

API Reference - OpenAI API

platform.openai.com/docs/api-reference/introduction

Introduction

You can interact with the API through HTTP requests from any language, via our official Python bindings, our official Node.js library, or a [community-maintained library](#).

To install the official Python bindings, run the following command:

```
pip install openai
```

To install the official Node.js library, run the following command in your Node.js project directory:

```
npm install openai
```

Authentication

API keys

The OpenAI API uses API keys for authentication. You can create API keys at a user or service account level. Service accounts are tied to a "bot" individual and should be used to provision access for production systems. Each API key can be scoped to one of the following,

1. **Project keys** - Provides access to a single project ([preferred option](#)); access [Project API keys](#) by selecting the specific project you wish to generate keys against.
2. **User keys** - Our legacy keys. Provides access to all organizations and all projects that user has been added to; access [API Keys](#) to view your available keys. We highly advise transitioning to project keys for best security practices, although access via this method is currently still supported.

Remember that your API key is a secret! Do not share it with others or expose it in any client-side code (browsers, apps). Production requests must be routed through your own backend server where your API key can be securely loaded from an environment variable or key management service.

All API requests should include your API key in an [Authorization](#) HTTP header as follows:

```
Authorization: Bearer OPENAI_API_KEY
```

Organizations and projects (optional)

For users who belong to multiple organizations or are accessing their projects through their legacy user API key, you can pass a header to specify which organization and project is used for an API request. Usage from these API requests will count as usage for the specified organization and project.

Example curl command:

```
1
2
3
4
curl https://api.openai.com/v1/models \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Organization: org-SqIq0BEvTdm3XtadHR6SFQfd" \
-H "OpenAI-Project: $PROJECT_ID"
```

Example with the `openai` Python package:

```
1
2
3
4
5
6
from openai import OpenAI

client = OpenAI(
    organization='org-SqIq0BEvTdm3XtadHR6SFQfd',
    project='$PROJECT_ID',
)
```

Example with the `openai` Node.js package:

```
1
2
3
4
5
6
import OpenAI from "openai";

const openai = new OpenAI({
    organization: "org-SqIq0BEvTdm3XtadHR6SFQfd",
    project: "$PROJECT_ID",
});
```

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.

Making requests

You can paste the command below into your terminal to run your first API request. Make sure to replace `$OPENAI_API_KEY` with your secret API key. If you are using a legacy user key and you have multiple projects, you will also need to [specify the Project Id](#). For improved security, we recommend transitioning to project based keys instead.

```
1
2
3
4
5
6
7
8
curl https://api.openai.com/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-d '{
  "model": "gpt-4o-mini",
  "messages": [{"role": "user", "content": "Say this is a test!"}],
  "temperature": 0.7
}'
```

This request queries the [gpt-4o-mini](#) model (which under the hood points to a [gpt-4o-mini model variant](#)) to complete the text starting with a prompt of "*Say this is a test!*". You should get a response back that resembles the following:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
{
  "id": "chatcmpl-abc123",
  "object": "chat.completion",
  "created": 1677858242,
  "model": "gpt-4o-mini",
  "usage": {
    "prompt_tokens": 13,
    "completion_tokens": 7,
    "total_tokens": 20,
    "completion_tokens_details": {
      "reasoning_tokens": 0,
      "accepted_prediction_tokens": 0,
      "rejected_prediction_tokens": 0
    }
  },
  "choices": [
    {
      "message": {
        "role": "assistant",
        "content": "\n\nThis is a test!"
      },
      "logprobs": null,
      "finish_reason": "stop",
      "index": 0
    }
  ]
}
```

Now that you've generated your first chat completion, let's break down the response object. We can see the `finish_reason` is `stop` which means the API returned the full chat completion generated by the model without running into any limits. In the choices list, we only generated a single message but you can set the `n` parameter to generate multiple messages choices.

Streaming²

The OpenAI API provides the ability to stream responses back to a client in order to allow partial results for certain requests. To achieve this, we follow the Server-sent events standard. Our official Node and Python libraries include helpers to make parsing these events simpler.

Streaming is supported for both the Chat Completions API and the Assistants API. This section focuses on how streaming works for Chat Completions. Learn more about how streaming works in the Assistants API here.

In Python, a streaming request looks like:

```
1
2
3
4
5
6
7
8
9
10
11
12
from openai import OpenAI

client = OpenAI()

stream = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[{"role": "user", "content": "Say this is a test"}],
    stream=True,
)
for chunk in stream:
    if chunk.choices[0].delta.content is not None:
        print(chunk.choices[0].delta.content, end="")
```

In Node / Typescript, a streaming request looks like:

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
import OpenAI from "openai";

const openai = new OpenAI();

async function main() {
  const stream = await openai.chat.completions.create({
    model: "gpt-4o-mini",
    messages: [{ role: "user", content: "Say this is a test" }],
    store: true,
    stream: true,
  });
  for await (const chunk of stream) {
    process.stdout.write(chunk.choices[0]?.delta?.content || "");
  }
}

main();

```

Parsing Server-sent events

Parsing Server-sent events is non-trivial and should be done with caution. Simple strategies like splitting by a new line may result in parsing errors. We recommend using [existing client libraries](#) when possible.

Debugging requests

In addition to [error codes](#) returned from API responses, it may sometimes be necessary to inspect HTTP response headers as well. Of particular interest will be the headers which contain the unique ID of a particular API request, and 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, which will allow 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.

Request ID in Python

```
1
2
3
4
5
6
7
8
9
10
11
12
from openai import OpenAI
client = OpenAI()

response = client.chat.completions.create(
    messages=[{
        "role": "user",
        "content": "Say this is a test",
    }],
    model="gpt-4o-mini",
)
print(response._request_id)
```

Request ID in JavaScript

```
1
2
3
4
5
6
7
8
9
import OpenAI from 'openai';
const client = new OpenAI();

const response = await client.chat.completions.create({
  messages: [{ role: 'user', content: 'Say this is a test' }],
  model: 'gpt-4o-mini'
});

console.log(response._request_id);
```

Access raw response objects in SDKs

If you are using a lower-level HTTP client (like [fetch](#) or [HttpClient](#) in C#), you should already have access to response headers as a part of the HTTP interface.

If you are using one of OpenAI's [official SDKs](#) (which largely abstract the HTTP request/response cycle), you will need to access raw HTTP responses in a slightly different way.

Below is an example of accessing the raw response object (and the `x-ratelimit-limit-tokens` header) using our [Python SDK](#).

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
from openai import OpenAI
client = OpenAI()

response = client.chat.completions.with_raw_response.create(
    messages=[{
        "role": "user",
        "content": "Say this is a test",
    }],
    model="gpt-4o-mini",
)
print(response.headers.get('x-ratelimit-limit-tokens'))

# get the object that `chat.completions.create()` would have returned
completion = response.parse()
print(completion)

```

Here is how you'd access a raw response (and the `x-ratelimit-limit-tokens` header) using our [JavaScript SDK](#).

```

1
2
3
4
5
6
7
8
9
10
import OpenAI from 'openai';
const client = new OpenAI();

const response = await client.chat.completions.create({
    messages: [{ role: 'user', content: 'Say this is a test' }],
    model: 'gpt-4o-mini'
}).asResponse();

// access the underlying Response object
console.log(response.headers.get('x-ratelimit-limit-tokens'));

```

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 will adhere to [semantic versioning](#))
- [Model families](#) (like [gpt-4o](#) or [o1-mini](#))

Backwards-compatible changes and upgrades will be continuously delivered over time. These and any rare breaking changes will be communicated in the [changelog](#). Here are some examples of changes which we consider to be backwards-compatible (non-breaking) changes.

Changes in model prompting behavior between snapshots

Model outputs are by their nature variable, so changes in prompting and model behavior between snapshots should be expected. 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)

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

One of the available [TTS models](#): [tts-1](#) or [tts-1-hd](#)

The text to generate audio for. The maximum length is 4096 characters.

The voice to use when generating the audio. Supported voices are `alloy`, `ash`, `coral`, `echo`, `fable`, `onyx`, `nova`, `sage` and `shimmer`. Previews of the voices are available in the [Text to speech guide](#).

The format to audio in. Supported formats are `mp3`, `opus`, `aac`, `flac`, `wav`, and `pcm`.

The speed of the generated audio. Select a value from `0.25` to `4.0`. `1.0` is the default.

Returns

The audio file content.

Example request

```
1
2
3
4
5
6
7
8
9
curl https://api.openai.com/v1/audio/speech \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
  "model": "tts-1",
  "input": "The quick brown fox jumped over the lazy dog.",
  "voice": "alloy"
}' \
--output speech.mp3
```

Create transcription

↪
post <https://api.openai.com/v1/audio/transcriptions>

Transcribes audio into the input language.

Request body

The audio file object (not file name) to transcribe, in one of these formats: flac, mp3, mp4, mpeg, mpga, m4a, ogg, wav, or webm.

ID of the model to use. Only `whisper-1` (which is powered by our open source Whisper V2 model) is currently available.

The language of the input audio. Supplying the input language in [ISO-639-1](#) (e.g. `en`) format will improve accuracy and latency.

An optional text to guide the model's style or continue a previous audio segment. The [prompt](#) should match the audio language.

The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, or `vtt`.

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.

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.

Returns

The [transcription object](#) or a [verbose transcription object](#).

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/audio/transcriptions \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: multipart/form-data" \
-F file="@/path/to/file/audio.mp3" \
-F model="whisper-1"
```

Response

```
1
2
3
{
  "text": "Imagine the wildest idea that you've ever had, and you're curious about how it
might scale to something that's a 100, a 1,000 times bigger. This is a place where you can
get to do that."
}
```

Create translation

⌚
post <https://api.openai.com/v1/audio/translations>

Translates audio into English.

Request body

The audio file object (not file name) translate, in one of these formats: flac, mp3, mp4, mpeg, mpg, m4a, ogg, wav, or webm.

ID of the model to use. Only `whisper-1` (which is powered by our open source Whisper V2 model) is currently available.

An optional text to guide the model's style or continue a previous audio segment. The `prompt` should be in English.

The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, or `vtt`.

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

```
1
2
3
4
5
curl https://api.openai.com/v1/audio/translations \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: multipart/form-data" \
-F file="@/path/to/file/german.m4a" \
-F model="whisper-1"
```

Response

```
1
2
3
{
  "text": "Hello, my name is Wolfgang and I come from Germany. Where are you heading today?"
}
```

The transcription object (JSON)



Represents a transcription response returned by model, based on the provided input.

The transcribed text.

OBJECT The transcription object (JSON)

```
1
2
3
{
  "text": "Imagine the wildest idea that you've ever had, and you're curious about how it
  might scale to something that's a 100, a 1,000 times bigger. This is a place where you can
  get to do that."
}
```

The transcription object (Verbose JSON)



Represents a verbose json transcription response returned by model, based on the provided input.

The language of the input audio.

The duration of the input audio.

The transcribed text.

Extracted words and their corresponding timestamps.

Segments of the transcribed text and their corresponding details.

OBJECT The transcription object (Verbose JSON)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "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, building sandcastles, and playing beach volleyball.",
  "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
    },
    ...
  ]
}
```

Chat



Given a list of messages comprising a conversation, the model will return a response. Related guide: [Chat Completions](#)

Create chat completion



post <https://api.openai.com/v1/chat/completions>

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

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](#).

ID of the model to use. See the [model endpoint compatibility](#) table for details on which models work with the Chat API.

Whether or not to store the output of this chat completion request for use in our [model distillation](#) or [evals](#) products.

o1 and o3-mini models only

Constrains effort on reasoning for [reasoning models](#). Currently supported values are [low](#), [medium](#), and [high](#). Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

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.

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.

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.

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](#).

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. `logprobs` must be set to `true` if this parameter is used.

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 [o1 series models](#).

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

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.

Output types that you would like the model to generate for this request. 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"]
```

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.

Parameters for audio output. Required when audio output is requested with `modalities: ["audio"]`. [Learn more](#).

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.

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 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.

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.

Specifies the latency tier to use for processing the request. This parameter is relevant for customers subscribed to the scale tier service:

- If set to 'auto', and the Project is Scale tier enabled, the system will utilize scale tier credits until they are exhausted.
- If set to 'auto', and the Project is not Scale tier enabled, the request will be processed using the default service tier with a lower uptime SLA and no latency guarantee.
- If set to 'default', the request will be processed using the default service tier with a lower uptime SLA and no latency guarantee.
- When not set, the default behavior is 'auto'.

Up to 4 sequences where the API will stop generating further tokens.

If set, partial message deltas will be sent, like in ChatGPT. Tokens will be sent as data-only `server-sent events` as they become available, with the stream terminated by a `data: [DONE]` message.

[Example Python code.](#)

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

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.

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.

A list of tools the model may call. Currently, only functions are supported as a tool. Use this to provide a list of functions the model may generate JSON inputs for. A max of 128 functions are supported.

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.

Whether to enable [parallel function calling](#) during tool use.

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse.
[Learn more.](#)

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.

Deprecated in favor of `tools`.

A list of functions the model may generate JSON inputs for.

Returns

Returns a [chat completion](#) object, or a streamed sequence of [chat completion chunk](#) objects if the request is streamed.

Example request

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
curl https://api.openai.com/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-d '{
  "model": "gpt-4o",
  "messages": [
    {
      "role": "developer",
      "content": "You are a helpful assistant."
    },
    {
      "role": "user",
      "content": "Hello!"
    }
  ]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
{
  "id": "chatcmpl-123",
  "object": "chat.completion",
  "created": 1677652288,
  "model": "gpt-4o-mini",
  "system_fingerprint": "fp_44709d6fcb",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "\n\nHello there, how may I assist you today?",
      },
      "logprobs": null,
      "finish_reason": "stop"
    },
    "service_tier": "default",
    "usage": {
      "prompt_tokens": 9,
      "completion_tokens": 12,
      "total_tokens": 21,
      "completion_tokens_details": {
        "reasoning_tokens": 0,
        "accepted_prediction_tokens": 0,
        "rejected_prediction_tokens": 0
      }
    }
  }
}
```

The chat completion object

🔗

Represents a chat completion response returned by model, based on the provided input.

A unique identifier for the chat completion.

A list of chat completion choices. Can be more than one if `n` is greater than 1.

The Unix timestamp (in seconds) of when the chat completion was created.

The model used for the chat completion.

The service tier used for processing the request.

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.

The object type, which is always `chat.completion`.

Usage statistics for the completion request.

OBJECT The chat completion object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
{
  "id": "chatcmpl-123456",
  "object": "chat.completion",
  "created": 1728933352,
  "model": "gpt-4o-2024-08-06",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "Hi there! How can I assist you today?",
        "refusal": null
      },
      "logprobs": null,
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 19,
    "completion_tokens": 10,
    "total_tokens": 29,
    "prompt_tokens_details": {
      "cached_tokens": 0
    },
    "completion_tokens_details": {
    }
  }
}
```

```
        "reasoning_tokens": 0,  
        "accepted_prediction_tokens": 0,  
        "rejected_prediction_tokens": 0  
    }  
,  
    "system_fingerprint": "fp_6b68a8204b"  
}
```

The chat completion chunk object



Represents a streamed chunk of a chat completion response returned by model, based on the provided input.

A unique identifier for the chat completion. Each chunk has the same ID.

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}`.

The Unix timestamp (in seconds) of when the chat completion was created. Each chunk has the same timestamp.

The model to generate the completion.

The service tier used for processing the request.

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.

The object type, which is always `chat.completion.chunk`.

An optional field that will only be present when you set `stream_options: {"include_usage": true}` in your request. When present, it contains a null value except for the last chunk which contains the token usage statistics for the entire request.

OBJECT The chat completion chunk object

```
1
2
3
4
5
6
7
{"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-4o-mini", "system_fingerprint": "fp_44709d6fcb", "choices":[{"index":0,"delta": {"role":"assistant","content":""}, "logprobs":null,"finish_reason":null}]}  

{"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-4o-mini", "system_fingerprint": "fp_44709d6fcb", "choices":[{"index":0,"delta": {"content":"Hello"}, "logprobs":null,"finish_reason":null}]}  

....  

>{"id":"chatcmpl-123","object":"chat.completion.chunk","created":1694268190,"model":"gpt-4o-mini", "system_fingerprint": "fp_44709d6fcb", "choices":[{"index":0,"delta": {}, "logprobs":null,"finish_reason":"stop"}]}
```

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 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 [text-embedding-ada-002](#)), cannot be an empty string, and any array must be 2048 dimensions or less. [Example Python code](#) for counting tokens. Some models may also impose a limit on total number of tokens summed across inputs.

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.

The format to return the embeddings in. Can be either [float](#) or [base64](#).

The number of dimensions the resulting output embeddings should have. Only supported in [text-embedding-3](#) and later models.

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

```
1
2
3
4
5
6
7
8
curl https://api.openai.com/v1/embeddings \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
  "input": "The food was delicious and the waiter...",
  "model": "text-embedding-ada-002",
  "encoding_format": "float"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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.

The index of the embedding in the list of embeddings.

The embedding vector, which is a list of floats. The length of vector depends on the model as listed in the [embedding guide](#).

The object type, which is always "embedding".

OBJECT The embedding object

```
1
2
3
4
5
6
7
8
9
10
{
  "object": "embedding",
  "embedding": [
    0.0023064255,
    -0.009327292,
    .... (1536 floats total for ada-002)
    -0.0028842222,
  ],
  "index": 0
}
```

Fine-tuning



Manage fine-tuning jobs to tailor a model to your specific training data. Related guide: [Fine-tune models](#)

Create fine-tuning job



post https://api.openai.com/v1/fine_tuning/jobs

Creates a fine-tuning job which begins the process of creating a new model from a given dataset.

Response includes details of the enqueued job including job status and the name of the fine-tuned models once complete.

[Learn more about fine-tuning](#)

Request body

The name of the model to fine-tune. You can select one of the [supported models](#).

The ID of an uploaded file that contains training data.

See [upload file](#) for how to upload a file.

Your dataset must be formatted as a JSONL file. Additionally, you must upload your file with the purpose [fine-tune](#).

The contents of the file should differ depending on if the model uses the [chat](#), [completions](#) format, or if the fine-tuning method uses the [preference](#) format.

See the [fine-tuning guide](#) for more details.

The hyperparameters used for the fine-tuning job. This value is now deprecated in favor of `method`, and should be passed in under the `method` parameter.

A string of up to 64 characters that will be added to your fine-tuned model name.

For example, a `suffix` of "custom-model-name" would produce a model name like `ft:gpt-4o-mini:openai:custom-model-name:7p4lURel`.

The ID of an uploaded file that contains validation data.

If you provide this file, the data is used to generate validation metrics periodically during fine-tuning. These metrics can be viewed in the fine-tuning results file. The same data should not be present in both train and validation files.

Your dataset must be formatted as a JSONL file. You must upload your file with the purpose `fine-tune`.

See the [fine-tuning guide](#) for more details.

A list of integrations to enable for your fine-tuning job.

The seed controls the reproducibility of the job. Passing in the same seed and job parameters should produce the same results, but may differ in rare cases. If a seed is not specified, one will be generated for you.

The method used for fine-tuning.

Returns

A [fine-tuning.job](#) object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/fine_tuning/jobs \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-d '{
  "training_file": "file-BK7bzQj3FfZFXr7DbL6xJwfo",
  "model": "gpt-4o-mini"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
{
  "object": "fine_tuning.job",
  "id": "ftjob-abc123",
  "model": "gpt-4o-mini-2024-07-18",
  "created_at": 1721764800,
  "fine_tuned_model": null,
  "organization_id": "org-123",
  "result_files": [],
  "status": "queued",
  "validation_file": null,
  "training_file": "file-abc123",
  "method": {
    "type": "supervised",
    "supervised": {
      "hyperparameters": {
        "batch_size": "auto",
        "learning_rate_multiplier": "auto",
        "n_epochs": "auto",
      }
    }
  }
}
```

List fine-tuning jobs

🔗
get https://api.openai.com/v1/fine_tuning/jobs

List your organization's fine-tuning jobs

Query parameters

Identifier for the last job from the previous pagination request.

Number of fine-tuning jobs to retrieve.

Returns

A list of paginated [fine-tuning.job](#) objects.

Example request

```
1
2
curl https://api.openai.com/v1/fine_tuning/jobs?limit=2 \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "object": "list",
  "data": [
    {
      "object": "fine_tuning.job",
      "id": "ftjob-abc123",
      "model": "gpt-4o-mini-2024-07-18",
      "created_at": 1721764800,
      "fine_tuned_model": null,
      "organization_id": "org-123",
      "result_files": [],
      "status": "queued",
      "validation_file": null,
      "training_file": "file-abc123"
    },
    { ... },
    { ... }
  ], "has_more": true
}
```

List fine-tuning events



get https://api.openai.com/v1/fine_tuning/jobs/{fine_tuning_job_id}/events

Get status updates for a fine-tuning job.

