run. If the run exceeds the number of completion tokens specified, the run will end with status `incomplete`. See `incomplete_details` for more info.

max_prompt_tokens integer or null Optional

The maximum number of prompt tokens that may be used over the course of the run. The run will make a best effort to use only the number of prompt tokens specified, across multiple turns of the run. If the run exceeds the number of prompt tokens specified, the run will end with status `incomplete`. See `incomplete_details` for more info.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

The ID of the [Model](#) to be used to execute this run. If a value is provided here, it will override the model associated with the assistant. If not, the model associated with the assistant will be used.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

`gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in gpt-5.1.

All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.

The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.

`xhigh` is supported for all models after `gpt-5.1-codex-max`.

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

stream boolean or null Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

temperature number or null Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message.

`auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required`

means the model must call one or more tools before responding to the user. Specifying a particular tool like

`{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

> Show possible types

tools array or null Optional

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

> Show possible types

top_p number or null Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with `top_p` probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

truncation_strategy object or null Optional

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

> Show properties

Returns

A run object.

Default

Streaming

Streaming with Functions

Example request

```
from openai import OpenAI
client = OpenAI()

stream = client.beta.threads.runs.create(
    thread_id="thread_123",
    assistant_id="asst_123",
    stream=True
)

for event in stream:
    print(event)
```

Response

```
event: thread.run.created
data: {"id": "run_123", "object": "thread.run", "created_at": 1710330640, "assistant_id": "asst_123", "tI

event: thread.run.queued
data: {"id": "run_123", "object": "thread.run", "created_at": 1710330640, "assistant_id": "asst_123", "tI
```

```
event: thread.run.in_progress
data: {"id": "run_123", "object": "thread.run", "created_at": 1710330640, "assistant_id": "asst_123", "tli": 1, "type": "run"}
```

```
event: thread.run.step.created
data: {"id": "step_001", "object": "thread.run.step", "created_at": 1710330641, "run_id": "run_123", "assistant_id": "asst_123", "type": "step"}
```

```
event: thread.run.step.in_progress
data: {"id": "step_001", "object": "thread.run.step", "created_at": 1710330641, "run_id": "run_123", "assistant_id": "asst_123", "type": "step"}
```

```
event: thread.message.created
data: {"id": "msg_001", "object": "thread.message", "created_at": 1710330641, "assistant_id": "asst_123", "type": "message"}
```

```
event: thread.message.in_progress
data: {"id": "msg_001", "object": "thread.message", "created_at": 1710330641, "assistant_id": "asst_123", "type": "message"}
```

```
event: thread.message.delta
data: {"id": "msg_001", "object": "thread.message.delta", "delta": {"content": [{"index": 0, "type": "text", "text": "Hello, world!"}], "type": "delta"}}
```

```
...
```

```
event: thread.message.delta
data: {"id": "msg_001", "object": "thread.message.delta", "delta": {"content": [{"index": 0, "type": "text", "text": "Hello, world!"}], "type": "delta"}}
```

```
event: thread.message.delta
data: {"id": "msg_001", "object": "thread.message.delta", "delta": {"content": [{"index": 0, "type": "text", "text": "Hello, world!"}], "type": "delta"}}
```

```
event: thread.message.completed
data: {"id": "msg_001", "object": "thread.message", "created_at": 1710330641, "assistant_id": "asst_123", "type": "message"}
```

```
event: thread.run.step.completed
data: {"id": "step_001", "object": "thread.run.step", "created_at": 1710330641, "run_id": "run_123", "ass
event: thread.run.completed
data: {"id": "run_123", "object": "thread.run", "created_at": 1710330640, "assistant_id": "asst_123", "ti
event: done
data: [DONE]
```

Create thread and run Beta



POST <https://api.openai.com/v1/threads/runs>

Create a thread and run it in one request.

Request body

assistant_id string Required

The ID of the assistant to use to execute this run.

instructions string or null Optional

Override the default system message of the assistant. This is useful for modifying the behavior on a per-run basis.

max_completion_tokens integer or null Optional

The maximum number of completion tokens that may be used over the course of the run. The run will make a best effort to use only the number of completion tokens specified, across multiple turns of the run. If the run exceeds the number of completion tokens specified, the run will end with status `incomplete`. See `incomplete_details` for more info.

max_prompt_tokens integer or null Optional

The maximum number of prompt tokens that may be used over the course of the run. The run will make a best effort to use only the number of prompt tokens specified, across multiple turns of the run. If the run exceeds the number of prompt tokens specified, the run will end with status `incomplete`. See `incomplete_details` for more info.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

The ID of the [Model](#) to be used to execute this run. If a value is provided here, it will override the model associated with the assistant. If not, the model associated with the assistant will be used.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since [gpt-3.5-turbo-1106](#).

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

stream boolean or null Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

temperature number or null Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

thread object Optional

Options to create a new thread. If no thread is provided when running a request, an empty thread will be created.

› Show properties

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message.

`auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required`

means the model must call one or more tools before responding to the user. Specifying a particular tool like

`{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

> Show possible types

tool_resources object or null Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the

`code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

> Show properties

tools array or null Optional

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

> Show possible types

top_p number or null Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

truncation_strategy object or null Optional

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

> Show properties

Returns

A run object.

Default **Streaming** **Streaming with Functions**

Example request

```
from openai import OpenAI
client = OpenAI()

run = client.beta.threads.create_and_run(
    assistant_id="asst_abc123",
    thread={
        "messages": [
            {"role": "user", "content": "Explain deep learning to a 5 year old."}
        ]
    }
)
print(run)
```

Response

```
{  
  "id": "run_abc123",  
  "object": "thread.run",  
  "created_at": 1699076792,  
  "assistant_id": "asst_abc123",  
  "thread_id": "thread_abc123",  
  "status": "queued",  
  "started_at": null,  
  "expires_at": 1699077392,  
  "cancelled_at": null,  
  "failed_at": null,  
  "completed_at": null,  
  "required_action": null,  
  "last_error": null,  
  "model": "gpt-4o",  
  "instructions": "You are a helpful assistant.",  
  "tools": [],  
  "tool_resources": {},  
  "metadata": {},  
  "temperature": 1.0,  
  "top_p": 1.0,  
  "max_completion_tokens": null,  
  "max_prompt_tokens": null,  
  "truncation_strategy": {  
    "type": "auto",  
    "last_messages": null  
  },  
  "incomplete_details": null,  
  "usage": null,  
}
```

```
"response_format": "auto",
"tool_choice": "auto",
"parallel_tool_calls": true
}
```

List runs Beta



GET https://api.openai.com/v1/threads/{thread_id}/runs

Returns a list of runs belonging to a thread.

Path parameters

thread_id string Required

The ID of the thread the run belongs to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with `obj_foo`, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with `obj_foo`, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of run objects.

