Example request

### Assistants Beta &

Build assistants that can call models and use tools to perform tasks.

#### Get started with the Assistants API

### Create assistant Beta &

POST https://api.openai.com/v1/assistants

Create an assistant with a model and instructions.

# Request body

#### model Required

ID of the model to use. You can use the **List models** API to see all of your available models, or see our **Model overview** for descriptions of them.

#### name string or null Optional

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

#### description string or null Optional

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

#### instructions string or null Optional

The system instructions that the assistant uses. The maximum length is 32768 characters.

#### tools array Optional Defaults to []

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of

types code\_interpreter, retrieval, or function.

∨Show possible types

### file\_ids array Optional Defaults to []

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.

#### metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

```
Code Interpreter Files
```

node.is ∨

币

```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const myAssistant = await openai.beta.assistants.create({
7
       instructions:
8
         "You are a personal math tutor. When asked a question, w
9
       name: "Math Tutor",
10
       tools: [{ type: "code_interpreter" }],
11
        model: "gpt-4",
12
      });
13
14
      console.log(myAssistant);
15
    }
16
     main();
```

# Response

```
1
2
       "id": "asst_abc123",
3
       "object": "assistant",
4
       "created_at": 1698984975,
5
       "name": "Math Tutor",
6
       "description": null,
7
       "model": "gpt-4",
8
       "instructions": "You are a personal math tutor. When asked a
9
       "tools": [
10
         "type": "code_interpreter"
11
12
13
      ],
14
       "file_ids": [],
15
       "metadata": {}
16
```

# **Returns**

An assistant object.

### Create assistant file Beta &

POST https://api.openai.com/v1/assistants/{assistant\_id}/files

Create an assistant file by attaching a File to an assistant.

### Path parameters

assistant id string Required

The ID of the assistant for which to create a File.

# Request body

file\_id string Required

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.

### List assistants

GET https://api.openai.com/v1/assistants

Returns a list of assistants.

# **Query parameters**

limit integer Optional Defaults to 20

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

order string Optional Defaults to desc

Sort order by the created\_at timestamp of the objects. asc for ascending order and desc for descending order.

after string Optional

A cursor for use in pagination. after is an object ID that defines your place in the list. For instance, if you make a list request and

```
Example request
```

node.is ∨

```
ന
```

```
import OpenAI from "openai";
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const myAssistantFile = await openai.beta.assistants.files.cr
       "asst_abc123",
6
7
       {
8
         file_id: "file-abc123"
9
10
      );
11
      console.log(myAssistantFile);
12
13
    main();
```

#### Response

```
凸
```

```
1 {
     "id": "file-abc123",
     "object": "assistant.file",
3
4
     "created_at": 1699055364,
5
     "assistant_id": "asst_abc123"
6 }
```

### Example request

Response

node.js ∨



```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const myAssistants = await openai.beta.assistants.list({
7
       order: "desc",
8
       limit: "20",
9
      });
10
11
      console.log(myAssistants.data);
12
13
     main();
```

凸

receive 100 objects, ending with obj\_foo, your subsequent call can include after=obj\_foo in order to fetch the next page of the list.

#### before string Optional

A cursor for use in pagination. before is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, 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.

### List assistant files Beta &

GET https://api.openai.com/v1/assistants/{assistant\_id}/files

Returns a list of assistant files.