Path parameters

The ID of the fine-tuning job to get events for.

Query parameters

Identifier for the last event from the previous pagination request.

Number of events to retrieve.

Returns

A list of fine-tuning event objects.

Example request

```
1
2
curl https://api.openai.com/v1/fine_tuning/jobs/ftjob-abc123/events \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
{
  "object": "list",
  "data": [
    {
      "object": "fine_tuning.job.event",
      "id": "ft-event-ddTJfwuMpfLXse00Am0Gqjm",
      "created_at": 1721764800,
      "level": "info",
      "message": "Fine tuning job successfully completed",
      "data": null,
      "type": "message"
    },
    {
      "object": "fine_tuning.job.event",
      "id": "ft-event-tyiGuB72evQncpH87xe505Sv",
      "created_at": 1721764800,
      "level": "info",
      "message": "New fine-tuned model created: ft:gpt-4o-mini:openai::7p4lURel",
      "data": null,
      "type": "message"
    }
  ],
  "has_more": true
}

```

List fine-tuning checkpoints



get https://api.openai.com/v1/fine_tuning/jobs/{fine_tuning_job_id}/checkpoints

List checkpoints for a fine-tuning job.

Path parameters

The ID of the fine-tuning job to get checkpoints for.

Query parameters

Identifier for the last checkpoint ID from the previous pagination request.

Number of checkpoints to retrieve.

Returns

A list of fine-tuning [checkpoint objects](#) for a fine-tuning job.

Example request

```
1
2
curl https://api.openai.com/v1/fine_tuning/jobs/ftjob-abc123/checkpoints \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
{
  "object": "list"
  "data": [
    {
      "object": "fine_tuning.job.checkpoint",
      "id": "ftckpt_zc4Q7MP6XxulcVzj4MZdwsAB",
      "created_at": 1721764867,
      "fine_tuned_model_checkpoint": "ft:gpt-4o-mini-2024-07-18:my-org:custom-suffix:96olL566:ckpt-step-2000",
      "metrics": {
        "full_valid_loss": 0.134,
        "full_valid_mean_token_accuracy": 0.874
      },
      "fine_tuning_job_id": "ftjob-abc123",
      "step_number": 2000,
    },
    {
      "object": "fine_tuning.job.checkpoint",
      "id": "ftckpt_enQCFmOTGj3syEpYVhBRLTSy",
      "created_at": 1721764800,
      "fine_tuned_model_checkpoint": "ft:gpt-4o-mini-2024-07-18:my-org:custom-suffix:7q8mpxmy:ckpt-step-1000",
      "metrics": {
        "full_valid_loss": 0.167,
        "full_valid_mean_token_accuracy": 0.781
      }
    }
  ]
}
```

```
        },
        "fine_tuning_job_id": "ftjob-abc123",
        "step_number": 1000,
    },
],
"first_id": "ftckpt_zc4Q7MP6XxulcVzj4MZdwsAB",
"last_id": "ftckpt_enQCFm0TGj3syEpYVhBRLTSy",
"has_more": true
}
```

Retrieve fine-tuning job

⌚
get https://api.openai.com/v1/fine_tuning/jobs/{fine_tuning_job_id}

Get info about a fine-tuning job.

[Learn more about fine-tuning](#)

Path parameters

The ID of the fine-tuning job.

Returns

The [fine-tuning](#) object with the given ID.

Example request

```
1
2
curl https://api.openai.com/v1/fine_tuning/jobs/ft-AF1WoRqd3aJAHSqc9NY7iL8F \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
{
  "object": "fine_tuning.job",
  "id": "ftjob-abc123",
  "model": "davinci-002",
  "created_at": 1692661014,
  "finished_at": 1692661190,
  "fine_tuned_model": "ft:davinci-002:my-org:custom_suffix:7q8mpxmy",
  "organization_id": "org-123",
  "result_files": [
    "file-abc123"
  ],
  "status": "succeeded",
  "validation_file": null,
  "training_file": "file-abc123",
  "hyperparameters": {
    "n_epochs": 4,
    "batch_size": 1,
    "learning_rate_multiplier": 1.0
  },
  "trained_tokens": 5768,
  "integrations": [],
  "seed": 0,
  "estimated_finish": 0,
```

```
"method": {  
    "type": "supervised",  
    "supervised": {  
        "hyperparameters": {  
            "n_epochs": 4,  
            "batch_size": 1,  
            "learning_rate_multiplier": 1.0  
        }  
    }  
}
```

Cancel fine-tuning

⌚
post https://api.openai.com/v1/fine_tuning/jobs/{fine_tuning_job_id}/cancel

Immediately cancel a fine-tune job.

Path parameters

The ID of the fine-tuning job to cancel.

Returns

The cancelled [fine-tuning](#) object.

Example request

```
1  
2  
curl -X POST https://api.openai.com/v1/fine_tuning/jobs/ftjob-abc123/cancel \  
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
{
  "object": "fine_tuning.job",
  "id": "ftjob-abc123",
  "model": "gpt-4o-mini-2024-07-18",
  "created_at": 1721764800,
  "fine_tuned_model": null,
  "organization_id": "org-123",
  "result_files": [],
  "status": "cancelled",
  "validation_file": "file-abc123",
  "training_file": "file-abc123"
}
```

Training format for chat models using the supervised method



The per-line training example of a fine-tuning input file for chat models using the supervised method.



messages

array

A list of tools the model may generate JSON inputs for.

Whether to enable parallel function calling during tool use.

A list of functions the model may generate JSON inputs for.

OBJECT Training format for chat models using the supervised method

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
{
  "messages": [
    { "role": "user", "content": "What is the weather in San Francisco?" },
    {
      "role": "assistant",
      "tool_calls": [
        {
          "id": "call_id",
          "type": "function",
          "function": {
            "name": "get_current_weather",
            "arguments": "{\"location\": \"San Francisco, USA\", \"format\": \"celsius\"}"
          }
        }
      ]
    }
  ],
  "parallel_tool_calls": false,
```

```
"tools": [
  {
    "type": "function",
    "function": {
      "name": "get_current_weather",
      "description": "Get the current weather",
      "parameters": {
        "type": "object",
        "properties": {
          "location": {
            "type": "string",
            "description": "The city and country, eg. San Francisco, USA"
          },
          "format": { "type": "string", "enum": ["celsius", "fahrenheit"] }
        },
        "required": ["location", "format"]
      }
    }
  }
]
```

Training format for chat models using the preference method

⌚
The per-line training example of a fine-tuning input file for chat models using the dpo method.

⌚
input

object

The preferred completion message for the output.

The non-preferred completion message for the output.

OBJECT Training format for chat models using the preference method

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "input": {
    "messages": [
      { "role": "user", "content": "What is the weather in San Francisco?" }
    ]
  },
  "preferred_completion": [
    {
      "role": "assistant",
      "content": "The weather in San Francisco is 70 degrees Fahrenheit."
    }
  ],
  "non_preferred_completion": [
    {
      "role": "assistant",
      "content": "The weather in San Francisco is 21 degrees Celsius."
    }
  ]
}
```

Training format for completions models



The per-line training example of a fine-tuning input file for completions models

The input prompt for this training example.

The desired completion for this training example.

OBJECT Training format for completions models

```
1
2
3
4
{
  "prompt": "What is the answer to 2+2",
  "completion": "4"
}
```

The fine-tuning job object



The `fine_tuning.job` object represents a fine-tuning job that has been created through the API.

The object identifier, which can be referenced in the API endpoints.

The Unix timestamp (in seconds) for when the fine-tuning job was created.

For fine-tuning jobs that have `failed`, this will contain more information on the cause of the failure.

The name of the fine-tuned model that is being created. The value will be null if the fine-tuning job is still running.

The Unix timestamp (in seconds) for when the fine-tuning job was finished. The value will be null if the fine-tuning job is still running.

The hyperparameters used for the fine-tuning job. This value will only be returned when running `supervised` jobs.

The base model that is being fine-tuned.

The object type, which is always "fine_tuning.job".

The organization that owns the fine-tuning job.

The compiled results file ID(s) for the fine-tuning job. You can retrieve the results with the [Files API](#).

The current status of the fine-tuning job, which can be either `validating_files`, `queued`, `running`, `succeeded`, `failed`, or `cancelled`.

The total number of billable tokens processed by this fine-tuning job. The value will be null if the fine-tuning job is still running.

The file ID used for training. You can retrieve the training data with the [Files API](#).

The file ID used for validation. You can retrieve the validation results with the [Files API](#).

A list of integrations to enable for this fine-tuning job.

The seed used for the fine-tuning job.

The Unix timestamp (in seconds) for when the fine-tuning job is estimated to finish. The value will be null if the fine-tuning job is not running.

The method used for fine-tuning.

OBJECT The fine-tuning job object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
{
  "object": "fine_tuning.job",
  "id": "ftjob-abc123",
  "model": "davinci-002",
  "created_at": 1692661014,
  "finished_at": 1692661190,
  "fine_tuned_model": "ft:davinci-002:my-org:custom_suffix:7q8mpxmy",
  "organization_id": "org-123",
  "result_files": [
    "file-abc123"
  ],
  "status": "succeeded",
  "validation_file": null,
  "training_file": "file-abc123",
  "hyperparameters": {
    "n_epochs": 4,
    "batch_size": 1,
    "learning_rate_multiplier": 1.0
  },
  "trained_tokens": 5768,
  "integrations": [],
  "seed": 0,
  "estimated_finish": 0,
```

```
"method": {  
    "type": "supervised",  
    "supervised": {  
        "hyperparameters": {  
            "n_epochs": 4,  
            "batch_size": 1,  
            "learning_rate_multiplier": 1.0  
        }  
    }  
}  
}
```

The fine-tuning job event object

⌚

Fine-tuning job event object

The object type, which is always "fine_tuning.job.event".

The object identifier.

The Unix timestamp (in seconds) for when the fine-tuning job was created.

The log level of the event.

The message of the event.

The type of event.

The data associated with the event.

OBJECT The fine-tuning job event object

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
{  
    "object": "fine_tuning.job.event",  
    "id": "ftevent-abc123",  
    "created_at": 1677610602,  
    "level": "info",  
    "message": "Created fine-tuning job",  
    "data": {},  
    "type": "message"  
}
```

The fine-tuning job checkpoint object



The `fine_tuning.job.checkpoint` object represents a model checkpoint for a fine-tuning job that is ready to use.

The checkpoint identifier, which can be referenced in the API endpoints.

The Unix timestamp (in seconds) for when the checkpoint was created.

The name of the fine-tuned checkpoint model that is created.

The step number that the checkpoint was created at.

Metrics at the step number during the fine-tuning job.

The name of the fine-tuning job that this checkpoint was created from.

The object type, which is always "fine_tuning.job.checkpoint".

OBJECT The fine-tuning job checkpoint object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
{
  "object": "fine_tuning.job.checkpoint",
  "id": "ftckpt_qtz5Gyk4BLq1SfLFWp3Rt03P",
  "created_at": 1712211699,
  "fine_tuned_model_checkpoint": "ft:gpt-4o-mini-2024-07-18:my-
org:custom_suffix:9ABel2dg:ckpt-step-88",
  "fine_tuning_job_id": "ftjob-fpbNQ3H1GrMehXRF8c097xTN",
  "metrics": {
    "step": 88,
    "train_loss": 0.478,
    "train_mean_token_accuracy": 0.924,
    "valid_loss": 10.112,
    "valid_mean_token_accuracy": 0.145,
    "full_valid_loss": 0.567,
    "full_valid_mean_token_accuracy": 0.944
  },
  "step_number": 88
}
```

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

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.

The endpoint to be used for all requests in the batch. Currently [/v1/chat/completions](#), [/v1/embeddings](#), and [/v1/completions](#) are supported. Note that [/v1/embeddings](#) batches are also restricted to a maximum of 50,000 embedding inputs across all requests in the batch.

The time frame within which the batch should be processed. Currently only [24h](#) is supported.

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 created [Batch](#) object.

Example request

```
1
2
3
4
5
6
7
8
curl https://api.openai.com/v1/batches \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
  "input_file_id": "file-abc123",
  "endpoint": "/v1/chat/completions",
  "completion_window": "24h"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
{
  "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

The ID of the batch to retrieve.

Returns

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

Example request

```
1  
2  
3  
curl https://api.openai.com/v1/batches/batch_abc123 \  
  -H "Authorization: Bearer $OPENAI_API_KEY" \  
  -H "Content-Type: application/json" \  
  
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
{
  "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-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 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

The ID of the batch to cancel.

Returns

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

Example request

```
1  
2  
3  
4  
curl https://api.openai.com/v1/batches/batch_abc123/cancel \  
  -H "Authorization: Bearer $OPENAI_API_KEY" \  
  -H "Content-Type: application/json" \  
  -X POST
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
{
  "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

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.

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

```
1  
2  
3  
curl https://api.openai.com/v1/batches?limit=2 \  
  -H "Authorization: Bearer $OPENAI_API_KEY" \  
  -H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
{
  "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

⌚
⌚
id

string

The object type, which is always **batch**.

The OpenAI API endpoint used by the batch.

⌚
errors

object

The ID of the input file for the batch.

The time frame within which the batch should be processed.

The current status of the batch.

The ID of the file containing the outputs of successfully executed requests.

The ID of the file containing the outputs of requests with errors.

The Unix timestamp (in seconds) for when the batch was created.

The Unix timestamp (in seconds) for when the batch started processing.

The Unix timestamp (in seconds) for when the batch will expire.

The Unix timestamp (in seconds) for when the batch started finalizing.

The Unix timestamp (in seconds) for when the batch was completed.

The Unix timestamp (in seconds) for when the batch failed.

The Unix timestamp (in seconds) for when the batch expired.

The Unix timestamp (in seconds) for when the batch started cancelling.

The Unix timestamp (in seconds) for when the batch was cancelled.

The request counts for different statuses within the batch.

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 The batch object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
{
  "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-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 eval job",
```

```
    }  
}
```

The request input object

⌚ The per-line object of the batch input file

A developer-provided per-request id that will be used to match outputs to inputs. Must be unique for each request in a batch.

The HTTP method to be used for the request. Currently only `POST` is supported.

The OpenAI API relative URL to be used for the request. Currently `/v1/chat/completions`, `/v1/embeddings`, and `/v1/completions` are supported.

OBJECT The request input object

```
{"custom_id": "request-1", "method": "POST", "url": "/v1/chat/completions", "body": {"model": "gpt-4o-mini", "messages": [{"role": "system", "content": "You are a helpful assistant."}, {"role": "user", "content": "What is 2+2?"}]}}
```

The request output object

⌚ The per-line object of the batch output and error files

⌚
id

string

A developer-provided per-request id that will be used to match outputs to inputs.

⌚
response

object or null

For requests that failed with a non-HTTP error, this will contain more information on the cause of the failure.

OBJECT The request output object

```
{"id": "batch_req_wnaDys", "custom_id": "request-2", "response": {"status_code": 200, "request_id": "req_c187b3", "body": {"id": "chatcmpl-9758Iw", "object": "chat.completion", "created": 1711475054, "model": "gpt-4o-mini", "choices": [{"index": 0, "message": {"role": "assistant", "content": "2 + 2 equals 4."}, "finish_reason": "stop"}], "usage": {"prompt_tokens": 24, "completion_tokens": 15, "total_tokens": 39}, "system_fingerprint": null}, "error": null}
```

Files



Files are used to upload documents that can be used with features like [Assistants](#), [Fine-tuning](#), and [Batch API](#).

Upload file



post <https://api.openai.com/v1/files>

Upload a file that can be used across various endpoints. Individual files can be up to 512 MB, and the size of all files uploaded by one organization can be up to 100 GB.

The Assistants API supports files up to 2 million tokens and of specific file types. See the [Assistants Tools guide](#) for details.

The Fine-tuning API only supports `.jsonl` files. The input also has certain required formats for fine-tuning [chat](#) or [completions](#) models.

The Batch API only supports `.jsonl` files up to 200 MB in size. The input also has a specific required format.

Please [contact us](#) if you need to increase these storage limits.

Request body

The File object (not file name) to be uploaded.

The intended purpose of the uploaded file.

Use "assistants" for [Assistants](#) and [Message](#) files, "vision" for Assistants image file inputs, "batch" for [Batch API](#), and "fine-tune" for [Fine-tuning](#).

Returns

The uploaded [File](#) object.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/files \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-F purpose="fine-tune" \
-F file="@mydata.jsonl"
```

Response

```
1
2
3
4
5
6
7
8
{
  "id": "file-abc123",
  "object": "file",
  "bytes": 120000,
  "created_at": 1677610602,
  "filename": "mydata.jsonl",
  "purpose": "fine-tune",
}
```

List files

⌚
get <https://api.openai.com/v1/files>

Returns a list of files.

Query parameters

Only return files with the given purpose.

A limit on the number of objects to be returned. Limit can range between 1 and 10,000, and the default is 10,000.

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

Returns

A list of [File](#) objects.

Example request

```
1
2
curl https://api.openai.com/v1/files \
  -H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
{
  "data": [
    {
      "id": "file-abc123",
      "object": "file",
      "bytes": 175,
      "created_at": 1613677385,
      "filename": "salesOverview.pdf",
      "purpose": "assistants",
    },
    {
      "id": "file-abc123",
      "object": "file",
      "bytes": 140,
      "created_at": 1613779121,
      "filename": "puppy.jsonl",
      "purpose": "fine-tune",
    }
  ],
  "object": "list"
}
```

Retrieve file

🔗
get https://api.openai.com/v1/files/{file_id}

Returns information about a specific file.

Path parameters

The ID of the file to use for this request.

Returns

The File object matching the specified ID.

Example request

```
1
2
curl https://api.openai.com/v1/files/file-abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
{
  "id": "file-abc123",
  "object": "file",
  "bytes": 120000,
  "created_at": 1677610602,
  "filename": "mydata.jsonl",
  "purpose": "fine-tune",
}
```

Delete file

⌚
delete https://api.openai.com/v1/files/{file_id}

Delete a file.

Path parameters

The ID of the file to use for this request.

Returns

Deletion status.

Example request

```
1
2
3
curl https://api.openai.com/v1/files/file-abc123 \
-X DELETE \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
{
  "id": "file-abc123",
  "object": "file",
  "deleted": true
}
```

Retrieve file content

🔗
get https://api.openai.com/v1/files/{file_id}/content

Returns the contents of the specified file.

Path parameters

The ID of the file to use for this request.

Returns

The file content.

Example request

```
1
2
curl https://api.openai.com/v1/files/file-abc123/content \
  -H "Authorization: Bearer $OPENAI_API_KEY" > file.json
```

The file object

🔗
The `File` object represents a document that has been uploaded to OpenAI.

The file identifier, which can be referenced in the API endpoints.

The size of the file, in bytes.

The Unix timestamp (in seconds) for when the file was created.

The name of the file.

The object type, which is always `file`.

The intended purpose of the file. Supported values are `assistants`, `assistants_output`, `batch`, `batch_output`, `fine-tune`, `fine-tune-results` and `vision`.

Deprecated. The current status of the file, which can be either `uploaded`, `processed`, or `error`.

Deprecated. For details on why a fine-tuning training file failed validation, see the `error` field on [fine_tuning.job](#).

OBJECT The file object

```
1
2
3
4
5
6
7
8
{
  "id": "file-abc123",
  "object": "file",
  "bytes": 120000,
  "created_at": 1677610602,
  "filename": "salesOverview.pdf",
  "purpose": "assistants",
}
```

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 [purposes](#), the correct [mime_type](#) must be specified. Please refer to documentation for the supported MIME types for your use case:

Assistants

For guidance on the proper filename extensions for each purpose, please follow the documentation on [creating a File](#).

Request body

The name of the file to upload.

The intended purpose of the uploaded file.

See the [documentation on File purposes](#).

The number of bytes in the file you are uploading.

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](#).

Returns

The [Upload](#) object with status `pending`.

Example request

```
1
2
3
4
5
6
7
8
curl https://api.openai.com/v1/uploads \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-d '{
  "purpose": "fine-tune",
  "filename": "training_examples.jsonl",
  "bytes": 2147483648,
  "mime_type": "text/jsonl"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
{
  "id": "upload_abc123",
  "object": "upload",
  "bytes": 2147483648,
  "created_at": 1719184911,
  "filename": "training_examples.jsonl",
  "purpose": "fine-tune",
  "status": "pending",
  "expires_at": 1719127296
}
```

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

The ID of the Upload.

Request body

The chunk of bytes for this Part.

Returns

The upload Part object.

Example request

```
1  
2  
curl https://api.openai.com/v1/uploads/upload_abc123/parts  
-F data="aHR0cHM6Ly9hcGkub3BlbmFpLmNvbS92MS91cGxvYWRz..."
```

Response

```
1  
2  
3  
4  
5  
6  
{  
  "id": "part_def456",  
  "object": "upload.part",  
  "created_at": 1719185911,  
  "upload_id": "upload_abc123"  
}
```

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

The ID of the Upload.

Request body

The ordered list of Part IDs.

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

```
1
2
3
4
curl https://api.openai.com/v1/uploads/upload_abc123/complete
-d '{
  "part_ids": ["part_def456", "part_ghi789"]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "id": "upload_abc123",
  "object": "upload",
  "bytes": 2147483648,
  "created_at": 1719184911,
  "filename": "training_examples.jsonl",
  "purpose": "fine-tune",
  "status": "completed",
  "expires_at": 1719127296,
  "file": {
    "id": "file-xyz321",
    "object": "file",
    "bytes": 2147483648,
    "created_at": 1719186911,
    "filename": "training_examples.jsonl",
    "purpose": "fine-tune",
  }
}
```

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

The ID of the Upload.

Returns

The Upload object with status **cancelled**.

Example request

```
curl https://api.openai.com/v1/uploads/upload_abc123/cancel
```

Response

```
1
2
3
4
5
6
7
8
9
10
{
  "id": "upload_abc123",
  "object": "upload",
  "bytes": 2147483648,
  "created_at": 1719184911,
  "filename": "training_examples.jsonl",
  "purpose": "fine-tune",
  "status": "cancelled",
  "expires_at": 1719127296
}
```

The upload object



The Upload object can accept byte chunks in the form of Parts.

The Upload unique identifier, which can be referenced in API endpoints.

The Unix timestamp (in seconds) for when the Upload was created.

The name of the file to be uploaded.

The intended number of bytes to be uploaded.

The intended purpose of the file. [Please refer here](#) for acceptable values.

The status of the Upload.

The Unix timestamp (in seconds) for when the Upload was created.

The object type, which is always "upload".

The **File** object represents a document that has been uploaded to OpenAI.

OBJECT The upload object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "id": "upload_abc123",
  "object": "upload",
  "bytes": 2147483648,
  "created_at": 1719184911,
  "filename": "training_examples.jsonl",
  "purpose": "fine-tune",
  "status": "completed",
  "expires_at": 1719127296,
  "file": {
    "id": "file-xyz321",
    "object": "file",
    "bytes": 2147483648,
    "created_at": 1719186911,
    "filename": "training_examples.jsonl",
    "purpose": "fine-tune",
  }
}
```

The upload part object



The upload Part represents a chunk of bytes we can add to an Upload object.