Example request

```
from openai import OpenAI
client = OpenAI()

runs = client.beta.threads.runs.list(
    "thread_abc123"
)
```

```
print(runs)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "run_abc123",
      "object": "thread.run",
      "created_at": 1699075072,
      "assistant_id": "asst_abc123",
      "thread_id": "thread_abc123",
      "status": "completed",
      "started_at": 1699075072,
      "expires_at": null,
      "cancelled_at": null,
      "failed_at": null,
      "completed_at": 1699075073,
      "last_error": null,
      "model": "gpt-4o",
      "instructions": null,
      "incomplete_details": null,
      "tools": [
        {
          "type": "code_interpreter"
        }
      ],
    }
  ]
}
```

```
"tool_resources": {  
    "code_interpreter": {  
        "file_ids": [  
            "file-abc123",  
            "file-abc456"  
        ]  
    }  
},  
"metadata": {},  
"usage": {  
    "prompt_tokens": 123,  
    "completion_tokens": 456,  
    "total_tokens": 579  
},  
"temperature": 1.0,  
"top_p": 1.0,  
"max_prompt_tokens": 1000,  
"max_completion_tokens": 1000,  
"truncation_strategy": {  
    "type": "auto",  
    "last_messages": null  
},  
"response_format": "auto",  
"tool_choice": "auto",  
"parallel_tool_calls": true  
},  
{  
    "id": "run_abc456",  
    "object": "thread.run",  
}
```

```
"created_at": 1699063290,  
"assistant_id": "asst_abc123",  
"thread_id": "thread_abc123",  
"status": "completed",  
"started_at": 1699063290,  
"expires_at": null,  
"cancelled_at": null,  
"failed_at": null,  
"completed_at": 1699063291,  
"last_error": null,  
"model": "gpt-4o",  
"instructions": null,  
"incomplete_details": null,  
"tools": [  
  {  
    "type": "code_interpreter"  
  }  
,  
  {"  
    "tool_resources": {  
      "code_interpreter": {  
        "file_ids": [  
          "file-abc123",  
          "file-abc456"  
        ]  
      }  
    },  
    "metadata": {},  
    "usage": {  
      "prompt_tokens": 123,  
      "completion_tokens": 50,  
      "total_tokens": 173  
    }  
  }  
],  
"run_type": "code_interpreter",  
"run_version": "v1",  
"run_status": "completed",  
"run_start_time": 1699063290,  
"run_end_time": 1699063291,  
"run_expires_time": null,  
"run_cancelled_time": null,  
"run_failed_time": null,  
"run_completed_time": 1699063291,  
"run_last_error": null,  
"run_model": "gpt-4o",  
"run_instructions": null,  
"run_incomplete_details": null,  
"run_tools": [{"type": "code_interpreter"}],  
"run_tool_resources": {"code_interpreter": {"file_ids": ["file-abc123", "file-abc456"]}},  
"run_metadata": {},  
"run_usage": {"prompt_tokens": 123, "completion_tokens": 50, "total_tokens": 173},  
"run_files": [{"id": "file-abc123", "type": "code_interpreter"}, {"id": "file-abc456", "type": "code_interpreter"}],  
"run_logs": [{"log": "Starting run with model gpt-4o."}, {"log": "Running code interpreter tool."}, {"log": "Completed run with status completed."}],  
"run_events": [{"event": "run_start", "time": 1699063290, "message": "Starting run with model gpt-4o."}, {"event": "run_tool_start", "time": 1699063290, "message": "Running code interpreter tool."}, {"event": "run_tool_end", "time": 1699063291, "message": "Tool completed successfully."}, {"event": "run_end", "time": 1699063291, "message": "Completed run with status completed."}]
```

```
        "completion_tokens": 456,  
        "total_tokens": 579  
    },  
    "temperature": 1.0,  
    "top_p": 1.0,  
    "max_prompt_tokens": 1000,  
    "max_completion_tokens": 1000,  
    "truncation_strategy": {  
        "type": "auto",  
        "last_messages": null  
    },  
    "response_format": "auto",  
    "tool_choice": "auto",  
    "parallel_tool_calls": true  
}  
],  
"first_id": "run_abc123",  
"last_id": "run_abc456",  
"has_more": false  
}
```

Retrieve run Beta

⌚

```
GET https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}
```

Retrieves a run.

Path parameters

run_id string Required

The ID of the run to retrieve.

thread_id string Required

The ID of the thread that was run.

Returns

The run object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

run = client.beta.threads.runs.retrieve(
    thread_id="thread_abc123",
    run_id="run_abc123"
)
```

```
print(run)
```

Response

```
{  
    "id": "run_abc123",  
    "object": "thread.run",  
    "created_at": 1699075072,  
    "assistant_id": "asst_abc123",  
    "thread_id": "thread_abc123",  
    "status": "completed",  
    "started_at": 1699075072,  
    "expires_at": null,  
    "cancelled_at": null,  
    "failed_at": null,  
    "completed_at": 1699075073,  
    "last_error": null,  
    "model": "gpt-4o",  
    "instructions": null,  
    "incomplete_details": null,  
    "tools": [  
        {  
            "type": "code_interpreter"  
        }  
    ],  
    "metadata": {},  
    "usage": {  
        "prompt_tokens": 123,  
        "completion_tokens": 100,  
        "total_tokens": 223  
    },  
    "error": null  
}
```

```
"completion_tokens": 456,  
"total_tokens": 579  
,  
"temperature": 1.0,  
"top_p": 1.0,  
"max_prompt_tokens": 1000,  
"max_completion_tokens": 1000,  
"truncation_strategy": {  
    "type": "auto",  
    "last_messages": null  
,  
"response_format": "auto",  
"tool_choice": "auto",  
"parallel_tool_calls": true  
}
```

Modify run Beta

ⓘ

POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}

Modifies a run.

Path parameters

run_id string Required

The ID of the run to modify.

thread_id string Required

The ID of the thread that was run.

Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The modified run object matching the specified ID.

Example request

```
from openai import OpenAI  
client = OpenAI()
```

```
run = client.beta.threads.runs.update(  
    thread_id="thread_abc123",  
    run_id="run_abc123",  
    metadata={"user_id": "user_abc123"},  
)  
  
print(run)
```

Response

```
{  
    "id": "run_abc123",  
    "object": "thread.run",  
    "created_at": 1699075072,  
    "assistant_id": "asst_abc123",  
    "thread_id": "thread_abc123",  
    "status": "completed",  
    "started_at": 1699075072,  
    "expires_at": null,  
    "cancelled_at": null,  
    "failed_at": null,  
    "completed_at": 1699075073,  
    "last_error": null,  
    "model": "gpt-4o",  
    "instructions": null,  
    "incomplete_details": null,  
    "tools": [  
        {  
            "type": "code_interpreter"
```

```
        },
      ],
      "tool_resources": {
        "code_interpreter": {
          "file_ids": [
            "file-abc123",
            "file-abc456"
          ]
        }
      },
      "metadata": {
        "user_id": "user_abc123"
      },
      "usage": {
        "prompt_tokens": 123,
        "completion_tokens": 456,
        "total_tokens": 579
      },
      "temperature": 1.0,
      "top_p": 1.0,
      "max_prompt_tokens": 1000,
      "max_completion_tokens": 1000,
      "truncation_strategy": {
        "type": "auto",
        "last_messages": null
      },
      "response_format": "auto",
      "tool_choice": "auto",
```