The upload Part unique identifier, which can be referenced in API endpoints.

The Unix timestamp (in seconds) for when the Part was created.

The ID of the Upload object that this Part was added to.

The object type, which is always `upload.part`.

OBJECT The upload part object

```
1
2
3
4
5
6
{
  "id": "part_def456",
  "object": "upload.part",
  "created_at": 1719186911,
  "upload_id": "upload_abc123"
}
```

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.

Request body

A text description of the desired image(s). The maximum length is 1000 characters for `dall-e-2` and 4000 characters for `dall-e-3`.

The model to use for image generation.

The number of images to generate. Must be between 1 and 10. For `dall-e-3`, only `n=1` is supported.

The quality of the image that will be generated. `hd` creates images with finer details and greater consistency across the image. This param is only supported for `dall-e-3`.

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.

The size of the generated images. Must be one of `256x256`, `512x512`, or `1024x1024` for `dall-e-2`. Must be one of `1024x1024`, `1792x1024`, or `1024x1792` for `dall-e-3` models.

The style of the generated images. 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. This param is only supported for `dall-e-3`.

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

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
curl https://api.openai.com/v1/images/generations \  
-H "Content-Type: application/json" \  
-H "Authorization: Bearer $OPENAI_API_KEY" \  
-d '{  
    "model": "dall-e-3",  
    "prompt": "A cute baby sea otter",  
    "n": 1,  
    "size": "1024x1024"  
}'
```

Response

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
{  
    "created": 1589478378,  
    "data": [  
        {  
            "url": "https://..."  
        },  
        {  
            "url": "https://..."  
        }  
    ]  
}
```

Create image edit

🔗
post <https://api.openai.com/v1/images/edits>

Creates an edited or extended image given an original image and a prompt.

Request body

The image to edit. Must be a valid PNG file, less than 4MB, and square. If mask is not provided, image must have transparency, which will be used as the mask.

A text description of the desired image(s). The maximum length is 1000 characters.

An additional image whose fully transparent areas (e.g. where alpha is zero) indicate where `image` should be edited. Must be a valid PNG file, less than 4MB, and have the same dimensions as `image`.

The model to use for image generation. Only `dall-e-2` is supported at this time.

The number of images to generate. Must be between 1 and 10.

The size of the generated images. Must be one of `256x256`, `512x512`, or `1024x1024`.

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.

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

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/images/edits \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-F image="@otter.png" \
-F mask="@mask.png" \
-F prompt="A cute baby sea otter wearing a beret" \
-F n=2 \
-F size="1024x1024"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
{
  "created": 1589478378,
  "data": [
    {
      "url": "https://..."
    },
    {
      "url": "https://..."
    }
  ]
}
```

Create image variation

⌚
post <https://api.openai.com/v1/images/variants>

Creates a variation of a given image.

Request body

The image to use as the basis for the variation(s). Must be a valid PNG file, less than 4MB, and square.

The model to use for image generation. Only `dall-e-2` is supported at this time.

The number of images to generate. Must be between 1 and 10. For `dall-e-3`, only `n=1` is supported.

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.

The size of the generated images. Must be one of `256x256`, `512x512`, or `1024x1024`.

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

```
1
2
3
4
5
curl https://api.openai.com/v1/images/variations \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-F image="@otter.png" \
-F n=2 \
-F size="1024x1024"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
{
  "created": 1589478378,
  "data": [
    {
      "url": "https://..."
    },
    {
      "url": "https://..."
    }
  ]
}
```

The image object



Represents the url or the content of an image generated by the OpenAI API.

The base64-encoded JSON of the generated image, if `response_format` is `b64_json`.

The URL of the generated image, if `response_format` is `url` (default).

The prompt that was used to generate the image, if there was any revision to the prompt.

OBJECT The image object

```
1
2
3
4
{
  "url": "...",
  "revised_prompt": "..."
}
```

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

```
1
2
curl https://api.openai.com/v1/models \
  -H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
{
  "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

The ID of the model to use for this request

Returns

The model object matching the specified ID.

Example request

```
1
2
curl https://api.openai.com/v1/models/gpt-4o \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
6
{
  "id": "gpt-4o",
  "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

The model to delete

Returns

Deletion status.

Example request

```
1
2
3
curl https://api.openai.com/v1/models/ft:gpt-4o-mini:acemeco:suffix:abc123 \
-X DELETE \
-H "Authorization: Bearer $OPENAI_API_KEY"
```

Response

```
1
2
3
4
5
{
  "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.

The model identifier, which can be referenced in the API endpoints.

The Unix timestamp (in seconds) when the model was created.

The object type, which is always "model".

The organization that owns the model.

OBJECT The model object

```
1
2
3
4
5
6
{
  "id": "gpt-4o",
  "object": "model",
  "created": 1686935002,
  "owned_by": "openai"
}
```

Moderations



Given text and/or image inputs, classifies if those inputs are potentially harmful across several categories. Related guide: [Moderations](#)

Create moderation



post <https://api.openai.com/v1/moderations>

Classifies if text and/or image inputs are potentially harmful. Learn more in the [moderation guide](#).

Request body

Input (or inputs) to classify. Can be a single string, an array of strings, or an array of multi-modal input objects similar to other models.

The content moderation model you would like to use. Learn more in [the moderation guide](#), and learn about available models [here](#).

Returns

A [moderation](#) object.

Example request

```
1
2
3
4
5
6
curl https://api.openai.com/v1/moderations \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-d '{
  "input": "I want to kill them."
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
{
  "id": "modr-AB8Cj0Tu2jiq12hp1AQPfeqFWaORR",
  "model": "text-moderation-007",
  "results": [
    {
      "flagged": true,
      "categories": {
        "sexual": false,
        "hate": false,
        "harassment": true,
        "self-harm": false,
        "sexual/minors": false,
        "hate/threatening": false,
        "violence/graphic": false,
        "self-harm/intent": false,
        "self-harm/instructions": false,
        "harassment/threatening": true,
        "violence": true
      },
      "category_scores": {
        "sexual": 0.000011726012417057063,
        "hate": 0.22706663608551025,
      }
    }
  ]
}
```

```
"harassment": 0.5215635299682617,  
"self-harm": 2.227119921371923e-6,  
"sexual/minors": 7.107352217872176e-8,  
"hate/threatening": 0.023547329008579254,  
"violence/graphic": 0.00003391829886822961,  
"self-harm/intent": 1.646940972932498e-6,  
"self-harm/instructions": 1.1198755256458526e-9,  
"harassment/threatening": 0.5694745779037476,  
"violence": 0.9971134662628174  
}  
}  
]  
}
```

The moderation object

⌚ Represents if a given text input is potentially harmful.

The unique identifier for the moderation request.

The model used to generate the moderation results.

A list of moderation objects.

OBJECT The moderation object

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

```
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
{
  "id": "modr-0d9740456c391e43c445bf0f010940c7",
  "model": "omni-moderation-latest",
  "results": [
    {
      "flagged": true,
      "categories": {
        "harassment": true,
        "harassment/threatening": true,
        "sexual": false,
        "hate": false,
        "hate/threatening": false,
        "illicit": false,
        "illicit/violent": false,
        "self-harm/intent": false,
        "self-harm/instructions": false,
        "self-harm": false,
        "sexual/minors": false,
        "violence": true,
        "violence/graphic": true
      },
      "category_scores": {
        "harassment": 0.8189693396524255,
        "harassment/threatening": 0.804985420696006,
        "sexual": 1.573112165348997e-6,
        "hate": 0.007562942636942845,
        "hate/threatening": 0.004208854591835476,
        "illicit": 0.030535955153511665,
      }
    }
  ]
}
```

```
"illicit/violent": 0.008925306722380033,  
"self-harm/intent": 0.00023023930975076432,  
"self-harm/instructions": 0.0002293869201073356,  
"self-harm": 0.012598046106750154,  
"sexual/minors": 2.212566909570261e-8,  
"violence": 0.9999992735124786,  
"violence/graphic": 0.843064871157054  
},  
"category_applied_input_types": {  
    "harassment": [  
        "text"  
    ],  
    "harassment/threatening": [  
        "text"  
    ],  
    "sexual": [  
        "text",  
        "image"  
    ],  
    "hate": [  
        "text"  
    ],  
    "hate/threatening": [  
        "text"  
    ],  
    "illicit": [  
        "text"  
    ],  
    "illicit/violent": [  
        "text"  
    ],  
    "self-harm/intent": [  
        "text",  
        "image"  
    ],  
    "self-harm/instructions": [  
        "text",  
        "image"  
    ],  
    "self-harm": [  
        "text",  
        "image"  
    ],  
    "sexual/minors": [  
        "text"  
    ],  
    "violence": [  
        "text",  
        "image"  
    ],  
    "violence/graphic": [  
        "text",  
        "image"  
    ]  
}
```

```
]  
}
```

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

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.

The name of the assistant. The maximum length is 256 characters.

The description of the assistant. The maximum length is 512 characters.

The system instructions that the assistant uses. The maximum length is 256,000 characters.

o1 and o3-mini models only

Constrains effort on reasoning for [reasoning models](#). Currently supported values are [low](#), [medium](#), and [high](#). Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

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](#).

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.

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.

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.

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.

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.

Returns

An [assistant](#) object.

Example request

```
1
2
3
4
5
6
7
8
9
10
curl "https://api.openai.com/v1/assistants" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "instructions": "You are a personal math tutor. When asked a question, write and run
Python code to answer the question.",
  "name": "Math Tutor",
  "tools": [{"type": "code_interpreter"}],
  "model": "gpt-4o"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "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 to answer the question.",
  "tools": [
    {
      "type": "code_interpreter"
    }
  ],
  "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

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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.

Returns

A list of assistant objects.

Example request

```
1
2
3
4
curl "https://api.openai.com/v1/assistants?order=desc&limit=20" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
{
  "object": "list",
  "data": [
    {
      "label": "Section 1"
    },
    {
      "label": "Section 2"
    }
  ]
}
```

```

    "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

The ID of the assistant to retrieve.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/assistants/asst_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "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.",
  "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

The ID of the assistant to modify.

Request body

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.

o1 and o3-mini models only

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `low`, `medium`, and `high`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

The name of the assistant. The maximum length is 256 characters.

The description of the assistant. The maximum length is 512 characters.

The system instructions that the assistant uses. The maximum length is 256,000 characters.

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`.

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.

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.

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.

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.

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.

Returns

The modified assistant object.

Example request

```
1
2
3
4
5
6
7
8
9
curl https://api.openai.com/v1/assistants/asst_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
    "instructions": "You are an HR bot, and you have access to files to answer employee
questions about company policies. Always response with info from either of the files.",
    "tools": [{"type": "file_search"}],
    "model": "gpt-4o"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "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 company policies. Always response with info from either of the files.",
  "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

The ID of the assistant to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/assistants/asst_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  "id": "asst_abc123",
  "object": "assistant.deleted",
  "deleted": true
}
```

The assistant object

Beta



Represents an [assistant](#) that can call the model and use tools.

The identifier, which can be referenced in API endpoints.

The object type, which is always [assistant](#).

The Unix timestamp (in seconds) for when the assistant was created.

The name of the assistant. The maximum length is 256 characters.

The description of the assistant. The maximum length is 512 characters.

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.

The system instructions that the assistant uses. The maximum length is 256,000 characters.

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`.

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.

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.

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.

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.

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.

OBJECT The assistant object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "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 to answer the question.",
  "tools": [
    {
      "type": "code_interpreter"
    }
  ],
  "metadata": {},
  "top_p": 1.0,
  "temperature": 1.0,
  "response_format": "auto"
}
```

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

A list of [messages](#) to start the thread with.

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.

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 [thread](#) object.

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/threads \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-d ''
```

Response

```
1
2
3
4
5
6
7
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1699012949,
  "metadata": {},
  "tool_resources": {}
}
```

Retrieve thread

Beta



get https://api.openai.com/v1/threads/{thread_id}

Retrieves a thread.

Path parameters

The ID of the thread to retrieve.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1699014083,
  "metadata": {},
  "tool_resources": {
    "code_interpreter": {
      "file_ids": []
    }
  }
}
```

Modify thread

Beta



post https://api.openai.com/v1/threads/{thread_id}

Modifies a thread.

Path parameters

The ID of the thread to modify. Only the `metadata` can be modified.

Request body

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.

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 `thread` object matching the specified ID.

Example request

```
1
2
3
4
5
6
7
8
9
10
curl https://api.openai.com/v1/threads/thread_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "metadata": {
    "modified": "true",
    "user": "abc123"
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1699014083,
  "metadata": {
    "modified": "true",
    "user": "abc123"
  },
  "tool_resources": {}
}
```

Delete thread

Beta



delete https://api.openai.com/v1/threads/{thread_id}

Delete a thread.

Path parameters

The ID of the thread to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/threads/thread_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  "id": "thread_abc123",
  "object": "thread.deleted",
  "deleted": true
}
```

The thread object

Beta



Represents a thread that contains [messages](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always [thread](#).

The Unix timestamp (in seconds) for when the thread was created.

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.

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 The thread object

```
1
2
3
4
5
6
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1698107661,
  "metadata": {}
}
```

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

The ID of the [thread](#) to create a message for.

Request body

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.



content

string or array

Required

A list of files attached to the message, and the tools they should be added to.

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

```
1
2
3
4
5
6
7
8
curl https://api.openai.com/v1/threads/thread_abc123/messages \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "role": "user",
  "content": "How does AI work? Explain it in simple terms."
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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

The ID of the thread the messages belong to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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 messages by the run ID that generated them.

Returns

A list of `message` objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/messages \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
{
  "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

The ID of the thread to which this message belongs.

The ID of the message to retrieve.

Returns

The message object matching the specified ID.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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

The ID of the thread to which this message belongs.

The ID of the message to modify.

Request body

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

```
1
2
3
4
5
6
7
8
9
10
curl https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "metadata": {
    "modified": "true",
    "user": "abc123"
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "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

The ID of the thread to which this message belongs.

The ID of the message to delete.

Returns

Deletion status

Example request

```
1
2
3
4
curl -X DELETE https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
{
  "id": "msg_abc123",
  "object": "thread.message.deleted",
  "deleted": true
}
```

The message object

Beta



Represents a message within a [thread](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always [thread.message](#).

The Unix timestamp (in seconds) for when the message was created.

The [thread](#) ID that this message belongs to.

The status of the message, which can be either [in_progress](#), [incomplete](#), or [completed](#).

On an incomplete message, details about why the message is incomplete.

The Unix timestamp (in seconds) for when the message was completed.

The Unix timestamp (in seconds) for when the message was marked as incomplete.

The entity that produced the message. One of `user` or `assistant`.

The content of the message in array of text and/or images.

If applicable, the ID of the assistant that authored this message.

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.

A list of files attached to the message, and the tools they were added to.

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 The message object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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": {}
}
```

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

The ID of the thread to run.

Query parameters

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

The ID of the [assistant](#) to use to execute this run.

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.

o1 and o3-mini models only

Constrains effort on reasoning for [reasoning models](#). Currently supported values are `low`, `medium`, and `high`. Reducing reasoning effort can result in faster responses and fewer tokens used on reasoning in a response.

Overrides the [instructions](#) of the assistant. This is useful for modifying the behavior on a per-run basis.

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.

Adds additional messages to the thread before creating the run.

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

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.

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.

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.

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.

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.

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.

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

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.

Whether to enable parallel function calling during tool use.

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.

Returns

A run object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/threads/thread_abc123/runs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "assistant_id": "asst_abc123"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
{
  "id": "run_abc123",
  "object": "thread.run",
  "created_at": 1699063290,
  "assistant_id": "asst_abc123",
  "thread_id": "thread_abc123",
  "status": "queued",
  "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"
    }
  ],
  "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  
}
```

Create thread and run

Beta



post <https://api.openai.com/v1/threads/runs>

Create a thread and run it in one request.

Request body

The ID of the assistant to use to execute this run.

Options to create a new thread. If no thread is provided when running a request, an empty thread will be created.

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.

Override the default system message of the assistant. This is useful for modifying the behavior on a per-run basis.

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

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.

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.

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.

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.

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.

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.

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.

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

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.

Whether to enable parallel function calling during tool use.

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.

Returns

A run object.

Example request

```
1
2
3
4
5
6
7
8
9
10
11
12
curl https://api.openai.com/v1/threads/runs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "assistant_id": "asst_abc123",
  "thread": {
    "messages": [
      {"role": "user", "content": "Explain deep learning to a 5 year old."}
    ]
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
{
  "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

The ID of the thread the run belongs to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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.

Returns

A list of `run` objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

```
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
{
  "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": 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

The ID of the thread that was run.

The ID of the run to retrieve.

Returns

The run object matching the specified ID.

Example request

```
1
2
3
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
{
  "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": [
    {
      "id": "tool_abc123",
      "object": "thread.tool",
      "assistant_id": "asst_abc123",
      "thread_id": "thread_abc123",
      "status": "active",
      "started_at": 1699075072,
      "expires_at": null,
      "cancelled_at": null,
      "failed_at": null,
      "completed_at": null,
      "last_error": null,
      "model": "gpt-4o",
      "instructions": null,
      "incomplete_details": null,
      "tools": []
    }
  ]
}
```

```
        "type": "code_interpreter"
    },
],
"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
}
```

Modify run

Beta



post https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}

Modifies a run.

Path parameters

The ID of the thread that was run.

The ID of the run to modify.

Request body

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

```
1
2
3
4
5
6
7
8
9
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "metadata": {
    "user_id": "user_abc123"
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
{
  "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

The ID of the thread to which this run belongs.

The ID of the run that requires the tool output submission.

Request body

A list of tools for which the outputs are being submitted.

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.

Example request

```
1
2
3
4
5
6
7
8
9
10
11
12
curl https://api.openai.com/v1/threads/thread_123/runs/run_123/submit_tool_outputs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "tool_outputs": [
    {
      "tool_call_id": "call_001",
      "output": "70 degrees and sunny."
    }
  ]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
{
  "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

The ID of the thread to which this run belongs.

The ID of the run to cancel.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123/cancel \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v2" \
-X POST
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
{
  "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](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always `thread.run`.

The Unix timestamp (in seconds) for when the run was created.

The ID of the [thread](#) that was executed on as a part of this run.

The ID of the [assistant](#) used for execution of this run.

The status of the run, which can be either `queued`, `in_progress`, `requires_action`, `cancelling`, `cancelled`, `failed`, `completed`, `incomplete`, or `expired`.

Details on the action required to continue the run. Will be `null` if no action is required.

The last error associated with this run. Will be `null` if there are no errors.

The Unix timestamp (in seconds) for when the run will expire.

The Unix timestamp (in seconds) for when the run was started.

The Unix timestamp (in seconds) for when the run was cancelled.

The Unix timestamp (in seconds) for when the run failed.

The Unix timestamp (in seconds) for when the run was completed.

Details on why the run is incomplete. Will be `null` if the run is not incomplete.

The model that the [assistant](#) used for this run.

The instructions that the [assistant](#) used for this run.

The list of tools that the [assistant](#) used for this run.

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.

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.).

The sampling temperature used for this run. If not set, defaults to 1.

The nucleus sampling value used for this run. If not set, defaults to 1.

The maximum number of prompt tokens specified to have been used over the course of the run.

The maximum number of completion tokens specified to have been used over the course of the run.

Controls for how a thread will be truncated prior to the run. Use this to control the initial context window of the run.

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.

Whether to enable parallel function calling during tool use.

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.

OBJECT The run object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
{
  "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": 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
}
```

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

The ID of the thread the run and run steps belong to.

The ID of the run the run steps belong to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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.

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

A list of `run_step` objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123/steps \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
{
  "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

The ID of the thread to which the run and run step belongs.

The ID of the run to which the run step belongs.

The ID of the run step to retrieve.

Query parameters

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

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123/steps/step_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
{
  "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
  }
}
```

The run step object

Beta



Represents a step in execution of a run.

The identifier of the run step, which can be referenced in API endpoints.

The object type, which is always `thread.run.step`.

The Unix timestamp (in seconds) for when the run step was created.

The ID of the assistant associated with the run step.

The ID of the thread that was run.

The ID of the run that this run step is a part of.

The type of run step, which can be either `message_creation` or `tool_calls`.

The status of the run step, which can be either `in_progress`, `cancelled`, `failed`, `completed`, or `expired`.

The details of the run step.

The last error associated with this run step. Will be `null` if there are no errors.

The Unix timestamp (in seconds) for when the run step expired. A step is considered expired if the parent run is expired.

The Unix timestamp (in seconds) for when the run step was cancelled.

The Unix timestamp (in seconds) for when the run step failed.

The Unix timestamp (in seconds) for when the run step completed.

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.

Usage statistics related to the run step. This value will be `null` while the run step's status is `in_progress`.

OBJECT The run step object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
{
  "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
  }
}
```

Vector stores

Beta



Vector stores are used to store files for use by the [file_search](#) tool.

Related guide: [File Search](#)

Create vector store

Beta



post https://api.openai.com/v1/vector_stores

Create a vector store.

Request body

A list of [File](#) IDs that the vector store should use. Useful for tools like [file_search](#) that can access files.

The name of the vector store.

The expiration policy for a vector store.

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.

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 [vector store](#) object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/vector_stores \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
-d '{
    "name": "Support FAQ"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "id": "vs_abc123",
  "object": "vector_store",
  "created_at": 1699061776,
  "name": "Support FAQ",
  "bytes": 139920,
  "file_counts": {
    "in_progress": 0,
    "completed": 3,
    "failed": 0,
    "cancelled": 0,
    "total": 3
  }
}
```

List vector stores

Beta



get https://api.openai.com/v1/vector_stores

Returns a list of vector stores.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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.

Returns

A list of [vector store](#) objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/vector_stores \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
{
  "object": "list",
  "data": [
    {
      "id": "vs_abc123",
      "object": "vector_store",
      "created_at": 1699061776,
      "name": "Support FAQ",
      "bytes": 139920,
      "file_counts": {
        "in_progress": 0,
        "completed": 3,
        "failed": 0,
        "cancelled": 0,
        "total": 3
      }
    },
    {
      "id": "vs_abc456",
      "object": "vector_store",
      "created_at": 1699061776,
```

```
        "name": "Support FAQ v2",
        "bytes": 139920,
        "file_counts": {
            "in_progress": 0,
            "completed": 3,
            "failed": 0,
            "cancelled": 0,
            "total": 3
        }
    },
],
"first_id": "vs_abc123",
"last_id": "vs_abc456",
"has_more": false
}
```

Retrieve vector store

Beta



get https://api.openai.com/v1/vector_stores/{vector_store_id}

Retrieves a vector store.

Path parameters

The ID of the vector store to retrieve.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/vector_stores/vs_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
{
  "id": "vs_abc123",
  "object": "vector_store",
  "created_at": 1699061776
}
```

Modify vector store

Beta



post https://api.openai.com/v1/vector_stores/{vector_store_id}

Modifies a vector store.

Path parameters

The ID of the vector store to modify.

Request body

The name of the vector store.

The expiration policy for a vector store.

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 vector store object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/vector_stores/vs_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
-d '{
  "name": "Support FAQ"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "id": "vs_abc123",
  "object": "vector_store",
  "created_at": 1699061776,
  "name": "Support FAQ",
  "bytes": 139920,
  "file_counts": {
    "in_progress": 0,
    "completed": 3,
    "failed": 0,
    "cancelled": 0,
    "total": 3
  }
}
```

Delete vector store Beta



```
delete https://api.openai.com/v1/vector_stores/{vector_store_id}
```

Delete a vector store.

Path parameters

The ID of the vector store to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/vector_stores/vs_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  id: "vs_abc123",
  object: "vector_store.deleted",
  deleted: true
}
```

The vector store object

Beta



A vector store is a collection of processed files can be used by the [file_search](#) tool.

The identifier, which can be referenced in API endpoints.

The object type, which is always [vector_store](#).

The Unix timestamp (in seconds) for when the vector store was created.

The name of the vector store.

The total number of bytes used by the files in the vector store.

[file_counts](#)

object

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.

The expiration policy for a vector store.

The Unix timestamp (in seconds) for when the vector store will expire.

The Unix timestamp (in seconds) for when the vector store was last active.

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 The vector store object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "id": "vs_123",
  "object": "vector_store",
  "created_at": 1698107661,
  "usage_bytes": 123456,
  "last_active_at": 1698107661,
  "name": "my_vector_store",
  "status": "completed",
  "file_counts": {
    "in_progress": 0,
    "completed": 100,
    "cancelled": 0,
    "failed": 0,
    "total": 100
  },
  "metadata": {},
  "last_used_at": 1698107661
}
```

Vector store files

Beta



Vector store files represent files inside a vector store.

Related guide: [File Search](#)

Create vector store file

Beta



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

The ID of the vector store for which to create a File.

Request body

A [File](#) ID that the vector store should use. Useful for tools like [file_search](#) that can access files.

The chunking strategy used to chunk the file(s). If not set, will use the [auto](#) strategy.

Returns

A [vector store file](#) object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/vector_stores/vs_abc123/files \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "file_id": "file-abc123"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
{
  "id": "file-abc123",
  "object": "vector_store.file",
  "created_at": 1699061776,
  "usage_bytes": 1234,
  "vector_store_id": "vs_abcd",
  "status": "completed",
  "last_error": null
}
```

List vector store files

Beta



get https://api.openai.com/v1/vector_stores/{vector_store_id}/files

Returns a list of vector store files.

Path parameters

The ID of the vector store that the files belong to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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 by file status. One of `in_progress`, `completed`, `failed`, `cancelled`.

Returns

A list of vector store file objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/vector_stores/vs_abc123/files \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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
}
```

Retrieve vector store file Beta



get https://api.openai.com/v1/vector_stores/{vector_store_id}/files/{file_id}

Retrieves a vector store file.

Path parameters

The ID of the vector store that the file belongs to.

The ID of the file being retrieved.

Returns

The vector store file object.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/vector_stores/vs_abc123/files/file-abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
{
  "id": "file-abc123",
  "object": "vector_store.file",
  "created_at": 1699061776,
  "vector_store_id": "vs_abcd",
  "status": "completed",
  "last_error": null
}
```

Delete vector store file

Beta



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

The ID of the vector store that the file belongs to.

The ID of the file to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/vector_stores/vs_abc123/files/file-abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  id: "file-abc123",
  object: "vector_store.file.deleted",
  deleted: true
}
```

The vector store file object

Beta



A list of files attached to a vector store.

The identifier, which can be referenced in API endpoints.

The object type, which is always `vector_store.file`.

The total vector store usage in bytes. Note that this may be different from the original file size.

The Unix timestamp (in seconds) for when the vector store file was created.

The ID of the vector store that the File is attached to.

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.

The last error associated with this vector store file. Will be `null` if there are no errors.

The strategy used to chunk the file.

OBJECT The vector store file object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
16
{
  "id": "file-abc123",
  "object": "vector_store.file",
  "usage_bytes": 1234,
  "created_at": 1698107661,
  "vector_store_id": "vs_abc123",
  "status": "completed",
  "last_error": null,
  "chunking_strategy": {
    "type": "static",
    "static": {
      "max_chunk_size_tokens": 800,
      "chunk_overlap_tokens": 400
    }
  }
}
```

Vector store file batches

Beta



Vector store file batches represent operations to add multiple files to a vector store. Related guide:
[File Search](#)

Create vector store file batch

Beta



post https://api.openai.com/v1/vector_stores/{vector_store_id}/file_batches

Create a vector store file batch.

Path parameters

The ID of the vector store for which to create a File Batch.

Request body

A list of [File](#) IDs that the vector store should use. Useful for tools like [file_search](#) that can access files.

The chunking strategy used to chunk the file(s). If not set, will use the [auto](#) strategy.

Returns

A [vector store file batch](#) object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/vector_stores/vs_abc123/file_batches \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-d '{
  "file_ids": ["file-abc123", "file-abc456"]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "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

Beta



get https://api.openai.com/v1/vector_stores/{vector_store_id}/file_batches/{batch_id}

Retrieves a vector store file batch.

Path parameters

The ID of the vector store that the file batch belongs to.

The ID of the file batch being retrieved.

Returns

The [vector store file batch](#) object.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/vector_stores/vs_abc123/files_batches/vsfb_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "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 Beta



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

The ID of the vector store that the file batch belongs to.

The ID of the file batch to cancel.

Returns

The modified vector store file batch object.

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/vector_stores/vs_abc123/files_batches/vsfb_abc123/cancel \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2" \
-X POST
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "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

Beta



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

The ID of the vector store that the files belong to.

The ID of the file batch that the files belong to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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 by file status. One of `in_progress`, `completed`, `failed`, `cancelled`.

Returns

A list of vector store file objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/vector_stores/vs_abc123/files_batches/vsfb_abc123/files \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v2"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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.

The identifier, which can be referenced in API endpoints.

The object type, which is always `vector_store.file_batch`.

The Unix timestamp (in seconds) for when the vector store files batch was created.

The ID of the vector store that the File is attached to.

The status of the vector store files batch, which can be either `in_progress`, `completed`, `cancelled` or `failed`.

🔗
file_counts

object

OBJECT The vector store files batch object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "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
  }
}
```

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.

The identifier of the message, which can be referenced in API endpoints.

The object type, which is always `thread.message.delta`.

The delta containing the fields that have changed on the Message.

OBJECT The message delta object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
{
  "id": "msg_123",
  "object": "thread.message.delta",
  "delta": {
    "content": [
      {
        "index": 0,
        "type": "text",
        "text": { "value": "Hello", "annotations": [] }
      }
    ]
  }
}
```

The run step delta object

Beta



Represents a run step delta i.e. any changed fields on a run step during streaming.

The identifier of the run step, which can be referenced in API endpoints.

The object type, which is always `thread.run.step.delta`.

The delta containing the fields that have changed on the run step.

OBJECT The run step delta object

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
{
  "id": "step_123",
  "object": "thread.run.step.delta",
  "delta": {
    "step_details": {
      "type": "tool_calls",
      "tool_calls": [
        {
          "index": 0,
          "id": "call_123",
          "type": "code_interpreter",
          "code_interpreter": { "input": "", "outputs": [] }
        }
      ]
    }
  }
}

```

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.

Occurs when a new thread is created.

Occurs when a new run is created.

Occurs when a run moves to a queued status.

Occurs when a run moves to an in_progress status.

Occurs when a run moves to a requires_action status.

Occurs when a run is completed.

Occurs when a run ends with status incomplete.

Occurs when a run fails.

Occurs when a run moves to a cancelling status.

Occurs when a run is cancelled.

Occurs when a run expires.

Occurs when a run step is created.

Occurs when a run step moves to an in_progress state.

Occurs when parts of a run step are being streamed.

Occurs when a run step is completed.

Occurs when a run step fails.

Occurs when a run step is cancelled.

Occurs when a run step expires.

Occurs when a message is created.

Occurs when a message moves to an in_progress state.

Occurs when parts of a Message are being streamed.

Occurs when a message is completed.

Occurs when a message ends before it is completed.

Occurs when an error occurs. This can happen due to an internal server error or a timeout.

Occurs when a stream ends.

Administration



Programmatically manage your organization. The Audit Logs endpoint provides a log of all actions taken in the organization for security and monitoring purposes. To access these endpoints please generate an Admin API Key through the [API Platform Organization overview](#). Admin API keys cannot be used for non-administration endpoints. For best practices on setting up your organization, please refer to this [guide](#)

Admin API Keys



The **Usage API** provides detailed insights into your activity across the OpenAI API. It also includes a separate [Costs endpoint](#), which offers visibility into your spend, breaking down consumption by invoice line items and project IDs. While the Usage API delivers granular usage data, it may not always reconcile perfectly with the Costs due to minor differences in how usage and spend are recorded. For financial purposes, we recommend using the [Costs endpoint](#) or the [Costs tab](#) in the Usage Dashboard, which will reconcile back to your billing invoice.

List admin API keys



get https://api.openai.com/v1/organization/admin_api_keys

List organization API keys

Query parameters

after
string or null

Optional

order
string

Optional

Defaults to asc

limit

integer

Optional

Defaults to 20

Returns

A list of admin API key objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/admin_api_keys?after=key_abc&limit=20 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "object": "list",
  "data": [
    {
      "object": "organization.admin_api_key",
      "id": "key_abc",
      "name": "Main Admin Key",
      "redacted_value": "sk-admin...def",
      "created_at": 1711471533,
      "owner": {
        "type": "service_account",
        "object": "organization.service_account",
        "id": "sa_456",
        "name": "My Service Account",
        "created_at": 1711471533,
        "role": "member"
      }
    }
  ],
  "first_id": "key_abc",
  "last_id": "key_abc",
  "has_more": false
}
```

Create admin API key

⌚
post https://api.openai.com/v1/organization/admin_api_keys

Create an organization admin API key

Request body

 name

string

Required

Returns

The created admin API key object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/admin_api_keys \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "name": "New Admin Key"
}'
```

Response

Retrieve admin API key

get https://api.openai.com/v1/organization/admin_api_keys/{key_id}

Retrieve a single organization API key

Path parameters

 key_id

string

Required

Returns

The requested admin API key object.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/admin_api_keys/key_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
{
  "object": "organization.admin_api_key",
  "id": "key_abc",
  "name": "Main Admin Key",
  "redacted_value": "sk-admin...xyz",
  "created_at": 1711471533,
  "owner": {
    "type": "user",
    "object": "organization.user",
    "id": "user_123",
    "name": "John Doe",
    "created_at": 1711471533,
    "role": "owner"
  }
}
```

Delete admin API key

🔗 [delete https://api.openai.com/v1/organization/admin_api_keys/{key_id}](https://api.openai.com/v1/organization/admin_api_keys/{key_id})

Delete an organization admin API key

Path parameters

🔗 **key_id**

string

Required

Returns

A confirmation object indicating the key was deleted.

Example request

```
1
2
3
curl -X DELETE https://api.openai.com/v1/organization/admin_api_keys/key_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
{
  "id": "key_abc",
  "object": "organization.admin_api_key.deleted",
  "deleted": true
}
```

Invites

 [Invite and manage invitations for an organization.](#)

List invites

 [get https://api.openai.com/v1/organization/invites](#)

Returns a list of invites in the organization.

Query parameters

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

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.

Returns

A list of [Invite](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/invites?after=invite-abc&limit=20 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "object": "list",
  "data": [
    {
      "object": "organization.invite",
      "id": "invite-abc",
      "email": "user@example.com",
      "role": "owner",
      "status": "accepted",
      "invited_at": 1711471533,
      "expires_at": 1711471533,
      "accepted_at": 1711471533
    }
  ],
  "first_id": "invite-abc",
  "last_id": "invite-abc",
  "has_more": false
}
```

Create invite

🔗
post <https://api.openai.com/v1/organization/invites>

Create an invite for a user to the organization. The invite must be accepted by the user before they have access to the organization.

Request body

Send an email to this address

owner or reader

An array of projects to which membership is granted at the same time the org invite is accepted. If omitted, the user will be invited to the default project for compatibility with legacy behavior.

Returns

The created [Invite](#) object.

Example request

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
curl -X POST https://api.openai.com/v1/organization/invites \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
  "email": "anotheruser@example.com",
  "role": "reader",
  "projects": [
    {
      "id": "project-xyz",
      "role": "member"
    },
    {
      "id": "project-abc",
      "role": "owner"
    }
  ]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "object": "organization.invite",
  "id": "invite-def",
  "email": "anotheruser@example.com",
  "role": "reader",
  "status": "pending",
  "invited_at": 1711471533,
  "expires_at": 1711471533,
  "accepted_at": null,
  "projects": [
    {
      "id": "project-xyz",
      "role": "member"
    },
    {
      "id": "project-abc",
      "role": "owner"
    }
  ]
}
```

Retrieve invite

⌚
get https://api.openai.com/v1/organization/invites/{invite_id}

Retrieves an invite.

Path parameters

The ID of the invite to retrieve.

Returns

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

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/invites/invite-abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
{
  "object": "organization.invite",
  "id": "invite-abc",
  "email": "user@example.com",
  "role": "owner",
  "status": "accepted",
  "invited_at": 1711471533,
  "expires_at": 1711471533,
  "accepted_at": 1711471533
}
```

Delete invite

🔗
delete https://api.openai.com/v1/organization/invites/{invite_id}

Delete an invite. If the invite has already been accepted, it cannot be deleted.

Path parameters

The ID of the invite to delete.

Returns

Confirmation that the invite has been deleted

Example request

```
1
2
3
curl -X DELETE https://api.openai.com/v1/organization/invites/invite-abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
{
  "object": "organization.invite.deleted",
  "id": "invite-abc",
  "deleted": true
}
```

The invite object



Represents an individual `invite` to the organization.

The object type, which is always `organization.invite`

The identifier, which can be referenced in API endpoints

The email address of the individual to whom the invite was sent

`owner` or `reader`

`accepted`,`expired`, or `pending`

The Unix timestamp (in seconds) of when the invite was sent.

The Unix timestamp (in seconds) of when the invite expires.

The Unix timestamp (in seconds) of when the invite was accepted.

The projects that were granted membership upon acceptance of the invite.

OBJECT The invite object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
{
  "object": "organization.invite",
  "id": "invite-abc",
  "email": "user@example.com",
  "role": "owner",
  "status": "accepted",
  "invited_at": 1711471533,
  "expires_at": 1711471533,
  "accepted_at": 1711471533,
  "projects": [
    {
      "id": "project-xyz",
      "role": "member"
    }
  ]
}
```

Users



Manage users and their role in an organization.

List users



get <https://api.openai.com/v1/organization/users>

Lists all of the users in the organization.

Query parameters

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

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.

Filter by the email address of users.

Returns

A list of User objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/users?after=user_abc&limit=20 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
{
  "object": "list",
  "data": [
    {
      "object": "organization.user",
      "id": "user_abc",
      "name": "First Last",
      "email": "user@example.com",
      "role": "owner",
      "added_at": 1711471533
    }
  ],
  "first_id": "user-abc",
  "last_id": "user-xyz",
  "has_more": false
}
```

Modify user

⌚
post https://api.openai.com/v1/organization/users/{user_id}

Modifies a user's role in the organization.

Path parameters

The ID of the user.

Request body

owner or reader

Returns

The updated [User](#) object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/users/user_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "role": "owner"
}'
```

Response

```
1
2
3
4
5
6
7
8
{
    "object": "organization.user",
    "id": "user_abc",
    "name": "First Last",
    "email": "user@example.com",
    "role": "owner",
    "added_at": 1711471533
}
```

Retrieve user

🔗
get https://api.openai.com/v1/organization/users/{user_id}

Retrieves a user by their identifier.