```
"parallel_tool_calls": true
```

Submit tool outputs to run Beta



```
POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/submit_tool_outputs
```

When a run has the `status: "requires_action"` and `required_action.type` is `submit_tool_outputs`, this endpoint can be used to submit the outputs from the tool calls once they're all completed. All outputs must be submitted in a single request.

Path parameters

run_id string Required

The ID of the run that requires the tool output submission.

thread_id string Required

The ID of the thread to which this run belongs.

Request body

tool_outputs array Required

A list of tools for which the outputs are being submitted.

> Show properties

stream boolean Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

Returns

The modified `run` object matching the specified ID.

Default Streaming

Example request

```
from openai import OpenAI
client = OpenAI()

run = client.beta.threads.runs.submit_tool_outputs(
    thread_id="thread_123",
    run_id="run_123",
    tool_outputs=[
        {
            "tool_call_id": "call_001",
            "output": "70 degrees and sunny."
```

```
    }
]
)

print(run)
```

Response

```
{
  "id": "run_123",
  "object": "thread.run",
  "created_at": 1699075592,
  "assistant_id": "asst_123",
  "thread_id": "thread_123",
  "status": "queued",
  "started_at": 1699075592,
  "expires_at": 1699076192,
  "cancelled_at": null,
  "failed_at": null,
  "completed_at": null,
  "last_error": null,
  "model": "gpt-4o",
  "instructions": null,
  "tools": [
    {
      "type": "function",
      "function": {
        "name": "get_current_weather",
        "description": "Get the current weather in a given location",
      }
    }
  ]
}
```

```
"parameters": {
    "type": "object",
    "properties": {
        "location": {
            "type": "string",
            "description": "The city and state, e.g. San Francisco, CA"
        },
        "unit": {
            "type": "string",
            "enum": ["celsius", "fahrenheit"]
        }
    },
    "required": ["location"]
},
],
"metadata": {},
"usage": null,
"temperature": 1.0,
"top_p": 1.0,
"max_prompt_tokens": 1000,
"max_completion_tokens": 1000,
"truncation_strategy": {
    "type": "auto",
    "last_messages": null
},
"response_format": "auto",
"tool_choice": "auto",
```

```
"parallel_tool_calls": true  
}
```

Cancel a run

Beta



```
POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/cancel
```

Cancels a run that is `in_progress`.

Path parameters

run_id string Required

The ID of the run to cancel.

thread_id string Required

The ID of the thread to which this run belongs.

Returns

The modified `run` object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

run = client.beta.threads.runs.cancel(
    thread_id="thread_abc123",
    run_id="run_abc123"
)

print(run)
```

Response

```
{  
    "id": "run_abc123",  
    "object": "thread.run",  
    "created_at": 1699076126,  
    "assistant_id": "asst_abc123",  
    "thread_id": "thread_abc123",  
    "status": "cancelling",  
    "started_at": 1699076126,  
    "expires_at": 1699076726,  
    "cancelled_at": null,  
    "failed_at": null,  
    "completed_at": null,  
    "last_error": null,  
    "model": "gpt-4o",  
}
```

```
"instructions": "You summarize books.",  
"tools": [  
  {  
    "type": "file_search"  
  }  
],  
"tool_resources": {  
  "file_search": {  
    "vector_store_ids": ["vs_123"]  
  }  
},  
"metadata": {},  
"usage": null,  
"temperature": 1.0,  
"top_p": 1.0,  
"response_format": "auto",  
"tool_choice": "auto",  
"parallel_tool_calls": true  
}
```

The run object Beta

🔗

Represents an execution run on a thread.

assistant_id string

The ID of the [assistant](#) used for execution of this run.

cancelled_at integer or null

The Unix timestamp (in seconds) for when the run was cancelled.

completed_at integer or null

The Unix timestamp (in seconds) for when the run was completed.

created_at integer

The Unix timestamp (in seconds) for when the run was created.

expires_at integer or null

The Unix timestamp (in seconds) for when the run will expire.

failed_at integer or null

The Unix timestamp (in seconds) for when the run failed.

id string

The identifier, which can be referenced in API endpoints.

incomplete_details object or null

Details on why the run is incomplete. Will be `null` if the run is not incomplete.

> Show properties

instructions string

The instructions that the assistant used for this run.

last_error object or null

The last error associated with this run. Will be `null` if there are no errors.

› Show properties

max_completion_tokens integer or null

The maximum number of completion tokens specified to have been used over the course of the run.

max_prompt_tokens integer or null

The maximum number of prompt tokens specified to have been used over the course of the run.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

The model that the assistant used for this run.

object string

The object type, which is always `thread.run`.

parallel_tool_calls boolean

Whether to enable parallel function calling during tool use.

required_action object or null

Details on the action required to continue the run. Will be `null` if no action is required.

› Show properties

response_format "auto" or object

Specifies the format that the model must output. Compatible with GPT-4o, GPT-4 Turbo, and all GPT-3.5 Turbo models since

`gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the Structured Outputs guide.

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

started_at integer or null

The Unix timestamp (in seconds) for when the run was started.

status string

The status of the run, which can be either `queued`, `in_progress`, `requires_action`, `cancelling`, `cancelled`, `failed`, `completed`, `incomplete`, or `expired`.

temperature number or null

The sampling temperature used for this run. If not set, defaults to 1.

thread_id string

The ID of the thread that was executed on as a part of this run.

tool_choice string or object

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message.

`auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required`

means the model must call one or more tools before responding to the user. Specifying a particular tool like

`{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tools array

The list of tools that the assistant used for this run.

› Show possible types

top_p number or null

The nucleus sampling value used for this run. If not set, defaults to 1.

truncation_strategy object or null

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

> Show properties

usage object

Usage statistics related to the run. This value will be `null` if the run is not in a terminal state (i.e. `in_progress`, `queued`, etc.).

> Show properties

OBJECT The run object

```
{  
  "id": "run_abc123",  
  "object": "thread.run",  
  "created_at": 1698107661,  
  "assistant_id": "asst_abc123",  
  "thread_id": "thread_abc123",  
  "status": "completed",  
  "started_at": 1699073476,  
  "expires_at": null,  
  "cancelled_at": null,  
  "failed_at": null,  
  "completed_at": 1699073498,  
  "last_error": null,  
  "model": "gpt-4o",  
  "instructions": null,  
  "tools": [{"type": "file_search"}, {"type": "code_interpreter"}],  
  "metadata": {},  
  "incomplete_details": null,  
  "usage": {  
    "prompt_tokens": 123,  
    "completion_tokens": 100,  
    "total_tokens": 223,  
    "prompt_cost": 0.0123,  
    "completion_cost": 0.0100,  
    "total_cost": 0.0223  
  }  
}
```

```
"completion_tokens": 456,  
"total_tokens": 579  
,  
"temperature": 1.0,  
"top_p": 1.0,  
"max_prompt_tokens": 1000,  
"max_completion_tokens": 1000,  
"truncation_strategy": {  
    "type": "auto",  
    "last_messages": null  
,  
"response_format": "auto",  
"tool_choice": "auto",  
"parallel_tool_calls": true  
}
```

< PREVIOUS
Messages

NEXT

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Runs Beta

Represents an execution run on a thread.

Related guide: [Assistants](#)

Create run Beta

```
POST https://api.openai.com/v1/threads/{thread_id}/runs
```

Create a run.

Path parameters

thread_id string Required

The ID of the thread to run.

Query parameters

include[] array Optional

A list of additional fields to include in the response. Currently the only supported value is

`step_details.tool_calls[*].file_search.results[*].content` to fetch the file search result content.

See the [file search tool documentation](#) for more information.

Request body

assistant_id string Required

The ID of the [assistant](#) to use to execute this run.

additional_instructions string or null Optional

Appends additional instructions at the end of the instructions for the run. This is useful for modifying the behavior on a per-run basis without overriding other instructions.

additional_messages array or null Optional

Adds additional messages to the thread before creating the run.

› Show properties

instructions string or null Optional

Overrides the [instructions](#) of the assistant. This is useful for modifying the behavior on a per-run basis.

max_completion_tokens integer or null Optional

The maximum number of completion tokens that may be used over the course of the run. The run will make a best effort to use only the number of completion tokens specified, across multiple turns of the run. If the run exceeds the number of completion tokens specified, the run will end with status [incomplete](#). See [incomplete_details](#) for more info.

max_prompt_tokens integer or null Optional

The maximum number of prompt tokens that may be used over the course of the run. The run will make a best effort to use only the number of prompt tokens specified, across multiple turns of the run. If the run exceeds the number of prompt tokens specified, the run will end with status [incomplete](#). See [incomplete_details](#) for more info.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

The ID of the [Model](#) to be used to execute this run. If a value is provided here, it will override the model associated with the assistant. If not, the model associated with the assistant will be used.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

reasoning_effort string Optional Defaults to medium

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `none`, `minimal`, `low`, `medium`, `high`, and `xhigh`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

- `gpt-5.1` defaults to `none`, which does not perform reasoning. The supported reasoning values for `gpt-5.1` are `none`, `low`, `medium`, and `high`. Tool calls are supported for all reasoning values in `gpt-5.1`.
 - All models before `gpt-5.1` default to `medium` reasoning effort, and do not support `none`.
 - The `gpt-5-pro` model defaults to (and only supports) `high` reasoning effort.
 - `xhigh` is supported for all models after `gpt-5.1-codex-max`.
-

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": {...} }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

stream boolean or null Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

temperature number or null Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message. `auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools before responding to the user. Specifying a particular tool like `{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tools array or null Optional

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

› Show possible types

top_p number or null Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

truncation_strategy object or null Optional

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

› Show properties

Returns

A [run](#) object.

Default

Streaming

Streaming with Functions

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 tools = [
5     {
6         "type": "function",
7         "function": {
8             "name": "get_current_weather",
9             "description": "Get the current weather in a given location",
10            "parameters": {
11                "type": "object",
12                "properties": {
13                    "location": {
14                        "type": "string",
15                        "description": "The city and state, e.g. San Francisco, CA",
16                    },
17                    "unit": {"type": "string", "enum": ["celsius", "fahrenheit"]},
18                },
19                "required": ["location"],
20            },
21        },
22    }
23 ]
24
25 stream = client.beta.threads.runs.create(
26     thread_id="thread_abc123",
27     assistant_id="asst_abc123",
```

```
28     tools=tools,  
29     stream=True  
30 )  
31  
32 for event in stream:  
33     print(event)
```

Response



```
1 event: thread.run.created  
2 data: {"id": "run_123", "object": "thread.run", "created_at": 1710348075, "assistant_id": ''  
3  
4 event: thread.run.queued  
5 data: {"id": "run_123", "object": "thread.run", "created_at": 1710348075, "assistant_id": ''  
6  
7 event: thread.run.in_progress  
8 data: {"id": "run_123", "object": "thread.run", "created_at": 1710348075, "assistant_id": ''  
9  
10 event: thread.run.step.created  
11 data: {"id": "step_001", "object": "thread.run.step", "created_at": 1710348076, "run_id": ''  
12  
13 event: thread.run.step.in_progress  
14 data: {"id": "step_001", "object": "thread.run.step", "created_at": 1710348076, "run_id": ''  
15  
16 event: thread.message.created  
17 data: {"id": "msg_001", "object": "thread.message", "created_at": 1710348076, "assistant_id": ''  
18  
19 event: thread.message.in_progress
```

```
20 data: {"id": "msg_001", "object": "thread.message", "created_at": 1710348076, "assistant_id": "assistant_123", "content": "Hello, how can I help you today?"}
21
22 event: thread.message.delta
23 data: {"id": "msg_001", "object": "thread.message.delta", "delta": {"content": [{"index": 0, "text": "Hello, how can I help you today?"}], "delta_type": "content"}, "version": 1}
24
25 ...
26
27 event: thread.message.delta
28 data: {"id": "msg_001", "object": "thread.message.delta", "delta": {"content": [{"index": 0, "text": "Hello, how can I help you today?"}], "delta_type": "content"}, "version": 2}
29
30 event: thread.message.delta
31 data: {"id": "msg_001", "object": "thread.message.delta", "delta": {"content": [{"index": 0, "text": "Hello, how can I help you today?"}], "delta_type": "content"}, "version": 3}
32
33 event: thread.message.completed
34 data: {"id": "msg_001", "object": "thread.message", "created_at": 1710348076, "assistant_id": "assistant_123", "content": "Hello, how can I help you today?", "version": 3}
35
36 event: thread.run.step.completed
37 data: {"id": "step_001", "object": "thread.run.step", "created_at": 1710348076, "run_id": "run_123", "status": "completed", "version": 1}
38
39 event: thread.run.completed
40 data: {"id": "run_123", "object": "thread.run", "created_at": 1710348075, "assistant_id": "assistant_123", "content": "The run has completed successfully.", "version": 1}
41
42 event: done
43 data: [DONE]
```

Create thread and run

Beta

POST <https://api.openai.com/v1/threads/runs>

Create a thread and run it in one request.

Request body

assistant_id string Required

The ID of the [assistant](#) to use to execute this run.

instructions string or null Optional

Override the default system message of the assistant. This is useful for modifying the behavior on a per-run basis.

max_completion_tokens integer or null Optional

The maximum number of completion tokens that may be used over the course of the run. The run will make a best effort to use only the number of completion tokens specified, across multiple turns of the run. If the run exceeds the number of completion tokens specified, the run will end with status [incomplete](#). See [incomplete_details](#) for more info.

max_prompt_tokens integer or null Optional

The maximum number of prompt tokens that may be used over the course of the run. The run will make a best effort to use only the number of prompt tokens specified, across multiple turns of the run. If the run exceeds the number of prompt tokens specified, the run will end with status `incomplete`. See `incomplete_details` for more info.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string Optional

The ID of the [Model](#) to be used to execute this run. If a value is provided here, it will override the model associated with the assistant. If not, the model associated with the assistant will be used.

parallel_tool_calls boolean Optional Defaults to true

Whether to enable [parallel function calling](#) during tool use.

response_format "auto" or object Optional

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

stream boolean or null Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

temperature number or null Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

thread object Optional

Options to create a new thread. If no thread is provided when running a request, an empty thread will be created.

› Show properties

tool_choice string or object Optional

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message. `auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools before responding to the user. Specifying a particular tool like `{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tool_resources object or null Optional

A set of resources that are used by the assistant's tools. The resources are specific to the type of tool. For example, the `code_interpreter` tool requires a list of file IDs, while the `file_search` tool requires a list of vector store IDs.

› Show properties

tools array or null Optional

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

› Show possible types

top_p number or null Optional Defaults to 1

An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with `top_p` probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered.

We generally recommend altering this or temperature but not both.

truncation_strategy object or null Optional

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

› Show properties

Returns

A [run](#) object.

Default Streaming Streaming with Functions

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.create_and_run(
5     assistant_id="asst_abc123",
6     thread={
7         "messages": [
8             {"role": "user", "content": "Explain deep learning to a 5 year old."}
9         ]
10    }
11 )
12
13
```

```
print(run)
```