Path parameters

The ID of the user.

Returns

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

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/users/user_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
{
  "object": "organization.user",
  "id": "user_abc",
  "name": "First Last",
  "email": "user@example.com",
  "role": "owner",
  "added_at": 1711471533
}
```

Delete user

🔗
delete https://api.openai.com/v1/organization/users/{user_id}

Deletes a user from the organization.

Path parameters

The ID of the user.

Returns

Confirmation of the deleted user

Example request

```
1
2
3
curl -X DELETE https://api.openai.com/v1/organization/users/user_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
{
  "object": "organization.user.deleted",
  "id": "user_abc",
  "deleted": true
}
```

The user object



Represents an individual `user` within an organization.

The object type, which is always `organization.user`

The identifier, which can be referenced in API endpoints

The name of the user

The email address of the user

`owner` or `reader`

The Unix timestamp (in seconds) of when the user was added.

OBJECT The user object

```
1
2
3
4
5
6
7
8
{
  "object": "organization.user",
  "id": "user_abc",
  "name": "First Last",
  "email": "user@example.com",
  "role": "owner",
  "added_at": 1711471533
}
```

Projects



Manage the projects within an organization includes creation, updating, and archiving or projects. The Default project cannot be archived.

List projects



get <https://api.openai.com/v1/organization/projects>

Returns a list of projects.

Query parameters

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

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.

If `true` returns all projects including those that have been `archived`. Archived projects are not included by default.

Returns

A list of [Project](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects?
after=proj_abc&limit=20&include_archived=false \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
{
  "object": "list",
  "data": [
    {
      "id": "proj_abc",
      "object": "organization.project",
      "name": "Project example",
      "created_at": 1711471533,
      "archived_at": null,
      "status": "active"
    }
  ],
  "first_id": "proj-abc",
  "last_id": "proj-xyz",
  "has_more": false
}
```

Create project

🔗
post <https://api.openai.com/v1/organization/projects>

Create a new project in the organization. Projects can be created and archived, but cannot be deleted.

Request body

The friendly name of the project, this name appears in reports.

Returns

The created [Project](#) object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/projects \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "name": "Project ABC"
}'
```

Response

```
1
2
3
4
5
6
7
8
{
    "id": "proj_abc",
    "object": "organization.project",
    "name": "Project ABC",
    "created_at": 1711471533,
    "archived_at": null,
    "status": "active"
}
```

Retrieve project

🔗
get https://api.openai.com/v1/organization/projects/{project_id}

Retrieves a project.

Path parameters

The ID of the project.

Returns

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

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
{
  "id": "proj_abc",
  "object": "organization.project",
  "name": "Project example",
  "created_at": 1711471533,
  "archived_at": null,
  "status": "active"
}
```

Modify project

⌚
post https://api.openai.com/v1/organization/projects/{project_id}

Modifies a project in the organization.

Path parameters

The ID of the project.

Request body

The updated name of the project, this name appears in reports.

Returns

The updated [Project](#) object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/projects/proj_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "name": "Project DEF"
}'
```

Archive project

⌚
post https://api.openai.com/v1/organization/projects/{project_id}/archive

Archives a project in the organization. Archived projects cannot be used or updated.

Path parameters

The ID of the project.

Returns

The archived [Project](#) object.

Example request

```
1
2
3
curl -X POST https://api.openai.com/v1/organization/projects/proj_abc/archive \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
{
  "id": "proj_abc",
  "object": "organization.project",
  "name": "Project DEF",
  "created_at": 1711471533,
  "archived_at": 1711471533,
  "status": "archived"
}
```

The project object



Represents an individual project.

The identifier, which can be referenced in API endpoints

The object type, which is always `organization.project`

The name of the project. This appears in reporting.

The Unix timestamp (in seconds) of when the project was created.

The Unix timestamp (in seconds) of when the project was archived or `null`.

`active` or `archived`

OBJECT The project object

```
1
2
3
4
5
6
7
8
{
  "id": "proj_abc",
  "object": "organization.project",
  "name": "Project example",
  "created_at": 1711471533,
  "archived_at": null,
  "status": "active"
}
```

Project users



Manage users within a project, including adding, updating roles, and removing users.

List project users



get https://api.openai.com/v1/organization/projects/{project_id}/users

Returns a list of users in the project.

Path parameters

The ID of the project.

Query parameters

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

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.

Returns

A list of [ProjectUser](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/users?after=user_abc&limit=20 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
{
  "object": "list",
  "data": [
    {
      "object": "organization.project.user",
      "id": "user_abc",
      "name": "First Last",
      "email": "user@example.com",
      "role": "owner",
      "added_at": 1711471533
    }
  ],
  "first_id": "user-abc",
  "last_id": "user-xyz",
  "has_more": false
}
```

Create project user

⌚
post https://api.openai.com/v1/organization/projects/{project_id}/users

Adds a user to the project. Users must already be members of the organization to be added to a project.

Path parameters

The ID of the project.

Request body

The ID of the user.

owner or member

Returns

The created [ProjectUser](#) object.

Example request

```
1
2
3
4
5
6
7
curl -X POST https://api.openai.com/v1/organization/projects/proj_abc/users \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "user_id": "user_abc",
    "role": "member"
}'
```

Response

```
1
2
3
4
5
6
7
{
    "object": "organization.project.user",
    "id": "user_abc",
    "email": "user@example.com",
    "role": "owner",
    "added_at": 1711471533
}
```

Retrieve project user

⌚
get https://api.openai.com/v1/organization/projects/{project_id}/users/{user_id}

Retrieves a user in the project.

Path parameters

The ID of the project.

The ID of the user.

Returns

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

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/users/user_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
{
  "object": "organization.project.user",
  "id": "user_abc",
  "name": "First Last",
  "email": "user@example.com",
  "role": "owner",
  "added_at": 1711471533
}
```

Modify project user

🔗
post https://api.openai.com/v1/organization/projects/{project_id}/users/{user_id}

Modifies a user's role in the project.

Path parameters

The ID of the project.

The ID of the user.

Request body

owner or member

Returns

The updated [ProjectUser](#) object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/projects/proj_abc/users/user_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "role": "owner"
}'
```

Response

```
1
2
3
4
5
6
7
8
{
    "object": "organization.project.user",
    "id": "user_abc",
    "name": "First Last",
    "email": "user@example.com",
    "role": "owner",
    "added_at": 1711471533
}
```

Delete project user

⌚
delete https://api.openai.com/v1/organization/projects/{project_id}/users/{user_id}

Deletes a user from the project.

Path parameters

The ID of the project.

The ID of the user.

Returns

Confirmation that project has been deleted or an error in case of an archived project, which has no users

Example request

```
1
2
3
curl -X DELETE https://api.openai.com/v1/organization/projects/proj_abc/users/user_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
{
  "object": "organization.project.user.deleted",
  "id": "user_abc",
  "deleted": true
}
```

The project user object



Represents an individual user in a project.

The object type, which is always `organization.project.user`

The identifier, which can be referenced in API endpoints

The name of the user

The email address of the user

`owner` or `member`

The Unix timestamp (in seconds) of when the project was added.

OBJECT The project user object

```
1
2
3
4
5
6
7
8
{
  "object": "organization.project.user",
  "id": "user_abc",
  "name": "First Last",
  "email": "user@example.com",
  "role": "owner",
  "added_at": 1711471533
}
```

Project service accounts



Manage service accounts within a project. A service account is a bot user that is not associated with a user. If a user leaves an organization, their keys and membership in projects will no longer work. Service accounts do not have this limitation. However, service accounts can also be deleted from a project.

List project service accounts



get https://api.openai.com/v1/organization/projects/{project_id}/service_accounts

Returns a list of service accounts in the project.

Path parameters

The ID of the project.

Query parameters

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

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.

Returns

A list of [ProjectServiceAccount](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/service_accounts?
after=custom_id&limit=20 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
{
  "object": "list",
  "data": [
    {
      "object": "organization.project.service_account",
      "id": "svc_acct_abc",
      "name": "Service Account",
      "role": "owner",
      "created_at": 1711471533
    }
  ],
  "first_id": "svc_acct_abc",
  "last_id": "svc_acct_xyz",
  "has_more": false
}
```

Create project service account

⌚
post https://api.openai.com/v1/organization/projects/{project_id}/service_accounts

Creates a new service account in the project. This also returns an unredacted API key for the service account.

Path parameters

The ID of the project.

Request body

The name of the service account being created.

Returns

The created [ProjectServiceAccount](#) object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/projects/proj_abc/service_accounts \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "name": "Production App"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
  "object": "organization.project.service_account",
  "id": "svc_acct_abc",
  "name": "Production App",
  "role": "member",
  "created_at": 1711471533,
  "api_key": {
    "object": "organization.project.service_account.api_key",
    "value": "sk-abcdefghijklmnop123",
    "name": "Secret Key",
    "created_at": 1711471533,
    "id": "key_abc"
  }
}
```

Retrieve project service account



get https://api.openai.com/v1/organization/projects/{project_id}/service_accounts/{service_account_id}

Retrieves a service account in the project.

Path parameters

The ID of the project.

The ID of the service account.

Returns

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

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/service_accounts/svc_acct_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
{
  "object": "organization.project.service_account",
  "id": "svc_acct_abc",
  "name": "Service Account",
  "role": "owner",
  "created_at": 1711471533
}
```

Delete project service account



delete https://api.openai.com/v1/organization/projects/{project_id}/service_accounts/{service_account_id}

Deletes a service account from the project.

Path parameters

The ID of the project.

The ID of the service account.

Returns

Confirmation of service account being deleted, or an error in case of an archived project, which has no service accounts

Example request

```
1
2
3
curl -X DELETE
https://api.openai.com/v1/organization/projects/proj_abc/service_accounts/svc_acct_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
{
  "object": "organization.project.service_account.deleted",
  "id": "svc_acct_abc",
  "deleted": true
}
```

The project service account object



Represents an individual service account in a project.

The object type, which is always `organization.project.service_account`

The identifier, which can be referenced in API endpoints

The name of the service account

`owner` or `member`

The Unix timestamp (in seconds) of when the service account was created

OBJECT The project service account object

```
1
2
3
4
5
6
7
{
  "object": "organization.project.service_account",
  "id": "svc_acct_abc",
  "name": "Service Account",
  "role": "owner",
  "created_at": 1711471533
}
```

Project API keys



Manage API keys for a given project. Supports listing and deleting keys for users. This API does not allow issuing keys for users, as users need to authorize themselves to generate keys.

List project API keys



get https://api.openai.com/v1/organization/projects/{project_id}/api_keys

Returns a list of API keys in the project.

Path parameters

The ID of the project.

Query parameters

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

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.

Returns

A list of [ProjectApiKey](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/api_keys?after=key_abc&limit=20
\
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
{
  "object": "list",
  "data": [
    {
      "object": "organization.project.api_key",
      "redacted_value": "sk-abc...def",
      "name": "My API Key",
      "created_at": 1711471533,
      "id": "key_abc",
      "owner": {
        "type": "user",
        "user": {
          "object": "organization.project.user",
          "id": "user_abc",
          "name": "First Last",
          "email": "user@example.com",
          "role": "owner",
          "added_at": 1711471533
        }
      }
    ],
    "first_id": "key_abc",
    "last_id": "key_xyz",
    "has_more": false
  }
}
```

Retrieve project API key



get https://api.openai.com/v1/organization/projects/{project_id}/api_keys/{key_id}

Retrieves an API key in the project.

Path parameters

The ID of the project.

The ID of the API key.

Returns

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

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/api_keys/key_abc \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "object": "organization.project.api_key",
  "redacted_value": "sk-abc...def",
  "name": "My API Key",
  "created_at": 1711471533,
  "id": "key_abc",
  "owner": {
    "type": "user",
    "user": {
      "object": "organization.project.user",
      "id": "user_abc",
      "name": "First Last",
      "email": "user@example.com",
      "role": "owner",
      "added_at": 1711471533
    }
  }
}
```

Delete project API key

🔗 [delete https://api.openai.com/v1/organization/projects/{project_id}/api_keys/{key_id}](https://api.openai.com/v1/organization/projects/{project_id}/api_keys/{key_id})

Deletes an API key from the project.

Path parameters

The ID of the project.

The ID of the API key.

Returns

Confirmation of the key's deletion or an error if the key belonged to a service account

Example request

```
1  
2  
3  
curl -X DELETE https://api.openai.com/v1/organization/projects/proj_abc/api_keys/key_abc \  
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \  
-H "Content-Type: application/json"
```

Response

```
1  
2  
3  
4  
5  
{  
  "object": "organization.project.api_key.deleted",  
  "id": "key_abc",  
  "deleted": true  
}
```

The project API key object



Represents an individual API key in a project.

The object type, which is always `organization.project.api_key`

The redacted value of the API key

The name of the API key

The Unix timestamp (in seconds) of when the API key was created

The identifier, which can be referenced in API endpoints

owner

object

OBJECT The project API key object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "object": "organization.project.api_key",
  "redacted_value": "sk-abc...def",
  "name": "My API Key",
  "created_at": 1711471533,
  "id": "key_abc",
  "owner": {
    "type": "user",
    "user": {
      "object": "organization.project.user",
      "id": "user_abc",
      "name": "First Last",
      "email": "user@example.com",
      "role": "owner",
      "created_at": 1711471533
    }
  }
}
```

Project rate limits



Manage rate limits per model for projects. Rate limits may be configured to be equal to or lower than the organization's rate limits.

List project rate limits



get https://api.openai.com/v1/organization/projects/{project_id}/rate_limits

Returns the rate limits per model for a project.

Path parameters

The ID of the project.

Query parameters

A limit on the number of objects to be returned. The default is 100.

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.

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, beginning with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Returns

A list of [ProjectRateLimit](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/projects/proj_abc/rate_limits?
after=rl_xxx&limit=20 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
{
  "object": "list",
  "data": [
    {
      "object": "project.rate_limit",
      "id": "rl-ada",
      "model": "ada",
      "max_requests_per_1_minute": 600,
      "max_tokens_per_1_minute": 150000,
      "max_images_per_1_minute": 10
    }
  ],
  "first_id": "rl-ada",
  "last_id": "rl-ada",
  "has_more": false
}
```

Modify project rate limit

🔗
post https://api.openai.com/v1/organization/projects/{project_id}/rate_limits/{rate_limit_id}

Updates a project rate limit.

Path parameters

The ID of the project.

The ID of the rate limit.

Request body

The maximum requests per minute.

The maximum tokens per minute.

The maximum images per minute. Only relevant for certain models.

The maximum audio megabytes per minute. Only relevant for certain models.

The maximum requests per day. Only relevant for certain models.

The maximum batch input tokens per day. Only relevant for certain models.

Returns

The updated [ProjectRateLimit](#) object.

Example request

```
1
2
3
4
5
6
curl -X POST https://api.openai.com/v1/organization/projects/proj_abc/rate_limits/rl_xxx \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json" \
-d '{
    "max_requests_per_1_minute": 500
}'
```

Response

```
1
2
3
4
5
6
7
8
{
    "object": "project.rate_limit",
    "id": "rl-ada",
    "model": "ada",
    "max_requests_per_1_minute": 600,
    "max_tokens_per_1_minute": 150000,
    "max_images_per_1_minute": 10
}
```

The project rate limit object



Represents a project rate limit config.

The object type, which is always `project.rate_limit`

The identifier, which can be referenced in API endpoints.

The model this rate limit applies to.

The maximum requests per minute.

The maximum tokens per minute.

The maximum images per minute. Only present for relevant models.

The maximum audio megabytes per minute. Only present for relevant models.

The maximum requests per day. Only present for relevant models.

The maximum batch input tokens per day. Only present for relevant models.

OBJECT The project rate limit object

```
1
2
3
4
5
6
7
8
{
  "object": "project.rate_limit",
  "id": "rl_ada",
  "model": "ada",
  "max_requests_per_1_minute": 600,
  "max_tokens_per_1_minute": 150000,
  "max_images_per_1_minute": 10
}
```

Audit logs



Logs of user actions and configuration changes within this organization. To log events, you must activate logging in the [Organization Settings](#). Once activated, for security reasons, logging cannot be deactivated.

List audit logs



get https://api.openai.com/v1/organization/audit_logs

List user actions and configuration changes within this organization.

Query parameters

Return only events whose `effective_at` (Unix seconds) is in this range.

Return only events for these projects.

Return only events with a `type` in one of these values. For example, `project.created`. For all options, see the documentation for the [audit log object](#).

Return only events performed by these actors. Can be a user ID, a service account ID, or an api key tracking ID.

Return only events performed by users with these emails.

Return only events performed on these targets. For example, a project ID updated.

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

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.

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.

Returns

A list of paginated [Audit Log](#) objects.

Example request

```
1
2
3
curl https://api.openai.com/v1/organization/audit_logs \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

```

58
59
{
  "object": "list",
  "data": [
    {
      "id": "audit_log-xxx_yyyyymmdd",
      "type": "project.archived",
      "effective_at": 1722461446,
      "actor": {
        "type": "api_key",
        "api_key": {
          "type": "user",
          "user": {
            "id": "user-xxx",
            "email": "user@example.com"
          }
        }
      },
      "project.archived": {
        "id": "proj_abc"
      },
    },
    {
      "id": "audit_log-yyy__20240101",
      "type": "api_key.updated",
      "effective_at": 1720804190,
      "actor": {
        "type": "session",
        "session": {
          "user": {
            "id": "user-xxx",
            "email": "user@example.com"
          },
          "ip_address": "127.0.0.1",
          "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36",
          "ja3": "a497151ce4338a12c4418c44d375173e",
          "ja4": "q13d0313h3_55b375c5d22e_c7319ce65786",
          "ip_address_details": {
            "country": "US",
            "city": "San Francisco",
            "region": "California",
            "region_code": "CA",
            "asn": "1234",
            "latitude": "37.77490",
            "longitude": "-122.41940"
          }
        }
      },
      "api_key.updated": {
        "id": "key_xxxx",
        "data": {
          "scopes": ["resource_2.operation_2"]
        }
      },
    }
  ]
}

```

```
],
"first_id": "audit_log-xxx_20240101",
"last_id": "audit_logyyy_20240101",
"has_more": true
}
```

The audit log object

⌚
A log of a user action or configuration change within this organization.

The ID of this log.

The event type.

The Unix timestamp (in seconds) of the event.

The project that the action was scoped to. Absent for actions not scoped to projects.

The actor who performed the audit logged action.

The details for events with this [type](#).

The details for events with this **type**.

OBJECT The audit log object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
{
  "id": "req_xxx_20240101",
  "type": "api_key.created",
  "effective_at": 1720804090,
  "actor": {
    "type": "session",
    "session": {
      "user": {
        "id": "user-xxx",
        "email": "user@example.com"
      },
      "ip_address": "127.0.0.1",
      "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
    }
  },
  "api_key.created": {
    "id": "key_xxxx",
    "data": {
      "scopes": ["resource.operation"]
    }
  }
}
```

Usage



The **Usage API** provides detailed insights into your activity across the OpenAI API. It also includes a separate [Costs endpoint](#), which offers visibility into your spend, breaking down consumption by invoice line items and project IDs.

While the Usage API delivers granular usage data, it may not always reconcile perfectly with the Costs due to minor differences in how usage and spend are recorded. For financial purposes, we recommend using the [Costs endpoint](#) or the [Costs tab](#) in the Usage Dashboard, which will reconcile back to your billing invoice.

Completions



get <https://api.openai.com/v1/organization/usage/completions>

Get completions usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usage for these projects.

Return only usage for these users.

Return only usage for these API keys.

Return only usage for these models.

If `true`, return batch jobs only. If `false`, return non-batch jobs only. By default, return both.

Group the usage data by the specified fields. Support fields include `project_id`, `user_id`, `api_key_id`, `model`, `batch` or any combination of them.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Completions usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/completions?start_time=1730419200&limit=1"
\
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.completions.result",
          "input_tokens": 1000,
          "output_tokens": 500,
          "input_cached_tokens": 800,
          "input_audio_tokens": 0,
          "output_audio_tokens": 0,
          "num_model_requests": 5,
          "project_id": null,
          "user_id": null,
          "api_key_id": null,
          "model": null,
          "batch": null
        }
      ]
    }
  ],
  "has_more": true,
  "next_page": "page_AAAAAGdGxdEiJdKOAAAAAGcqsYA="
}
```

Completions usage object

⌚

The aggregated completions usage details of the specific time bucket.