Response



```
1  {
2      "id": "run_abc123",
3      "object": "thread.run",
4      "created_at": 1699076792,
5      "assistant_id": "asst_abc123",
6      "thread_id": "thread_abc123",
7      "status": "queued",
8      "started_at": null,
9      "expires_at": 1699077392,
10     "cancelled_at": null,
11     "failed_at": null,
12     "completed_at": null,
13     "required_action": null,
14     "last_error": null,
15     "model": "gpt-4o",
16     "instructions": "You are a helpful assistant.",
17     "tools": [],
18     "tool_resources": {},
19     "metadata": {},
20     "temperature": 1.0,
21     "top_p": 1.0,
22     "max_completion_tokens": null,
23     "max_prompt_tokens": null,
```

```
24 "truncation_strategy": {  
25     "type": "auto",  
26     "last_messages": null  
27 },  
28 "incomplete_details": null,  
29 "usage": null,  
30 "response_format": "auto",  
31 "tool_choice": "auto",  
32 "parallel_tool_calls": true  
33 }
```

List runs Beta

```
GET https://api.openai.com/v1/threads/{thread_id}/runs
```

Returns a list of runs belonging to a thread.

Path parameters

thread_id string Required

The ID of the thread the run belongs to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with `obj_foo`, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with `obj_foo`, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of run objects.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 runs = client.beta.threads.runs.list(
5     "thread_abc123"
6 )
7
8 print(runs)
```

Response

🔗

```
1 {
2     "object": "list",
3     "data": [
4         {
5             "id": "run_abc123",
6             "object": "thread.run",
7             "created_at": 1699075072,
8             "assistant_id": "asst_abc123",
9             "thread_id": "thread_abc123",
10            "status": "completed",
11            "started_at": 1699075072,
12            "expires_at": null,
13            "cancelled_at": null,
14            "failed_at": null,
15            "completed_at": 1699075073,
```

```
16     "last_error": null,
17     "model": "gpt-4o",
18     "instructions": null,
19     "incomplete_details": null,
20     "tools": [
21       {
22         "type": "code_interpreter"
23       }
24     ],
25     "tool_resources": {
26       "code_interpreter": {
27         "file_ids": [
28           "file-abc123",
29           "file-abc456"
30         ]
31       }
32     },
33     "metadata": {},
34     "usage": {
35       "prompt_tokens": 123,
36       "completion_tokens": 456,
37       "total_tokens": 579
38     },
39     "temperature": 1.0,
40     "top_p": 1.0,
41     "max_prompt_tokens": 1000,
42     "max_completion_tokens": 1000,
43     "truncation_strategy": {
44       "type": "auto",
```

```
45     "last_messages": null
46   },
47   "response_format": "auto",
48   "tool_choice": "auto",
49   "parallel_tool_calls": true
50 },
51 {
52   "id": "run_abc456",
53   "object": "thread.run",
54   "created_at": 1699063290,
55   "assistant_id": "asst_abc123",
56   "thread_id": "thread_abc123",
57   "status": "completed",
58   "started_at": 1699063290,
59   "expires_at": null,
60   "cancelled_at": null,
61   "failed_at": null,
62   "completed_at": 1699063291,
63   "last_error": null,
64   "model": "gpt-4o",
65   "instructions": null,
66   "incomplete_details": null,
67   "tools": [
68     {
69       "type": "code_interpreter"
70     }
71   ],
72   "tool_resources": {
73     "code_interpreter": {
```

```
74         "file_ids": [
75             "file-abc123",
76             "file-abc456"
77         ]
78     }
79 },
80 "metadata": {},
81 "usage": {
82     "prompt_tokens": 123,
83     "completion_tokens": 456,
84     "total_tokens": 579
85 },
86     "temperature": 1.0,
87     "top_p": 1.0,
88     "max_prompt_tokens": 1000,
89     "max_completion_tokens": 1000,
90     "truncation_strategy": {
91         "type": "auto",
92         "last_messages": null
93     },
94     "response_format": "auto",
95     "tool_choice": "auto",
96     "parallel_tool_calls": true
97 }
98 ],
99 "first_id": "run_abc123",
100 "last_id": "run_abc456",
101 "has_more": false
102 }
```

Retrieve run Beta

```
GET https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}
```

Retrieves a run.

Path parameters

run_id string Required

The ID of the run to retrieve.

thread_id string Required

The ID of the thread that was run.

Returns

The run object matching the specified ID.

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.retrieve(
5     thread_id="thread_abc123",
6     run_id="run_abc123"
7 )
8
9 print(run)
```

Response

🔗

```
1 {
2     "id": "run_abc123",
3     "object": "thread.run",
4     "created_at": 1699075072,
5     "assistant_id": "asst_abc123",
6     "thread_id": "thread_abc123",
7     "status": "completed",
8     "started_at": 1699075072,
9     "expires_at": null,
10    "cancelled_at": null,
11    "failed_at": null,
12    "completed_at": 1699075073,
13    "last_error": null,
14    "model": "gpt-4o",
```

```
15 "instructions": null,  
16 "incomplete_details": null,  
17 "tools": [  
18     {  
19         "type": "code_interpreter"  
20     }  
21 ],  
22 "metadata": {},  
23 "usage": {  
24     "prompt_tokens": 123,  
25     "completion_tokens": 456,  
26     "total_tokens": 579  
27 },  
28 "temperature": 1.0,  
29 "top_p": 1.0,  
30 "max_prompt_tokens": 1000,  
31 "max_completion_tokens": 1000,  
32 "truncation_strategy": {  
33     "type": "auto",  
34     "last_messages": null  
35 },  
36 "response_format": "auto",  
37 "tool_choice": "auto",  
38 "parallel_tool_calls": true  
39 }
```

Modify run

Beta

```
POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}
```

Modifies a run.

Path parameters

run_id string Required

The ID of the run to modify.

thread_id string Required

The ID of the thread that was run.

Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

Returns

The modified [run](#) object matching the specified ID.

Example request

python ⚡

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.update(
5     thread_id="thread_abc123",
6     run_id="run_abc123",
7     metadata={"user_id": "user_abc123"},
8 )
9
10 print(run)
```

Response

🔗

```
1 {
2     "id": "run_abc123",
3     "object": "thread.run",
4     "created_at": 1699075072,
5     "assistant_id": "asst_abc123",
6     "thread_id": "thread_abc123",
7     "status": "completed",
8     "started_at": 1699075072,
```

```
9     "expires_at": null,  
10    "cancelled_at": null,  
11    "failed_at": null,  
12    "completed_at": 1699075073,  
13    "last_error": null,  
14    "model": "gpt-4o",  
15    "instructions": null,  
16    "incomplete_details": null,  
17    "tools": [  
18      {  
19        "type": "code_interpreter"  
20      }  
21    ],  
22    "tool_resources": {  
23      "code_interpreter": {  
24        "file_ids": [  
25          "file-abc123",  
26          "file-abc456"  
27        ]  
28      }  
29    },  
30    "metadata": {  
31      "user_id": "user_abc123"  
32    },  
33    "usage": {  
34      "prompt_tokens": 123,  
35      "completion_tokens": 456,  
36      "total_tokens": 579  
37    },
```

```
38 "temperature": 1.0,  
39 "top_p": 1.0,  
40 "max_prompt_tokens": 1000,  
41 "max_completion_tokens": 1000,  
42 "truncation_strategy": {  
43     "type": "auto",  
44     "last_messages": null  
45 },  
46 "response_format": "auto",  
47 "tool_choice": "auto",  
48 "parallel_tool_calls": true  
49 }
```

Submit tool outputs to run Beta

POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/submit_tool_outputs

When a run has the `status: "requires_action"` and `required_action.type` is `submit_tool_outputs`, this endpoint can be used to submit the outputs from the tool calls once they're all completed. All outputs must be submitted in a single request.

Path parameters

run_id string Required

The ID of the run that requires the tool output submission.

thread_id string Required

The ID of the thread to which this run belongs.

Request body

tool_outputs array Required

A list of tools for which the outputs are being submitted.

› Show properties

stream boolean Optional

If `true`, returns a stream of events that happen during the Run as server-sent events, terminating when the Run enters a terminal state with a `data: [DONE]` message.

Returns

The modified run object matching the specified ID.

Default

Streaming

Example request

python ⚙️

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.submit_tool_outputs(
5     thread_id="thread_123",
6     run_id="run_123",
7     tool_outputs=[
8         {
9             "tool_call_id": "call_001",
10            "output": "70 degrees and sunny."
11        }
12    ]
13 )
14
15 print(run)
```

Response

🔗

```
1 {
2     "id": "run_123",
3     "object": "thread.run",
4     "created_at": 1699075592,
5     "assistant_id": "asst_123",
6     "thread_id": "thread_123",
7     "status": "queued",
8     "started_at": 1699075592,
```

```
9     "expires_at": 1699076192,  
10    "cancelled_at": null,  
11    "failed_at": null,  
12    "completed_at": null,  
13    "last_error": null,  
14    "model": "gpt-4o",  
15    "instructions": null,  
16    "tools": [  
17      {  
18        "type": "function",  
19        "function": {  
20          "name": "get_current_weather",  
21          "description": "Get the current weather in a given location",  
22          "parameters": {  
23            "type": "object",  
24            "properties": {  
25              "location": {  
26                "type": "string",  
27                "description": "The city and state, e.g. San Francisco, CA"  
28              },  
29              "unit": {  
30                "type": "string",  
31                "enum": ["celsius", "fahrenheit"]  
32              }  
33            },  
34            "required": ["location"]  
35          }  
36        }  
37      }
```

```
38 ],
39 "metadata": {},
40 "usage": null,
41 "temperature": 1.0,
42 "top_p": 1.0,
43 "max_prompt_tokens": 1000,
44 "max_completion_tokens": 1000,
45 "truncation_strategy": {
46     "type": "auto",
47     "last_messages": null
48 },
49 "response_format": "auto",
50 "tool_choice": "auto",
51 "parallel_tool_calls": true
52 }
```

Cancel a run Beta

POST https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/cancel

Cancels a run that is `in_progress`.

Path parameters

run_id string Required

The ID of the run to cancel.

thread_id string Required

The ID of the thread to which this run belongs.

Returns

The modified [run](#) object matching the specified ID.

Example request

python dropdown copy

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 run = client.beta.threads.runs.cancel(
5     thread_id="thread_abc123",
6     run_id="run_abc123"
7 )
8
9 print(run)
```

Response



```
1  {
2      "id": "run_abc123",
3      "object": "thread.run",
4      "created_at": 1699076126,
5      "assistant_id": "asst_abc123",
6      "thread_id": "thread_abc123",
7      "status": "cancelling",
8      "started_at": 1699076126,
9      "expires_at": 1699076726,
10     "cancelled_at": null,
11     "failed_at": null,
12     "completed_at": null,
13     "last_error": null,
14     "model": "gpt-4o",
15     "instructions": "You summarize books.",
16     "tools": [
17         {
18             "type": "file_search"
19         }
20     ],
21     "tool_resources": {
22         "file_search": {
23             "vector_store_ids": ["vs_123"]
24         }
25     },
26     "metadata": {},
27     "usage": null,
```