⌚

object

string

The aggregated number of text input tokens used, including cached tokens. For customers subscribe to scale tier, this includes scale tier tokens.

The aggregated number of text input tokens that has been cached from previous requests. For customers subscribe to scale tier, this includes scale tier tokens.

The aggregated number of text output tokens used. For customers subscribe to scale tier, this includes scale tier tokens.

The aggregated number of audio input tokens used, including cached tokens.

The aggregated number of audio output tokens used.

The count of requests made to the model.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

When `group_by=user_id`, this field provides the user ID of the grouped usage result.

When `group_by=api_key_id`, this field provides the API key ID of the grouped usage result.

When `group_by=model`, this field provides the model name of the grouped usage result.

When `group_by=batch`, this field tells whether the grouped usage result is batch or not.

OBJECT Completions usage object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
{
    "object": "organization.usage.completions.result",
    "input_tokens": 5000,
    "output_tokens": 1000,
    "input_cached_tokens": 4000,
    "input_audio_tokens": 300,
    "output_audio_tokens": 200,
    "num_model_requests": 5,
    "project_id": "proj_abc",
    "user_id": "user-abc",
    "api_key_id": "key_abc",
    "model": "gpt-4o-mini-2024-07-18",
    "batch": false
}
```

Embeddings

🔗
get <https://api.openai.com/v1/organization/usage/embeddings>

Get embeddings usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently **1m**, **1h** and **1d** are supported, default to **1d**.

Return only usage for these projects.

Return only usage for these users.

Return only usage for these API keys.

Return only usage for these models.

Group the usage data by the specified fields. Support fields include **project_id**, **user_id**, **api_key_id**, **model** or any combination of them.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Embeddings usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/embeddings?start_time=1730419200&limit=1"
\
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.embeddings.result",
          "input_tokens": 16,
          "num_model_requests": 2,
          "project_id": null,
          "user_id": null,
          "api_key_id": null,
          "model": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}

```

Embeddings usage object

 The aggregated embeddings usage details of the specific time bucket.

 object

string

The aggregated number of input tokens used.

The count of requests made to the model.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

When `group_by=user_id`, this field provides the user ID of the grouped usage result.

When `group_by=api_key_id`, this field provides the API key ID of the grouped usage result.

When `group_by=model`, this field provides the model name of the grouped usage result.

OBJECT Embeddings usage object

```
1
2
3
4
5
6
7
8
9
{
  "object": "organization.usage.embeddings.result",
  "input_tokens": 20,
  "num_model_requests": 2,
  "project_id": "proj_abc",
  "user_id": "user-abc",
  "api_key_id": "key_abc",
  "model": "text-embedding-ada-002-v2"
}
```

Moderations

🔗
get <https://api.openai.com/v1/organization/usage/moderations>

Get moderations usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usage for these projects.

Return only usage for these users.

Return only usage for these API keys.

Return only usage for these models.

Group the usage data by the specified fields. Support fields include `project_id`, `user_id`, `api_key_id`, `model` or any combination of them.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Moderations usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/moderations?start_time=1730419200&limit=1"
 \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.moderations.result",
          "input_tokens": 16,
          "num_model_requests": 2,
          "project_id": null,
          "user_id": null,
          "api_key_id": null,
          "model": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}

```

Moderations usage object

 The aggregated moderations usage details of the specific time bucket.

 object

string

The aggregated number of input tokens used.

The count of requests made to the model.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

When `group_by=user_id`, this field provides the user ID of the grouped usage result.

When `group_by=api_key_id`, this field provides the API key ID of the grouped usage result.

When `group_by=model`, this field provides the model name of the grouped usage result.

OBJECT Moderations usage object

```
1
2
3
4
5
6
7
8
9
{
  "object": "organization.usage.moderations.result",
  "input_tokens": 20,
  "num_model_requests": 2,
  "project_id": "proj_abc",
  "user_id": "user-abc",
  "api_key_id": "key_abc",
  "model": "text-moderation"
}
```

Images

🔗
get <https://api.openai.com/v1/organization/usage/images>

Get images usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usages for these sources. Possible values are `image.generation`, `image.edit`, `image.variation` or any combination of them.

Return only usages for these image sizes. Possible values are `256x256`, `512x512`, `1024x1024`, `1792x1792`, `1024x1792` or any combination of them.

Return only usage for these projects.

Return only usage for these users.

Return only usage for these API keys.

Return only usage for these models.

Group the usage data by the specified fields. Support fields include `project_id`, `user_id`, `api_key_id`, `model`, `size`, `source` or any combination of them.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed [Images usage](#) objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/images?start_time=1730419200&limit=1" \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.images.result",
          "images": 2,
          "num_model_requests": 2,
          "size": null,
          "source": null,
          "project_id": null,
          "user_id": null,
          "api_key_id": null,
          "model": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}
```

Images usage object



The aggregated images usage details of the specific time bucket.

object

string

The number of images processed.

The count of requests made to the model.

When `group_by=source`, this field provides the source of the grouped usage result, possible values are `image.generation`, `image.edit`, `image.variation`.

When `group_by=size`, this field provides the image size of the grouped usage result.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

When `group_by=user_id`, this field provides the user ID of the grouped usage result.

When `group_by=api_key_id`, this field provides the API key ID of the grouped usage result.

When `group_by=model`, this field provides the model name of the grouped usage result.

OBJECT Images usage object

```
1
2
3
4
5
6
7
8
9
10
11
{
  "object": "organization.usage.images.result",
  "images": 2,
  "num_model_requests": 2,
  "size": "1024x1024",
  "source": "image.generation",
  "project_id": "proj_abc",
  "user_id": "user-abc",
  "api_key_id": "key_abc",
  "model": "dall-e-3"
}
```

Audio speeches

get https://api.openai.com/v1/organization/usage/audio_speeches

Get audio speeches usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usage for these projects.

Return only usage for these users.

Return only usage for these API keys.

Return only usage for these models.

Group the usage data by the specified fields. Support fields include `project_id`, `user_id`, `api_key_id`, `model` or any combination of them.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Audio speeches usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/audio_speeches?
start_time=1730419200&limit=1" \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.audio_speeches.result",
          "characters": 45,
          "num_model_requests": 1,
          "project_id": null,
          "user_id": null,
          "api_key_id": null,
          "model": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}

```

Audio speeches usage object



The aggregated audio speeches usage details of the specific time bucket.



object

string

The number of characters processed.

The count of requests made to the model.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

When `group_by=user_id`, this field provides the user ID of the grouped usage result.

When `group_by=api_key_id`, this field provides the API key ID of the grouped usage result.

When `group_by=model`, this field provides the model name of the grouped usage result.

OBJECT Audio speeches usage object

```
1
2
3
4
5
6
7
8
9
{
  "object": "organization.usage.audio_speeches.result",
  "characters": 45,
  "num_model_requests": 1,
  "project_id": "proj_abc",
  "user_id": "user-abc",
  "api_key_id": "key_abc",
  "model": "tts-1"
}
```

Audio transcriptions

🔗
get https://api.openai.com/v1/organization/usage/audio_transcriptions

Get audio transcriptions usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usage for these projects.

Return only usage for these users.

Return only usage for these API keys.

Return only usage for these models.

Group the usage data by the specified fields. Support fields include `project_id`, `user_id`, `api_key_id`, `model` or any combination of them.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Audio transcriptions usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/audio_transcriptions?
start_time=1730419200&limit=1" \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.audio_transcriptions.result",
          "seconds": 20,
          "num_model_requests": 1,
          "project_id": null,
          "user_id": null,
          "api_key_id": null,
          "model": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}

```

Audio transcriptions usage object



The aggregated audio transcriptions usage details of the specific time bucket.



object

string

The number of seconds processed.

The count of requests made to the model.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

When `group_by=user_id`, this field provides the user ID of the grouped usage result.

When `group_by=api_key_id`, this field provides the API key ID of the grouped usage result.

When `group_by=model`, this field provides the model name of the grouped usage result.

OBJECT Audio transcriptions usage object

```
1
2
3
4
5
6
7
8
9
{
  "object": "organization.usage.audio_transcriptions.result",
  "seconds": 10,
  "num_model_requests": 1,
  "project_id": "proj_abc",
  "user_id": "user-abc",
  "api_key_id": "key_abc",
  "model": "tts-1"
}
```

Vector stores

🔗
get https://api.openai.com/v1/organization/usage/vector_stores

Get vector stores usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usage for these projects.

Group the usage data by the specified fields. Support fields include `project_id`.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Vector stores usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/vector_stores?
start_time=1730419200&limit=1" \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.vector_stores.result",
          "usage_bytes": 1024,
          "project_id": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}

```

Vector stores usage object

⌚

The aggregated vector stores usage details of the specific time bucket.

⌚
object

string

The vector stores usage in bytes.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

OBJECT Vector stores usage object

```
1
2
3
4
5
{
  "object": "organization.usage.vector_stores.result",
  "usage_bytes": 1024,
  "project_id": "proj_abc"
}
```

Code interpreter sessions

⌚
get https://api.openai.com/v1/organization/usage/code_interpreter_sessions

Get code interpreter sessions usage details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently `1m`, `1h` and `1d` are supported, default to `1d`.

Return only usage for these projects.

Group the usage data by the specified fields. Support fields include `project_id`.

Specifies the number of buckets to return.

- `bucket_width=1d`: default: 7, max: 31
- `bucket_width=1h`: default: 24, max: 168
- `bucket_width=1m`: default: 60, max: 1440

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed Code interpreter sessions usage objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/usage/code_interpreter_sessions?
start_time=1730419200&limit=1" \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.usage.code_interpreter_sessions.result",
          "sessions": 1,
          "project_id": null
        }
      ]
    }
  ],
  "has_more": false,
  "next_page": null
}

```

Code interpreter sessions usage object

⌚

The aggregated code interpreter sessions usage details of the specific time bucket.

⌚
object

string

The number of code interpreter sessions.

When `group_by=project_id`, this field provides the project ID of the grouped usage result.

OBJECT Code interpreter sessions usage object

```
1
2
3
4
5
{
  "object": "organization.usage.code_interpreter_sessions.result",
  "sessions": 1,
  "project_id": "proj_abc"
}
```

Costs

🔗
get <https://api.openai.com/v1/organization/costs>

Get costs details for the organization.

Query parameters

Start time (Unix seconds) of the query time range, inclusive.

End time (Unix seconds) of the query time range, exclusive.

Width of each time bucket in response. Currently only `1d` is supported, default to `1d`.

Return only costs for these projects.

Group the costs by the specified fields. Support fields include `project_id`, `line_item` and any combination of them.

A limit on the number of buckets to be returned. Limit can range between 1 and 180, and the default is 7.

A cursor for use in pagination. Corresponding to the `next_page` field from the previous response.

Returns

A list of paginated, time bucketed [Costs](#) objects.

Example request

```
1
2
3
curl "https://api.openai.com/v1/organization/costs?start_time=1730419200&limit=1" \
-H "Authorization: Bearer $OPENAI_ADMIN_KEY" \
-H "Content-Type: application/json"
```

Response

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "object": "page",
  "data": [
    {
      "object": "bucket",
      "start_time": 1730419200,
      "end_time": 1730505600,
      "results": [
        {
          "object": "organization.costs.result",
          "amount": {
            "value": 0.06,
            "currency": "usd"
          },
          "line_item": null,
          "project_id": null
        }
      ]
    },
    {
      "has_more": false,
      "next_page": null
    }
  ]
}

```

Costs object

 The aggregated costs details of the specific time bucket.

 object

string

The monetary value in its associated currency.

When `group_by=line_item`, this field provides the line item of the grouped costs result.

When `group_by=project_id`, this field provides the project ID of the grouped costs result.

OBJECT Costs object

```
1
2
3
4
5
6
7
8
9
{
  "object": "organization.costs.result",
  "amount": {
    "value": 0.06,
    "currency": "usd"
  },
  "line_item": "Image models",
  "project_id": "proj_abc"
}
```

Realtime Beta



Communicate with a GPT-4o class model in real time using WebRTC or WebSockets. Supports text and audio inputs and outputs, along with audio transcriptions. [Learn more about the Realtime API](#).

Session tokens



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

Create session



post <https://api.openai.com/v1/realtime/sessions>

Create an ephemeral API token for use in client-side applications with the Realtime API. Can be configured with the same session parameters as the `session.update` client event.

It responds with a session object, plus a `client_secret` key which contains a usable ephemeral API token that can be used to authenticate browser clients for the Realtime API.

Request body

The set of modalities the model can respond with. To disable audio, set this to ["text"].

The Realtime model used for this session.

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.

The voice the model uses to respond. Voice cannot be changed during the session once the model has responded with audio at least once. Current voice options are `alloy`, `ash`, `ballad`, `coral`, `echo sage`, `shimmer` and `verse`.

The format of input audio. Options are `pcm16`, `g711_ulaw`, or `g711_alaw`. For `pcm16`, input audio must be 16-bit PCM at a 24kHz sample rate, single channel (mono), and little-endian byte order.

The format of output audio. Options are `pcm16`, `g711_ulaw`, or `g711_alaw`. For `pcm16`, output audio is sampled at a rate of 24kHz.

Configuration for input audio transcription, defaults to off and can be set to `null` to turn off once on. Input audio transcription is not native to the model, since the model consumes audio directly.

Transcription runs asynchronously through [OpenAI Whisper transcription](#) and should be treated as rough guidance rather than the representation understood by the model. The client can optionally set the language and prompt for transcription, these fields will be passed to the Whisper API.

Configuration for turn detection. Can be set to `null` to turn off. Server VAD means that the model will detect the start and end of speech based on audio volume and respond at the end of user speech.

Tools (functions) available to the model.

How the model chooses tools. Options are `auto`, `none`, `required`, or specify a function.

Sampling temperature for the model, limited to [0.6, 1.2]. Defaults to 0.8.

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`.

Returns

The created Realtime session object, plus an ephemeral key

Example request

```
1
2
3
4
5
6
7
8
curl -X POST https://api.openai.com/v1/realtim sessions \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-d '{
  "model": "gpt-4o-realtime-preview-2024-12-17",
  "modalities": ["audio", "text"],
  "instructions": "You are a friendly assistant."
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
{
  "id": "sess_001",
  "object": "realtime.session",
  "model": "gpt-4o-realtime-preview-2024-12-17",
  "modalities": ["audio", "text"],
  "instructions": "You are a friendly assistant.",
  "voice": "alloy",
  "input_audio_format": "pcm16",
  "output_audio_format": "pcm16",
  "input_audio_transcription": {
    "model": "whisper-1"
  },
  "turn_detection": null,
  "tools": [],
  "tool_choice": "none",
  "temperature": 0.7,
  "max_response_output_tokens": 200,
  "client_secret": {
    "value": "ek_abc123",
    "expires_at": 1234567890
  }
}
```

The session object



A new Realtime session configuration, with an ephemeral key. Default TTL for keys is one minute.

Ephemeral key returned by the API.

The set of modalities the model can respond with. To disable audio, set this to ["text"].

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.

The voice the model uses to respond. Voice cannot be changed during the session once the model has responded with audio at least once. Current voice options are `alloy`, `ash`, `ballad`, `coral`, `echo sage`, `shimmer` and `verse`.

The format of input audio. Options are `pcm16`, `g711_ulaw`, or `g711_alaw`.

The format of output audio. Options are `pcm16`, `g711_ulaw`, or `g711_alaw`.

Configuration for input audio transcription, defaults to off and can be set to `null` to turn off once on. Input audio transcription is not native to the model, since the model consumes audio directly. Transcription runs asynchronously through Whisper and should be treated as rough guidance rather than the representation understood by the model.

Configuration for turn detection. Can be set to `null` to turn off. Server VAD means that the model will detect the start and end of speech based on audio volume and respond at the end of user speech.

Tools (functions) available to the model.

How the model chooses tools. Options are `auto`, `none`, `required`, or specify a function.

Sampling temperature for the model, limited to [0.6, 1.2]. Defaults to 0.8.

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`.

OBJECT The session object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
{
  "id": "sess_001",
  "object": "realtime.session",
  "model": "gpt-4o-realtime-preview-2024-12-17",
  "modalities": ["audio", "text"],
  "instructions": "You are a friendly assistant.",
  "voice": "alloy",
  "input_audio_format": "pcm16",
  "output_audio_format": "pcm16",
  "input_audio_transcription": {
    "model": "whisper-1"
  },
  "turn_detection": null,
  "tools": [],
  "tool_choice": "none",
  "temperature": 0.7,
  "max_response_output_tokens": 200,
  "client_secret": {
    "value": "ek_abc123",
    "expires_at": 1234567890
  }
}
```

Client events



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



session.update

⌚

Send this event to update the session's default configuration. The client may send this event at any time to update the session configuration, and any field may be updated at any time, except for "voice". The server will respond with a `session.updated` event that shows the full effective configuration. Only fields that are present are updated, thus the correct way to clear a field like "instructions" is to pass an empty string.

Optional client-generated ID used to identify this event.

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

Realtime session object configuration.

OBJECT `session.update`

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
{
  "event_id": "event_123",
  "type": "session.update",
  "session": {
    "modalities": ["text", "audio"],
    "instructions": "You are a helpful assistant.",
    "voice": "sage",
    "input_audio_format": "pcm16",
    "output_audio_format": "pcm16",
    "input_audio_transcription": {
      "model": "whisper-1"
    },
    "turn_detection": {
      "type": "server_vad",
      "threshold": 0.5,
      "prefix_padding_ms": 300,
      "silence_duration_ms": 500,
      "create_response": true
    },
  }
}
```

```

    "tools": [
        {
            "type": "function",
            "name": "get_weather",
            "description": "Get the current weather...",
            "parameters": {
                "type": "object",
                "properties": {
                    "location": { "type": "string" }
                },
                "required": ["location"]
            }
        }
    ],
    "tool_choice": "auto",
    "temperature": 0.8,
    "max_response_output_tokens": "inf"
}
}

```

⌚

input_audio_buffer.append

⌚

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

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

Optional client-generated ID used to identify this event.

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

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

OBJECT input_audio_buffer.append

```

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

```

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.

Optional client-generated ID used to identify this event.

The event type, must be [input_audio_buffer.commit](#).

OBJECT input_audio_buffer.commit

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

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.

Optional client-generated ID used to identify this event.

The event type, must be [input_audio_buffer.clear](#).

OBJECT input_audio_buffer.clear

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



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.

Optional client-generated ID used to identify this event.

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

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.

The item to add to the conversation.

OBJECT `conversation.item.create`

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
{
  "event_id": "event_345",
  "type": "conversation.item.create",
  "previous_item_id": null,
  "item": {
    "id": "msg_001",
    "type": "message",
    "role": "user",
    "content": [
      {
        "type": "input_text",
        "text": "Hello, how are you?"
      }
    ]
  }
}
```

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 not text in the context that hasn't been heard by the user.

If successful, the server will respond with a `conversation.item.truncated` event.

Optional client-generated ID used to identify this event.

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

The ID of the assistant message item to truncate. Only assistant message items can be truncated.

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

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.

OBJECT conversation.item.truncate

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

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.

Optional client-generated ID used to identify this event.

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

The ID of the item to delete.

OBJECT conversation.item.delete

```
1
2
3
4
5
6
{
  "event_id": "event_901",
  "type": "conversation.item.delete",
  "item_id": "msg_003"
}
```



response.create



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

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

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

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

Optional client-generated ID used to identify this event.

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

Create a new Realtime response with these parameters

OBJECT `response.create`

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
{
  "event_id": "event_234",
  "type": "response.create",
  "response": {
    "modalities": ["text", "audio"],
    "instructions": "Please assist the user.",
    "voice": "sage",
    "output_audio_format": "pcm16",
    "tools": [
      {
        "type": "function",
        "name": "calculate_sum",
        "description": "Calculates the sum of two numbers.",
        "parameters": {
          "type": "object",
          "properties": {
            "a": { "type": "number" },
            "b": { "type": "number" }
          },
          "required": ["a", "b"]
        }
      }
    ],
    "tool_choice": "auto",
    "temperature": 0.8,
    "max_output_tokens": 1024
  }
}
```

response.cancel



Send this event to cancel an in-progress response. The server will respond with a `response.cancelled` event or an error if there is no response to cancel.

Optional client-generated ID used to identify this event.

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

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

OBJECT response.cancel

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

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.

The unique ID of the server event.

The event type, must be `error`.

Details of the error.

OBJECT error

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



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.

The unique ID of the server event.

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

Realtime session object configuration.

OBJECT `session.created`

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
{
  "event_id": "event_1234",
  "type": "session.created",
  "session": {
    "id": "sess_001",
    "object": "realtime.session",
    "model": "gpt-4o-realtime-preview-2024-12-17",
    "modalities": ["text", "audio"],
    "instructions": "...model instructions here...",
    "voice": "sage",
    "input_audio_format": "pcm16",
    "output_audio_format": "pcm16",
    "input_audio_transcription": null,
    "turn_detection": {
      "type": "server_vad",
      "threshold": 0.5,
      "prefix_padding_ms": 300,
      "silence_duration_ms": 200
    },
    "tools": [],
    "tool_choice": "auto",
    "temperature": 0.8,
    "max_response_output_tokens": "inf"
  }
}
```

session.updated



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

The unique ID of the server event.

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

Realtime session object configuration.

OBJECT `session.updated`

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
{
  "event_id": "event_5678",
  "type": "session.updated",
  "session": {
    "id": "sess_001",
    "object": "realtime.session",
    "model": "gpt-4o-realtime-preview-2024-12-17",
    "modalities": ["text"],
    "instructions": "New instructions",
    "voice": "sage",
    "input_audio_format": "pcm16",
    "output_audio_format": "pcm16",
    "input_audio_transcription": {
      "model": "whisper-1"
    },
    "turn_detection": null,
    "tools": [],
    "tool_choice": "none",
    "temperature": 0.7,
    "max_response_output_tokens": 200
  }
}
```



conversation.created



Returned when a conversation is created. Emitted right after session creation.

The unique ID of the server event.

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

The conversation resource.

OBJECT `conversation.created`

```
1
2
3
4
5
6
7
8
{
  "event_id": "event_9101",
  "type": "conversation.created",
  "conversation": {
    "id": "conv_001",
    "object": "realtime.conversation"
  }
}
```



conversation.item.created



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

- The server is generating a Response, which if successful will produce either one or two Items, which will be of type `message` (`role assistant`) or type `function_call`.
- The input audio buffer has been committed, either by the client or the server (in `server_vad` mode). The server will take the content of the input audio buffer and add it to a new user message Item.
- The client has sent a `conversation.item.create` event to add a new Item to the Conversation.

The unique ID of the server event.

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

The ID of the preceding item in the Conversation context, allows the client to understand the order of the conversation.

The item to add to the conversation.

OBJECT conversation.item.created

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "event_id": "event_1920",
  "type": "conversation.item.created",
  "previous_item_id": "msg_002",
  "item": {
    "id": "msg_003",
    "object": "realtime.item",
    "type": "message",
    "status": "completed",
    "role": "user",
    "content": [
      {
        "type": "input_audio",
        "transcript": "hello how are you",
        "audio": "base64encodedaudio=="
      }
    ]
  }
}
```



conversation.item.input_audio_transcription.completed



This event is the output of audio transcription for user audio written to the user audio buffer.

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

Realtime API models accept audio natively, and thus input transcription is a separate process run on a separate ASR (Automatic Speech Recognition) model, currently always `whisper-1`. Thus the transcript may diverge somewhat from the model's interpretation, and should be treated as a rough guide.

The unique ID of the server event.

The event type, must be `conversation.item.input_audio_transcription.completed`.

The ID of the user message item containing the audio.

The index of the content part containing the audio.

The transcribed text.

OBJECT `conversation.item.input_audio_transcription.completed`

```
1
2
3
4
5
6
7
{
  "event_id": "event_2122",
  "type": "conversation.item.input_audio_transcription.completed",
  "item_id": "msg_003",
  "content_index": 0,
  "transcript": "Hello, how are you?"
}
```

`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.

The unique ID of the server event.

The event type, must be `conversation.item.input_audio_transcription.failed`.

The ID of the user message item.

The index of the content part containing the audio.

Details of the transcription error.

OBJECT `conversation.item.input_audio_transcription.failed`

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

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.

The unique ID of the server event.

The event type, must be [conversation.item.truncated](#).

The ID of the assistant message item that was truncated.

The index of the content part that was truncated.

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

OBJECT [conversation.item.truncated](#)

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

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.

The unique ID of the server event.

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

The ID of the item that was deleted.

OBJECT conversation.item.deleted

```
1
2
3
4
5
6
{
  "event_id": "event_2728",
  "type": "conversation.item.deleted",
  "item_id": "msg_005"
}
```



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.

The unique ID of the server event.

The event type, must be [input_audio_buffer.committed](#).

The ID of the preceding item after which the new item will be inserted.

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

OBJECT [input_audio_buffer.committed](#)

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

[input_audio_buffer.cleared](#)



Returned when the input audio buffer is cleared by the client with a [input_audio_buffer.clear](#) event.

The unique ID of the server event.

The event type, must be [input_audio_buffer.cleared](#).

OBJECT [input_audio_buffer.cleared](#)

```
1
2
3
4
5
{
  "event_id": "event_1314",
  "type": "input_audio_buffer.cleared"
}
```

[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).

The unique ID of the server event.

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

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.

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

OBJECT `input_audio_buffer.speech_started`

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

`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.

The unique ID of the server event.

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

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.

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

OBJECT `input_audio_buffer.speech_stopped`

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

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`.

The unique ID of the server event.

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

The response resource.

OBJECT response.created

2
3
4
5
6
7
8
9
10
11
12
{
}

`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.

The unique ID of the server event.

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

The response resource.

OBJECT `response.done`

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
{
  "event_id": "event_3132",
  "type": "response.done",
  "response": {
    "id": "resp_001",
    "object": "realtime.response",
    "status": "completed",
    "status_details": null,
    "output": [
      {
        "id": "msg_006",
        "object": "realtime.item",
        "type": "message",
        "status": "completed",
```

```
        "role": "assistant",
        "content": [
            {
                "type": "text",
                "text": "Sure, how can I assist you today?"
            }
        ],
        "usage": {
            "total_tokens":275,
            "input_tokens":127,
            "output_tokens":148,
            "input_token_details": {
                "cached_tokens":384,
                "text_tokens":119,
                "audio_tokens":8,
                "cached_tokens_details": {
                    "text_tokens": 128,
                    "audio_tokens": 256
                }
            },
            "output_token_details": {
                "text_tokens":36,
                "audio_tokens":112
            }
        }
    }
}
```

⌚

`response.output_item.added`

⌚

Returned when a new Item is created during Response generation.

The unique ID of the server event.

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

The ID of the Response to which the item belongs.

The index of the output item in the Response.

The item to add to the conversation.

OBJECT `response.output_item.added`

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

`response.output_item.done`



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

The unique ID of the server event.

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

The ID of the Response to which the item belongs.

The index of the output item in the Response.

The item to add to the conversation.

OBJECT `response.output_item.done`

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

⌚

`response.content_part.added`

⌚

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

The unique ID of the server event.

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

The ID of the response.

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

The index of the output item in the response.

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

The content part that was added.

OBJECT `response.content_part.added`

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

`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.

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

The content part that is done.

OBJECT response.content_part.done

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



response.text.delta



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

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

The text delta.

OBJECT response.text.delta

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

response.text.done



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

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

The final text content.

OBJECT `response.text.done`

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



response.audio_transcript.delta



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

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

The transcript delta.

OBJECT `response.audio_transcript.delta`

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

response.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.

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

The final transcript of the audio.

OBJECT `response.audio_transcript.done`

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



response.audio.delta



Returned when the model-generated audio is updated.

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

Base64-encoded audio data delta.

OBJECT `response.audio.delta`

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

response.audio.done



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

The unique ID of the server event.

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

The ID of the response.

The ID of the item.

The index of the output item in the response.

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

OBJECT `response.audio.done`

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

⌚

`response.function_call_arguments.delta`

⌚

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

The unique ID of the server event.

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

The ID of the response.

The ID of the function call item.

The index of the output item in the response.

The ID of the function call.

The arguments delta as a JSON string.

OBJECT `response.function_call_arguments.delta`

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

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.

The unique ID of the server event.

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

The ID of the response.

The ID of the function call item.

The index of the output item in the response.

The ID of the function call.

The final arguments as a JSON string.

OBJECT `response.function_call_arguments.done`

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



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.

The unique ID of the server event.

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

List of rate limit information.

OBJECT `rate_limits.updated`

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

Completions

Legacy



Given a prompt, the model will return one or more predicted completions along with the probabilities of alternative tokens at each position. Most developer should use our [Chat Completions API](#) to leverage our best and newest models.

Create completion

Legacy



post <https://api.openai.com/v1/completions>

Creates a completion for the provided prompt and parameters.

Request body

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.

The prompt(s) to generate completions for, encoded as a string, array of strings, array of tokens, or array of token arrays.

Note that <|endoftext|> is the document separator that the model sees during training, so if a prompt is not specified the model will generate as if from the beginning of a new document.

Generates `best_of` completions server-side and returns the "best" (the one with the highest log probability per token). Results cannot be streamed.

When used with `n`, `best_of` controls the number of candidate completions and `n` specifies how many to return – `best_of` must be greater than `n`.

Note: Because this parameter generates many completions, it can quickly consume your token quota. Use carefully and ensure that you have reasonable settings for `max_tokens` and `stop`.

Echo back the prompt in addition to the completion

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.

[See more information about frequency and presence penalties.](#)

Modify the likelihood of specified tokens appearing in the completion.

Accepts a JSON object that maps tokens (specified by their token ID in the GPT tokenizer) to an associated bias value from -100 to 100. You can use this [tokenizer tool](#) to convert text to token IDs. 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.

As an example, you can pass `{"50256": -100}` to prevent the <|endoftext|> token from being generated.

Include the log probabilities on the `logprobs` most likely output tokens, as well the chosen tokens. For example, if `logprobs` is 5, the API will return a list of the 5 most likely tokens. The API will always return the `logprob` of the sampled token, so there may be up to `logprobs+1` elements in the response.

The maximum value for `logprobs` is 5.

The maximum number of `tokens` that can be generated in the completion.

The token count of your prompt plus `max_tokens` cannot exceed the model's context length. [Example Python code](#) for counting tokens.

How many completions to generate for each prompt.

Note: Because this parameter generates many completions, it can quickly consume your token quota. Use carefully and ensure that you have reasonable settings for `max_tokens` and `stop`.

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.

[See more information about frequency and presence penalties.](#)

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.

Up to 4 sequences where the API will stop generating further tokens. The returned text will not contain the stop sequence.

Whether to stream back partial progress. If set, tokens will be sent as data-only [server-sent events](#) as they become available, with the stream terminated by a `data: [DONE]` message. [Example Python code](#).

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

The suffix that comes after a completion of inserted text.

This parameter is only supported for `gpt-3.5-turbo-instruct`.

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.

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.

A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse. [Learn more](#).

Returns

Returns a [completion](#) object, or a sequence of completion objects if the request is streamed.

Example request

```
1
2
3
4
5
6
7
8
9
curl https://api.openai.com/v1/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-d '{
  "model": "gpt-3.5-turbo-instruct",
  "prompt": "Say this is a test",
  "max_tokens": 7,
  "temperature": 0
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "id": "cmpl-uqkv1QyYK7bGYrRHQ0eXlWi7",
  "object": "text_completion",
  "created": 1589478378,
  "model": "gpt-3.5-turbo-instruct",
  "system_fingerprint": "fp_44709d6fcb",
  "choices": [
    {
      "text": "\n\nThis is indeed a test",
      "index": 0,
      "logprobs": null,
      "finish_reason": "length"
    }
  ],
  "usage": {
    "prompt_tokens": 5,
    "completion_tokens": 7,
    "total_tokens": 12
  }
}
```

The completion object

Legacy



Represents a completion response from the API. Note: both the streamed and non-streamed response objects share the same shape (unlike the chat endpoint).

A unique identifier for the completion.

The list of completion choices the model generated for the input prompt.

The Unix timestamp (in seconds) of when the completion was created.

The model used for completion.

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.

The object type, which is always "text_completion"

Usage statistics for the completion request.

OBJECT The completion object

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "id": "cmpl-uqkv1QyYK7bGYrRHQ0eXlWi7",
  "object": "text_completion",
  "created": 1589478378,
  "model": "gpt-4-turbo",
  "choices": [
    {
      "text": "\n\nThis is indeed a test",
      "index": 0,
      "logprobs": null,
      "finish_reason": "length"
    }
  ],
  "usage": {
    "prompt_tokens": 5,
    "completion_tokens": 7,
    "total_tokens": 12
  }
}
```

Assistants (v1)

Legacy



Build assistants that can call models and use tools to perform tasks.

[Get started with the Assistants API](#)

Create assistant (v1)

Legacy



post <https://api.openai.com/v1/assistants>

Create an assistant with a model and instructions.

Request body

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. type: string

The name of the assistant. The maximum length is 256 characters.

The description of the assistant. The maximum length is 512 characters.

The system instructions that the assistant uses. The maximum length is 256,000 characters.

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`, `retrieval`, or `function`.

A list of [file](#) IDs attached to this assistant. There can be a maximum of 20 files attached to the assistant. Files are ordered by their creation date in ascending order.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

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.

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.

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_object" }` enables JSON mode, which guarantees 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.

Returns

An assistant object.

Example request

```
1
2
3
4
5
6
7
8
9
10
curl "https://api.openai.com/v1/assistants" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
  "instructions": "You are a personal math tutor. When asked a question, write and run
Python code to answer the question.",
  "name": "Math Tutor",
  "tools": [{"type": "code_interpreter"}],
  "model": "gpt-4-turbo"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "id": "asst_abc123",
  "object": "assistant",
  "created_at": 1698984975,
  "name": "Math Tutor",
  "description": null,
  "model": "gpt-4-turbo",
  "instructions": "You are a personal math tutor. When asked a question, write and run Python code to answer the question.",
  "tools": [
    {
      "type": "code_interpreter"
    }
  ],
  "file_ids": [],
  "metadata": {},
  "top_p": 1.0,
  "temperature": 1.0,
  "response_format": "auto"
}
```

Create assistant file (v1) Legacy



post https://api.openai.com/v1/assistants/{assistant_id}/files

Create an assistant file by attaching a File to an assistant.

Path parameters

The ID of the assistant for which to create a File.

Request body

A File ID (with `purpose="assistants"`) that the assistant should use. Useful for tools like `retrieval` and `code_interpreter` that can access files.

Returns

An assistant file object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/assistants/asst_abc123/files \
      -H 'Authorization: Bearer $OPENAI_API_KEY' \
      -H 'Content-Type: application/json' \
      -H 'OpenAI-Beta: assistants=v1' \
      -d '{
            "file_id": "file-abc123"
        }'
```

Response

```
1
2
3
4
5
6
{
  "id": "file-abc123",
  "object": "assistant.file",
  "created_at": 1699055364,
  "assistant_id": "asst_abc123"
}
```

List assistants (v1) Legacy

2

get https://api.openai.com/v1/assistants

Returns a list of assistants.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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, ending with `obj_foo`, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Returns

A list of assistant objects.

Example request

```
1
2
3
4
curl "https://api.openai.com/v1/assistants?order=desc&limit=20" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
{
  "object": "list",
  "data": [
    {
      "label": "Section 1"
    },
    {
      "label": "Section 2"
    }
  ]
}
```

```

    "id": "asst_abc123",
    "object": "assistant",
    "created_at": 1698982736,
    "name": "Coding Tutor",
    "description": null,
    "model": "gpt-4-turbo",
    "instructions": "You are a helpful assistant designed to make me better at coding!",
    "tools": [],
    "file_ids": [],
    "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-4-turbo",
    "instructions": "You are a helpful assistant designed to make me better at coding!",
    "tools": [],
    "file_ids": [],
    "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-4-turbo",
    "instructions": null,
    "tools": [],
    "file_ids": [],
    "metadata": {},
    "top_p": 1.0,
    "temperature": 1.0,
    "response_format": "auto"
}
],
"first_id": "asst_abc123",
"last_id": "asst_abc789",
"has_more": false
}

```

List assistant files (v1)

Legacy



get https://api.openai.com/v1/assistants/{assistant_id}/files

Returns a list of assistant files.

Path parameters

The ID of the assistant the file belongs to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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, ending with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Returns

A list of [assistant file](#) objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/assistants/asst_abc123/files \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "object": "list",
  "data": [
    {
      "id": "file-abc123",
      "object": "assistant.file",
      "created_at": 1699060412,
      "assistant_id": "asst_abc123"
    },
    {
      "id": "file-abc456",
      "object": "assistant.file",
      "created_at": 1699060412,
      "assistant_id": "asst_abc123"
    }
  ],
  "first_id": "file-abc123",
  "last_id": "file-abc456",
  "has_more": false
}
```

Retrieve assistant (v1)

Legacy



get https://api.openai.com/v1/assistants/{assistant_id}

Retrieves an assistant.

Path parameters

The ID of the assistant to retrieve.

Returns

The assistant object matching the specified ID.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/assistants/asst_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
{
  "id": "asst_abc123",
  "object": "assistant",
  "created_at": 1699009709,
  "name": "HR Helper",
  "description": null,
  "model": "gpt-4-turbo",
  "instructions": "You are an HR bot, and you have access to files to answer employee questions about company policies.",
  "tools": [
    {
      "type": "retrieval"
    }
  ],
  "file_ids": [
    "file-abc123"
  ],
  "metadata": {}
}
```

Retrieve assistant file (v1)

Legacy



get https://api.openai.com/v1/assistants/{assistant_id}/files/{file_id}

Retrieves an AssistantFile.

Path parameters

The ID of the assistant who the file belongs to.

The ID of the file we're getting.

Returns

The assistant_file object matching the specified ID.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/assistants/asst_abc123/files/file-abc123 \
-H 'Authorization: Bearer $OPENAI_API_KEY' \
-H 'Content-Type: application/json' \
-H 'OpenAI-Beta: assistants=v1'
```

Response

```
1
2
3
4
5
6
{
  "id": "file-abc123",
  "object": "assistant.file",
  "created_at": 1699055364,
  "assistant_id": "asst_abc123"
}
```

Modify assistant (v1)

Legacy



post https://api.openai.com/v1/assistants/{assistant_id}

Modifies an assistant.

Path parameters

The ID of the assistant to modify.

Request body

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. type: string

The name of the assistant. The maximum length is 256 characters.

The description of the assistant. The maximum length is 512 characters.

The system instructions that the assistant uses. The maximum length is 256,000 characters.

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`, `retrieval`, or `function`.

A list of [File](#) IDs attached to this assistant. There can be a maximum of 20 files attached to the assistant. Files are ordered by their creation date in ascending order. If a file was previously attached to the list but does not show up in the list, it will be deleted from the assistant.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

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.

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.

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_object" }` enables JSON mode, which guarantees 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.

Returns

The modified [assistant](#) object.

Example request

```
1
2
3
4
5
6
7
8
9
10
curl https://api.openai.com/v1/assistants/asst_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
    "instructions": "You are an HR bot, and you have access to files to answer employee
questions about company policies. Always response with info from either of the files.",
    "tools": [{"type": "retrieval"}],
    "model": "gpt-4-turbo",
    "file_ids": ["file-abc123", "file-abc456"]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
{
  "id": "asst_abc123",
  "object": "assistant",
  "created_at": 1699009709,
  "name": "HR Helper",
  "description": null,
  "model": "gpt-4-turbo",
  "instructions": "You are an HR bot, and you have access to files to answer employee questions about company policies. Always response with info from either of the files.",
  "tools": [
    {
      "type": "retrieval"
    }
  ],
  "file_ids": [
    "file-abc123",
    "file-abc456"
  ],
  "metadata": {},
  "top_p": 1.0,
  "temperature": 1.0,
  "response_format": "auto"
}
```

Delete assistant (v1) Legacy



delete https://api.openai.com/v1/assistants/{assistant_id}

Delete an assistant.

Path parameters

The ID of the assistant to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/assistants/asst_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  "id": "asst_abc123",
  "object": "assistant.deleted",
  "deleted": true
}
```

Delete assistant file (v1)

Legacy



delete https://api.openai.com/v1/assistants/{assistant_id}/files/{file_id}

Delete an assistant file.

Path parameters

The ID of the assistant that the file belongs to.

The ID of the file to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/assistants/asst_abc123/files/file-abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  id: "file-abc123",
  object: "assistant.file.deleted",
  deleted: true
}
```

The assistant object (v1)

Legacy



Represents an [assistant](#) that can call the model and use tools.

The identifier, which can be referenced in API endpoints.

The object type, which is always [assistant](#).

The Unix timestamp (in seconds) for when the assistant was created.

The name of the assistant. The maximum length is 256 characters.

The description of the assistant. The maximum length is 512 characters.

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. type: string

The system instructions that the assistant uses. The maximum length is 256,000 characters.

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](#), [retrieval](#), or [function](#).

A list of [file](#) IDs attached to this assistant. There can be a maximum of 20 files attached to the assistant. Files are ordered by their creation date in ascending order.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

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.