```
28     "temperature": 1.0,  
29     "top_p": 1.0,  
30     "response_format": "auto",  
31     "tool_choice": "auto",  
32     "parallel_tool_calls": true  
33 }
```

The run object Beta

Represents an execution run on a [thread](#).

assistant_id string

The ID of the [assistant](#) used for execution of this run.

cancelled_at integer or null

The Unix timestamp (in seconds) for when the run was cancelled.

completed_at integer or null

The Unix timestamp (in seconds) for when the run was completed.

created_at integer

The Unix timestamp (in seconds) for when the run was created.

expires_at integer or null

The Unix timestamp (in seconds) for when the run will expire.

failed_at integer or null

The Unix timestamp (in seconds) for when the run failed.

id string

The identifier, which can be referenced in API endpoints.

incomplete_details object or null

Details on why the run is incomplete. Will be `null` if the run is not incomplete.

› Show properties

instructions string

The instructions that the assistant used for this run.

last_error object or null

The last error associated with this run. Will be `null` if there are no errors.

› Show properties

max_completion_tokens integer or null

The maximum number of completion tokens specified to have been used over the course of the run.

max_prompt_tokens integer or null

The maximum number of prompt tokens specified to have been used over the course of the run.

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

model string

The model that the assistant used for this run.

object string

The object type, which is always `thread.run`.

parallel_tool_calls boolean

Whether to enable parallel function calling during tool use.

required_action object or null

Details on the action required to continue the run. Will be `null` if no action is required.

› Show properties

response_format "auto" or object

Specifies the format that the model must output. Compatible with [GPT-4o](#), [GPT-4 Turbo](#), and all GPT-3.5 Turbo models since `gpt-3.5-turbo-1106`.

Setting to `{ "type": "json_schema", "json_schema": { ... } }` enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the [Structured Outputs guide](#).

Setting to `{ "type": "json_object" }` enables JSON mode, which ensures the message the model generates is valid JSON.

Important: when using JSON mode, you **must** also instruct the model to produce JSON yourself via a system or user message. Without this, the model may generate an unending stream of whitespace until the generation reaches the token limit, resulting in a long-running and seemingly "stuck" request. Also note that the message content may be partially cut off if `finish_reason="length"`, which indicates the generation exceeded `max_tokens` or the conversation exceeded the max context length.

› Show possible types

started_at integer or null

The Unix timestamp (in seconds) for when the run was started.

status string

The status of the run, which can be either `queued`, `in_progress`, `requires_action`, `cancelling`, `cancelled`, `failed`, `completed`, `incomplete`, or `expired`.

temperature number or null

The sampling temperature used for this run. If not set, defaults to 1.

thread_id string

The ID of the thread that was executed on as a part of this run.

tool_choice string or object

Controls which (if any) tool is called by the model. `none` means the model will not call any tools and instead generates a message. `auto` is the default value and means the model can pick between generating a message or calling one or more tools. `required` means the model must call one or more tools before responding to the user. Specifying a particular tool like `{"type": "file_search"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

› Show possible types

tools array

The list of tools that the assistant used for this run.

› Show possible types

top_p number or null

The nucleus sampling value used for this run. If not set, defaults to 1.

truncation_strategy object or null

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

› Show properties

usage object

Usage statistics related to the run. This value will be `null` if the run is not in a terminal state (i.e. `in_progress`, `queued`, etc.).

› Show properties

OBJECT The run object



```
1  {
2    "id": "run_abc123",
3    "object": "thread.run",
4    "created_at": 1698107661,
5    "assistant_id": "asst_abc123",
6    "thread_id": "thread_abc123",
7    "status": "completed",
8    "started_at": 1699073476,
9    "expires_at": null,
10   "cancelled_at": null,
11   "failed_at": null,
12   "completed_at": 1699073498,
13   "last_error": null,
14   "model": "gpt-4o",
15   "instructions": null,
16   "tools": [{"type": "file_search"}, {"type": "code_interpreter"}],
17   "metadata": {},
18   "incomplete_details": null,
19   "usage": {
```

```
20     "prompt_tokens": 123,  
21     "completion_tokens": 456,  
22     "total_tokens": 579  
23 },  
24     "temperature": 1.0,  
25     "top_p": 1.0,  
26     "max_prompt_tokens": 1000,  
27     "max_completion_tokens": 1000,  
28     "truncation_strategy": {  
29         "type": "auto",  
30         "last_messages": null  
31     },  
32     "response_format": "auto",  
33     "tool_choice": "auto",  
34     "parallel_tool_calls": true  
35 }
```

PREVIOUS
< **Messages**

NEXT
Run steps >

Run steps

Beta



Represents the steps (model and tool calls) taken during the run.

Related guide: [Assistants](#)

List run steps

Beta



```
GET https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/steps
```

Returns a list of run steps belonging to a run.

Path parameters

run_id string Required

The ID of the run the run steps belong to.

thread_id string Required

The ID of the thread the run and run steps belong to.

Query parameters

after string Optional

A cursor for use in pagination. `after` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with `obj_foo`, your subsequent call can include `after=obj_foo` in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. `before` is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, starting with `obj_foo`, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

include[] array Optional

A list of additional fields to include in the response. Currently the only supported value is

`step_details.tool_calls[*].file_search.results[*].content` to fetch the file search result content.

See the [file search tool documentation](#) for more information.

limit integer Optional Defaults to 20

A limit on the number of objects to be returned. Limit can range between 1 and 100, and the default is 20.

order string Optional Defaults to desc

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

Returns

A list of [run step](#) objects.

Example request

```
from openai import OpenAI
client = OpenAI()

run_steps = client.beta.threads.runs.steps.list(
    thread_id="thread_abc123",
    run_id="run_abc123"
)

print(run_steps)
```

Response

```
{
  "object": "list",
  "data": [
    {
      "id": "step_abc123",
      "object": "thread.run.step",
      "created_at": 1699063291,
      "run_id": "run_abc123",
      "assistant_id": "asst_abc123",
      "thread_id": "thread_abc123",
```

```
"type": "message_creation",
"status": "completed",
"cancelled_at": null,
"completed_at": 1699063291,
"expired_at": null,
"failed_at": null,
"last_error": null,
"step_details": [
    "type": "message_creation",
    "message_creation": {
        "message_id": "msg_abc123"
    }
},
"usage": {
    "prompt_tokens": 123,
    "completion_tokens": 456,
    "total_tokens": 579
}
],
"first_id": "step_abc123",
"last_id": "step_abc456",
"has_more": false
}
```

Retrieve run step

Beta

```
GET https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/steps/{step_id}
```

Retrieves a run step.

Path parameters

run_id string Required

The ID of the run to which the run step belongs.

step_id string Required

The ID of the run step to retrieve.

thread_id string Required

The ID of the thread to which the run and run step belongs.

Query parameters

include[] array Optional

A list of additional fields to include in the response. Currently the only supported value is

```
step_details.tool_calls[*].file_search.results[*].content
```

 to fetch the file search result content.

See the [file search tool documentation](#) for more information.

Returns

The [run step](#) object matching the specified ID.

Example request

```
from openai import OpenAI
client = OpenAI()

run_step = client.beta.threads.runs.steps.retrieve(
    thread_id="thread_abc123",
    run_id="run_abc123",
    step_id="step_abc123"
)

print(run_step)
```

Response

```
{  
  "id": "step_abc123",  
  "object": "thread.run.step",  
  "created_at": 1699063291,  
  "run_id": "run_abc123",  
  "content": "Hello, world!"}
```

```
"assistant_id": "asst_abc123",
"thread_id": "thread_abc123",
"type": "message_creation",
"status": "completed",
"cancelled_at": null,
"completed_at": 1699063291,
"expired_at": null,
"failed_at": null,
"last_error": null,
"step_details": {
    "type": "message_creation",
    "message_creation": {
        "message_id": "msg_abc123"
    }
},
"usage": {
    "prompt_tokens": 123,
    "completion_tokens": 456,
    "total_tokens": 579
}
}
```

The run step object

Beta

Represents a step in execution of a run.

assistant_id string

The ID of the [assistant](#) associated with the run step.

cancelled_at integer

The Unix timestamp (in seconds) for when the run step was cancelled.

completed_at integer

The Unix timestamp (in seconds) for when the run step completed.

created_at integer

The Unix timestamp (in seconds) for when the run step was created.

expired_at integer

The Unix timestamp (in seconds) for when the run step expired. A step is considered expired if the parent run is expired.

failed_at integer

The Unix timestamp (in seconds) for when the run step failed.

id string

The identifier of the run step, which can be referenced in API endpoints.

last_error object

The last error associated with this run step. Will be `null` if there are no errors.

› Show properties

metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format, and querying for objects via API or the dashboard.

Keys are strings with a maximum length of 64 characters. Values are strings with a maximum length of 512 characters.

object string

The object type, which is always `thread.run.step`.

run_id string

The ID of the run that this run step is a part of.

status string

The status of the run step, which can be either `in_progress`, `cancelled`, `failed`, `completed`, or `expired`.

step_details object

The details of the run step.

› Show possible types

thread_id string

The ID of the thread that was run.

type string

The type of run step, which can be either `message_creation` or `tool_calls`.

usage object

Usage statistics related to the run step. This value will be `null` while the run step's status is `in_progress`.

> Show properties

OBJECT The run step object

```
{  
  "id": "step_abc123",  
  "object": "thread.run.step",  
  "created_at": 1699063291,  
  "run_id": "run_abc123",  
  "assistant_id": "asst_abc123",  
  "thread_id": "thread_abc123",  
  "type": "message_creation",  
  "status": "completed",  
  "cancelled_at": null,  
  "completed_at": 1699063291,  
  "expired_at": null,  
  "failed_at": null,  
  "last_error": null,  
  "step_details": {  
    "type": "message_creation",  
    "message_creation": {  
      "message_id": "msg_abc123"  
    }  
  }  
}
```

```
 },
"usage": {
  "prompt_tokens": 123,
  "completion_tokens": 456,
  "total_tokens": 579
}
}
```

< PREVIOUS
Runs

NEXT ▾

[Dashboard](#)[Docs](#)[API](#)

g

OpenAI Platform

Streaming

Beta

Stream the result of executing a Run or resuming a Run after submitting tool outputs. You can stream events from the [Create Thread and Run](#), [Create Run](#), and [Submit Tool Outputs](#) endpoints by passing `"stream": true`. The response will be a [Server-Sent events](#) stream. Our Node and Python SDKs provide helpful utilities to make streaming easy. Reference the [Assistants API quickstart](#) to learn more.

The message delta object

Beta

Represents a message delta i.e. any changed fields on a message during streaming.

delta object

The delta containing the fields that have changed on the Message.

› Show properties

id string

The identifier of the message, which can be referenced in API endpoints.

object string

The object type, which is always `thread.message.delta`.

OBJECT The message delta object



```
1  {
2      "id": "msg_123",
3      "object": "thread.message.delta",
4      "delta": {
5          "content": [
6              {
7                  "index": 0,
8                  "type": "text",
9                  "text": { "value": "Hello", "annotations": [] }
10             }
11         ]
12     }
13 }
```

The run step delta object

Beta

Represents a run step delta i.e. any changed fields on a run step during streaming.

delta object

The delta containing the fields that have changed on the run step.

› Show properties

id string

The identifier of the run step, which can be referenced in API endpoints.

object string

The object type, which is always `thread.run.step.delta`.

OBJECT The run step delta object



```
1  {
2    "id": "step_123",
3    "object": "thread.run.step.delta",
4    "delta": {
5      "step_details": {
6        "type": "tool_calls",
```

```
7     "tool_calls": [
8       {
9         "index": 0,
10        "id": "call_123",
11        "type": "code_interpreter",
12        "code_interpreter": { "input": "", "outputs": [] }
13      }
14    ]
15  }
16}
17}
```

Assistant stream events Beta

Represents an event emitted when streaming a Run.

Each event in a server-sent events stream has an `event` and `data` property:

```
event: thread.created
data: {"id": "thread_123", "object": "thread", ...}
```



We emit events whenever a new object is created, transitions to a new state, or is being streamed in parts (deltas). For example, we emit `thread.run.created` when a new run is created, `thread.run.completed` when a run completes, and so on. When an Assistant chooses to create a message during a run, we emit a `thread.message.created` event, a `thread.message.in_progress` event, many `thread.message.delta` events, and finally a `thread.message.completed` event.

We may add additional events over time, so we recommend handling unknown events gracefully in your code. See the [Assistants API quickstart](#) to learn how to integrate the Assistants API with streaming.

done `data` is `[DONE]`

Occurs when a stream ends.

error `data` is an [error](#)

Occurs when an [error](#) occurs. This can happen due to an internal server error or a timeout.

thread.created `data` is a [thread](#)

Occurs when a new [thread](#) is created.

thread.message.completed `data` is a [message](#)

Occurs when a [message](#) is completed.

thread.message.created `data` is a message

Occurs when a message is created.

thread.message.delta `data` is a message delta

Occurs when parts of a Message are being streamed.

thread.message.in_progress `data` is a message

Occurs when a message moves to an `in_progress` state.

thread.message.incomplete `data` is a message

Occurs when a message ends before it is completed.

thread.run.cancelled `data` is a run

Occurs when a run is cancelled.

thread.run.cancelling `data` is a run

Occurs when a run moves to a `cancelling` status.

thread.run.completed `data` is a run

Occurs when a run is completed.

thread.run.created `data` is a [run](#)

Occurs when a new [run](#) is created.

thread.run.expired `data` is a [run](#)

Occurs when a [run](#) expires.

thread.run.failed `data` is a [run](#)

Occurs when a [run](#) fails.

thread.run.in_progress `data` is a [run](#)

Occurs when a [run](#) moves to an [in_progress](#) status.

thread.run.incomplete `data` is a [run](#)

Occurs when a [run](#) ends with status [incomplete](#).

thread.run.queued `data` is a [run](#)

Occurs when a [run](#) moves to a [queued](#) status.

thread.run.requires_action `data` is a [run](#)

Occurs when a [run](#) moves to a [requires_action](#) status.

thread.run.step.cancelled `data` is a [run step](#)

Occurs when a [run step](#) is cancelled.

thread.run.step.completed `data` is a [run step](#)

Occurs when a [run step](#) is completed.

thread.run.step.created `data` is a [run step](#)

Occurs when a [run step](#) is created.

thread.run.step.delta `data` is a [run step delta](#)

Occurs when parts of a [run step](#) are being streamed.

thread.run.step.expired `data` is a [run step](#)

Occurs when a [run step](#) expires.

thread.run.step.failed `data` is a [run step](#)

Occurs when a [run step](#) fails.

thread.run.step.in_progress `data` is a [run step](#)

Occurs when a [run step](#) moves to an `in_progress` state.

< PREVIOUS

Run steps

NEXT >

Administration