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.

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_object" }` enables JSON mode, which guarantees 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.

OBJECT The assistant object (v1)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
{
  "id": "asst_abc123",
  "object": "assistant",
  "created_at": 1698984975,
  "name": "Math Tutor",
  "description": null,
  "model": "gpt-4-turbo",
  "instructions": "You are a personal math tutor. When asked a question, write and run Python code to answer the question.",
  "tools": [
    {
      "type": "code_interpreter"
    }
  ],
  "file_ids": [],
  "metadata": {},
  "top_p": 1.0,
  "temperature": 1.0,
  "response_format": "auto"
}
```

The assistant file object (v1) Legacy



A list of [Files](#) attached to an [assistant](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always [assistant.file](#).

The Unix timestamp (in seconds) for when the assistant file was created.

The assistant ID that the file is attached to.

OBJECT The assistant file object (v1)

```
1
2
3
4
5
6
{
  "id": "file-abc123",
  "object": "assistant.file",
  "created_at": 1699055364,
  "assistant_id": "asst_abc123"
}
```

Threads (v1) Legacy



Create threads that assistants can interact with.

Related guide: [Assistants](#)

Create thread (v1) Legacy



post <https://api.openai.com/v1/threads>

Create a thread.

Request body

A list of [messages](#) to start the thread with.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

Returns

A [thread](#) object.

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/threads \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-d ''
```

Response

```
1
2
3
4
5
6
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1699012949,
  "metadata": {}
}
```

Retrieve thread (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}

Retrieves a thread.

Path parameters

The ID of the thread to retrieve.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1699014083,
  "metadata": {}
}
```

Modify thread (v1)

Legacy



post https://api.openai.com/v1/threads/{thread_id}

Modifies a thread.

Path parameters

The ID of the thread to modify. Only the `metadata` can be modified.

Request body

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

Returns

The modified `thread` object matching the specified ID.

Example request

```
1
2
3
4
5
6
7
8
9
10
curl https://api.openai.com/v1/threads/thread_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
    "metadata": {
        "modified": "true",
        "user": "abc123"
    }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1699014083,
  "metadata": {
    "modified": "true",
    "user": "abc123"
  }
}
```

Delete thread (v1) Legacy



delete https://api.openai.com/v1/threads/{thread_id}

Delete a thread.

Path parameters

The ID of the thread to delete.

Returns

Deletion status

Example request

```
1
2
3
4
5
curl https://api.openai.com/v1/threads/thread_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-X DELETE
```

Response

```
1
2
3
4
5
{
  "id": "thread_abc123",
  "object": "thread.deleted",
  "deleted": true
}
```

The thread object (v1)

Legacy



Represents a thread that contains [messages](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always [thread](#).

The Unix timestamp (in seconds) for when the thread was created.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

OBJECT The thread object (v1)

```
1
2
3
4
5
6
{
  "id": "thread_abc123",
  "object": "thread",
  "created_at": 1698107661,
  "metadata": {}
}
```

Messages (v1)

Legacy



Create messages within threads

Related guide: [Assistants](#)

Create message (v1)

Legacy



post https://api.openai.com/v1/threads/{thread_id}/messages

Create a message.

Path parameters

The ID of the thread to create a message for.

Request body

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.

The content of the message.

A list of [File](#) IDs that the message should use. There can be a maximum of 10 files attached to a message. Useful for tools like [retrieval](#) and [code_interpreter](#) that can access and use files.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

Returns

A message object.

Example request

```
1
2
3
4
5
6
7
8
curl https://api.openai.com/v1/threads/thread_abc123/messages \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
    "role": "user",
    "content": "How does AI work? Explain it in simple terms."
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "id": "msg_abc123",
  "object": "thread.message",
  "created_at": 1699017614,
  "thread_id": "thread_abc123",
  "role": "user",
  "content": [
    {
      "type": "text",
      "text": {
        "value": "How does AI work? Explain it in simple terms.",
        "annotations": []
      }
    }
  ],
  "file_ids": [],
  "assistant_id": null,
  "run_id": null,
  "metadata": {}
}
```

List messages (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}/messages

Returns a list of messages for a given thread.

Path parameters

The ID of the thread the messages belong to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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, ending with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Filter messages by the run ID that generated them.

Returns

A list of `message` objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/messages \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
{
  "object": "list",
  "data": [
    {
      "id": "msg_abc123",
      "object": "thread.message",
      "created_at": 1699016383,
```

```
"thread_id": "thread_abc123",
"role": "user",
"content": [
  {
    "type": "text",
    "text": {
      "value": "How does AI work? Explain it in simple terms.",
      "annotations": []
    }
  }
],
"file_ids": [],
"assistant_id": null,
"run_id": null,
"metadata": {}
},
{
  "id": "msg_abc456",
  "object": "thread.message",
  "created_at": 1699016383,
  "thread_id": "thread_abc123",
  "role": "user",
  "content": [
    {
      "type": "text",
      "text": {
        "value": "Hello, what is AI?",
        "annotations": []
      }
    }
  ],
  "file_ids": [
    "file-abc123"
  ],
  "assistant_id": null,
  "run_id": null,
  "metadata": {}
}
],
"first_id": "msg_abc123",
"last_id": "msg_abc456",
"has_more": false
}
```

List message files (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}/files

Returns a list of message files.

Path parameters

The ID of the thread that the message and files belong to.

The ID of the message that the files belongs to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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, ending with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Returns

A list of message file objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123/files \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "object": "list",
  "data": [
    {
      "id": "file-abc123",
      "object": "thread.message.file",
      "created_at": 1699061776,
      "message_id": "msg_abc123"
    },
    {
      "id": "file-abc123",
      "object": "thread.message.file",
      "created_at": 1699061776,
      "message_id": "msg_abc123"
    }
  ],
  "first_id": "file-abc123",
  "last_id": "file-abc123",
  "has_more": false
}
```

Retrieve message (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}

Retrieve a message.

Path parameters

The ID of the thread to which this message belongs.

The ID of the message to retrieve.

Returns

The message object matching the specified ID.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "id": "msg_abc123",
  "object": "thread.message",
  "created_at": 1699017614,
  "thread_id": "thread_abc123",
  "role": "user",
  "content": [
    {
      "type": "text",
      "text": {
        "value": "How does AI work? Explain it in simple terms.",
        "annotations": []
      }
    }
  ],
  "file_ids": [],
  "assistant_id": null,
  "run_id": null,
  "metadata": {}
}
```

Retrieve message file (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}/files/{file_id}

Retrieves a message file.

Path parameters

The ID of the thread to which the message and File belong.

The ID of the message the file belongs to.

The ID of the file being retrieved.

Returns

The message_file object.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123/files/file-abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
{
  "id": "file-abc123",
  "object": "thread.message.file",
  "created_at": 1699061776,
  "message_id": "msg_abc123"
}
```

Modify message (v1)

Legacy



post https://api.openai.com/v1/threads/{thread_id}/messages/{message_id}

Modifies a message.

Path parameters

The ID of the thread to which this message belongs.

The ID of the message to modify.

Request body

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

Returns

The modified message object.

Example request

```
1
2
3
4
5
6
7
8
9
10
curl https://api.openai.com/v1/threads/thread_abc123/messages/msg_abc123 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
  "metadata": {
    "modified": "true",
    "user": "abc123"
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
{
  "id": "msg_abc123",
  "object": "thread.message",
  "created_at": 1699017614,
  "thread_id": "thread_abc123",
  "role": "user",
  "content": [
    {
      "type": "text",
      "text": {
        "value": "How does AI work? Explain it in simple terms.",
        "annotations": []
      }
    }
  ],
  "file_ids": [],
  "assistant_id": null,
  "run_id": null,
  "metadata": {
    "modified": "true",
    "user": "abc123"
  }
}
```

The message object (v1)

Legacy



Represents a message within a [thread](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always `thread.message`.

The Unix timestamp (in seconds) for when the message was created.

The thread ID that this message belongs to.

The status of the message, which can be either `in_progress`, `incomplete`, or `completed`.

On an incomplete message, details about why the message is incomplete.

The Unix timestamp (in seconds) for when the message was completed.

The Unix timestamp (in seconds) for when the message was marked as incomplete.

The entity that produced the message. One of `user` or `assistant`.

The content of the message in array of text and/or images.

If applicable, the ID of the assistant that authored this message.

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.

A list of file IDs that the assistant should use. Useful for tools like retrieval and code_interpreter that can access files. A maximum of 10 files can be attached to a message.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

OBJECT The message object (v1)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
{
  "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": []
      }
    }
  ],
  "file_ids": [],
  "assistant_id": "asst_abc123",
  "run_id": "run_abc123",
  "metadata": {}
}
```

The message file object (v1)

Legacy



A list of files attached to a [message](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always [thread.message.file](#).

The Unix timestamp (in seconds) for when the message file was created.

The ID of the [message](#) that the [File](#) is attached to.

OBJECT The message file object (v1)

```
1
2
3
4
5
6
7
{
  "id": "file-abc123",
  "object": "thread.message.file",
  "created_at": 1698107661,
  "message_id": "message_QLoItBbqwyAJEz1Ty4y9kOMM",
  "file_id": "file-abc123"
}
```

Runs (v1)

Legacy



Represents an execution run on a thread.

Related guide: [Assistants](#)

Create run (v1)

Legacy



post https://api.openai.com/v1/threads/{thread_id}/runs

Create a run.

Path parameters

The ID of the thread to run.

Request body

The ID of the [assistant](#) to use to execute this run.

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.

Overrides the [instructions](#) of the assistant. This is useful for modifying the behavior on a per-run basis.

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.

Adds additional messages to the thread before creating the run.

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

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.

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.

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.

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 `complete`. See `incomplete_details` for more info.

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 `complete`. See `incomplete_details` for more info.

⌚
truncation_strategy

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 a tool. Specifying a particular tool like `{"type": "TOOL_TYPE"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

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_object" }` enables JSON mode, which guarantees 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.

Returns

A `run` object.

Example request

```
1
2
3
4
5
6
7
curl https://api.openai.com/v1/threads/thread_abc123/runs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
  "assistant_id": "asst_abc123"
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
{
  "id": "run_abc123",
  "object": "thread.run",
  "created_at": 1699063290,
  "assistant_id": "asst_abc123",
  "thread_id": "thread_abc123",
  "status": "queued",
  "started_at": 1699063290,
  "expires_at": null,
  "cancelled_at": null,
  "failed_at": null,
  "completed_at": 1699063291,
  "last_error": null,
  "model": "gpt-4-turbo",
  "instructions": null,
  "incomplete_details": null,
  "tools": [
    {
      "type": "code_interpreter"
```

```
        },
    ],
    "file_ids": [
        "file-abc123",
        "file-abc456"
    ],
    "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"
}
```

Create thread and run (v1)

Legacy



post <https://api.openai.com/v1/threads/runs>

Create a thread and run it in one request.

Request body

The ID of the assistant to use to execute this run.



thread

object

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.

Override the default system message of the assistant. This is useful for modifying the behavior on a per-run basis.

Override the tools the assistant can use for this run. This is useful for modifying the behavior on a per-run basis.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

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.

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.

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.

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 `complete`. See `incomplete_details` for more info.

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 `complete`. See `incomplete_details` for more info.

⌚
truncation_strategy

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 a tool. Specifying a particular tool like `{"type": "TOOL_TYPE"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

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_object" }` enables JSON mode, which guarantees 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.

Returns

A [run](#) object.

Example request

```
1
2
3
4
5
6
7
8
9
10
11
12
curl https://api.openai.com/v1/threads/runs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
  "assistant_id": "asst_abc123",
  "thread": {
    "messages": [
      {"role": "user", "content": "Explain deep learning to a 5 year old."}
    ]
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
{
  "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,
  "last_error": null,
  "model": "gpt-4-turbo",
  "instructions": "You are a helpful assistant.",
  "tools": [],
  "file_ids": [],
  "metadata": {},
  "usage": null,
  "temperature": 1
}
```

List runs (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}/runs

Returns a list of runs belonging to a thread.

Path parameters

The ID of the thread the run belongs to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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, ending with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Returns

A list of `run` objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1"
```

Response

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

```
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
{
  "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-4-turbo",
      "instructions": null,
      "incomplete_details": null,
      "tools": [
        {
          "type": "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"
},
{
    "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-4-turbo",
    "instructions": null,
    "incomplete_details": null,
    "tools": [
        {
            "type": "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"
    },
],
"first_id": "run_abc123",
"last_id": "run_abc456",
"has_more": false
}
```

List run steps (v1)

Legacy



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

The ID of the thread the run and run steps belong to.

The ID of the run the run steps belong to.

Query parameters

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

Sort order by the `created_at` timestamp of the objects. `asc` for ascending order and `desc` for descending order.

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.

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, ending with obj_foo, your subsequent call can include `before=obj_foo` in order to fetch the previous page of the list.

Returns

A list of `run_step` objects.

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123/steps \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
{
  "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 (v1)

Legacy



get https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}

Retrieves a run.

Path parameters

The ID of the thread that was run.

The ID of the run to retrieve.

Returns

The run object matching the specified ID.

Example request

```
1
2
3
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
{
  "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-4-turbo",
  "instructions": null,
```

```
"incomplete_details": null,  
"tools": [  
  {  
    "type": "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"  
}  
}
```

Retrieve run step (v1) Legacy



get https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/steps/{step_id}

Retrieves a run step.

Path parameters

The ID of the thread to which the run and run step belongs.

The ID of the run to which the run step belongs.

The ID of the run step to retrieve.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123/steps/step_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1"
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
{
  "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
  }
}
```

Modify run (v1)

Legacy



post https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}

Modifies a run.

Path parameters

The ID of the thread that was run.

The ID of the run to modify.

Request body

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

Returns

The modified run object matching the specified ID.

Example request

```
1
2
3
4
5
6
7
8
9
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123 \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
  "metadata": {
    "user_id": "user_abc123"
  }
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
{
  "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-4-turbo",
"instructions": null,
"incomplete_details": null,
"tools": [
  {
    "type": "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"
}
```

Submit tool outputs to run (v1)

Legacy



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

The ID of the `thread` to which this run belongs.

The ID of the run that requires the tool output submission.

Request body

A list of tools for which the outputs are being submitted.

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.

Example request

```
1
2
3
4
5
6
7
8
9
10
11
12
curl https://api.openai.com/v1/threads/thread_123/runs/run_123/submit_tool_outputs \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "Content-Type: application/json" \
-H "OpenAI-Beta: assistants=v1" \
-d '{
  "tool_outputs": [
    {
      "tool_call_id": "call_001",
      "output": "70 degrees and sunny."
    }
  ]
}'
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
{
  "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-4-turbo",
"instructions": null,
"incomplete_details": 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"]
      }
    }
  }
],
"file_ids": [],
"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"
}
```

Cancel a run (v1)

Legacy



post https://api.openai.com/v1/threads/{thread_id}/runs/{run_id}/cancel

Cancels a run that is [in_progress](#).

Path parameters

The ID of the thread to which this run belongs.

The ID of the run to cancel.

Returns

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

Example request

```
1
2
3
4
curl https://api.openai.com/v1/threads/thread_abc123/runs/run_abc123/cancel \
-H "Authorization: Bearer $OPENAI_API_KEY" \
-H "OpenAI-Beta: assistants=v1" \
-X POST
```

Response

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
{
  "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-4-turbo",
  "instructions": "You summarize books.",
  "tools": [
    {
      "type": "retrieval"
    }
  ],
  "file_ids": [],
  "metadata": {},
  "usage": null,
  "temperature": 1.0,
  "top_p": 1.0,
}
```

The run object (v1) Legacy



Represents an execution run on a [thread](#).

The identifier, which can be referenced in API endpoints.

The object type, which is always `thread.run`.

The Unix timestamp (in seconds) for when the run was created.

The ID of the [thread](#) that was executed on as a part of this run.

The ID of the [assistant](#) used for execution of this run.

The status of the run, which can be either `queued`, `in_progress`, `requires_action`, `cancelling`, `cancelled`, `failed`, `completed`, or `expired`.

Details on the action required to continue the run. Will be `null` if no action is required.

The last error associated with this run. Will be `null` if there are no errors.

The Unix timestamp (in seconds) for when the run will expire.

The Unix timestamp (in seconds) for when the run was started.

The Unix timestamp (in seconds) for when the run was cancelled.

The Unix timestamp (in seconds) for when the run failed.

The Unix timestamp (in seconds) for when the run was completed.

Details on why the run is incomplete. Will be `null` if the run is not incomplete.

The model that the [assistant](#) used for this run.

The instructions that the [assistant](#) used for this run.

The list of tools that the [assistant](#) used for this run.

The list of [File](#) IDs the [assistant](#) used for this run.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.



usage

The sampling temperature used for this run. If not set, defaults to 1.

The nucleus sampling value used for this run. If not set, defaults to 1.

The maximum number of prompt tokens specified to have been used over the course of the run.

The maximum number of completion tokens specified to have been used over the course of the run.

⌚
truncation_strategy

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 a tool. Specifying a particular tool like `{"type": "TOOL_TYPE"}` or `{"type": "function", "function": {"name": "my_function"}}` forces the model to call that tool.

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_object" }` enables JSON mode, which guarantees 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.

OBJECT The run object (v1)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
{
  "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-4-turbo",
  "instructions": null,
  "tools": [{"type": "retrieval"}, {"type": "code_interpreter"}],
  "file_ids": [],
  "metadata": {},
  "incomplete_details": null,
  "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"
}
```

The run step object (v1)

Legacy



Represents a step in execution of a run.

The identifier of the run step, which can be referenced in API endpoints.

The object type, which is always `thread.run.step`.

The Unix timestamp (in seconds) for when the run step was created.

The ID of the [assistant](#) associated with the run step.

The ID of the [thread](#) that was run.

The ID of the [run](#) that this run step is a part of.

The type of run step, which can be either `message_creation` or `tool_calls`.

The status of the run step, which can be either `in_progress`, `cancelled`, `failed`, `completed`, or `expired`.

The details of the run step.

The last error associated with this run step. Will be `null` if there are no errors.

The Unix timestamp (in seconds) for when the run step expired. A step is considered expired if the parent run is expired.

The Unix timestamp (in seconds) for when the run step was cancelled.

The Unix timestamp (in seconds) for when the run step failed.

The Unix timestamp (in seconds) for when the run step completed.

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. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

⌚
usage

OBJECT The run step object (v1)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
{
  "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
  }
}
```

Streaming (v1) Legacy



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 (v1)

Legacy



Represents a message delta i.e. any changed fields on a message during streaming.

The identifier of the message, which can be referenced in API endpoints.

The object type, which is always `thread.message.delta`.

The delta containing the fields that have changed on the Message.

OBJECT The message delta object (v1)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
{
  "id": "msg_123",
  "object": "thread.message.delta",
  "delta": {
    "content": [
      {
        "index": 0,
        "type": "text",
        "text": { "value": "Hello", "annotations": [] }
      }
    ]
  }
}
```

The run step delta object (v1)

Legacy



Represents a run step delta i.e. any changed fields on a run step during streaming.

The identifier of the run step, which can be referenced in API endpoints.

The object type, which is always `thread.run.step.delta`.

The delta containing the fields that have changed on the run step.

OBJECT The run step delta object (v1)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
{
  "id": "step_123",
  "object": "thread.run.step.delta",
  "delta": {
    "step_details": {
      "type": "tool_calls",
      "tool_calls": [
        {
          "index": 0,
          "id": "call_123",
          "type": "code_interpreter",
          "code_interpreter": { "input": "", "outputs": [] }
        }
      ]
    }
  }
}
```

Assistant stream events (v1)

Legacy



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.

Occurs when a new `thread` is created.

Occurs when a new `run` is created.

Occurs when a `run` moves to a `queued` status.

Occurs when a `run` moves to an `in_progress` status.

Occurs when a `run` moves to a `requires_action` status.

Occurs when a `run` is completed.

Occurs when a `run` fails.

Occurs when a `run` moves to a `cancelling` status.

Occurs when a `run` is cancelled.

Occurs when a `run` expires.

Occurs when a `run step` is created.

Occurs when a `run step` moves to an `in_progress` state.

Occurs when parts of a `run step` are being streamed.

Occurs when a `run step` is completed.

Occurs when a `run step` fails.

Occurs when a `run step` is cancelled.

Occurs when a `run step` expires.

Occurs when a `message` is created.

Occurs when a `message` moves to an `in_progress` state.

Occurs when parts of a `Message` are being streamed.

Occurs when a message is completed.

Occurs when a message ends before it is completed.

Occurs when an error occurs. This can happen due to an internal server error or a timeout.

Occurs when a stream ends.