# Path parameters

```
assistant_id string Required
```

```
1
2
       "object": "list",
3
       "data": [
4
        {
5
         "id": "asst abc123",
6
         "object": "assistant",
7
         "created_at": 1698982736,
8
         "name": "Coding Tutor",
9
         "description": null,
10
         "model": "gpt-4",
11
         "instructions": "You are a helpful assistant designed to ma
12
         "tools": [],
13
         "file_ids": [],
14
         "metadata": {}
15
16
17
         "id": "asst_abc456",
18
         "object": "assistant",
19
         "created_at": 1698982718,
20
         "name": "My Assistant",
21
         "description": null,
22
         "model": "gpt-4",
23
         "instructions": "You are a helpful assistant designed to ma
24
         "tools": [],
25
         "file ids": [].
26
         "metadata": {}
27
        },
28
        {
         "id": "asst_abc789",
29
30
         "object": "assistant",
31
         "created_at": 1698982643,
32
         "name": null,
33
         "description": null,
34
         "model": "gpt-4",
35
         "instructions": null,
36
         "tools": [],
37
         "file_ids": [],
38
         "metadata": {}
39
        }
40
       "first_id": "asst_abc123",
41
42
       "last_id": "asst_abc789",
43
       "has_more": false
44 }
```

```
Example request
```

```
node.js ∨
```



```
import OpenAl from "openai";
const openai = new OpenAl();

async function main() {
   const assistantFiles = await openai.beta.assistants.files.list(
   "asst_abc123"
   );
```

The ID of the assistant the file belongs to.

# **Query parameters**

limit integer Optional Defaults to 20

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

order string Optional Defaults to desc

Sort order by the created\_at timestamp of the objects. asc for ascending order and desc for descending order.

after string Optional

A cursor for use in pagination. after is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj\_foo, your subsequent call can include after=obj\_foo in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. before is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, 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.

#### Retrieve assistant Beta 🔗

GET https://api.openai.com/v1/assistants/{assistant\_id}

Retrieves an assistant.

### Path parameters

assistant id string Required

The ID of the assistant to retrieve.

#### **Returns**

The assistant object matching the specified ID.

```
console.log(assistantFiles);
9
     }
10
    main();
11
```

```
Response
```

1

```
币
```

```
{
2
       "object": "list",
3
       "data": [
4
        {
5
          "id": "file-abc123",
6
         "object": "assistant.file",
7
         "created_at": 1699060412,
8
         "assistant_id": "asst_abc123"
9
10
11
         "id": "file-abc456".
12
         "object": "assistant.file",
13
         "created_at": 1699060412,
14
         "assistant_id": "asst_abc123"
15
        }
16
       ],
17
       "first id": "file-abc123",
18
       "last_id": "file-abc456",
       "has_more": false
19
20
```

### Example request

node.js ∨



```
1
     import OpenAI from "openai";
2
3
     const openai = new OpenAI();
5
     async function main() {
6
      const myAssistant = await openai.beta.assistants.retrieve(
7
       "asst_abc123"
8
      );
9
10
      console.log(myAssistant);
11
12
    main();
```



```
1
2
      "id": "asst_abc123",
3
      "object": "assistant",
4
      "created_at": 1699009709,
5
      "name": "HR Helper",
6
      "description": null,
7
      "model": "gpt-4",
8
      "instructions": "You are an HR bot, and you have access to fi
9
      "tools": [
10
       {
11
         "type": "retrieval"
12
13
      ],
14
      "file_ids": [
       "file-abc123"
15
16
17
      "metadata": {}
18 }
```

# Retrieve assistant file Beta &

GET https://api.openai.com/v1/assistants/{assistant\_id}/files/{file\_id}

Retrieves an AssistantFile.

# Path parameters

assistant\_id string Required

The ID of the assistant who the file belongs to.

file\_id string Required

The ID of the file we're getting.

#### **Returns**

The assistant file object matching the specified ID.

#### neturns

# Modify assistant Beta &

POST https://api.openai.com/v1/assistants/{assistant\_id}

Modifies an assistant.

# Path parameters

```
1
     import OpenAI from "openai";
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const myAssistantFile = await openai.beta.assistants.files.re
6
       "asst_abc123",
7
       "file-abc123"
8
9
      console.log(myAssistantFile);
10
    }
```

#### Response

12 main();

11

Example request

```
1 {
2 "id": "file-abc123",
3 "object": "assistant.file",
4 "created_at": 1699055364,
5 "assistant_id": "asst_abc123"
6 }
```

import OpenAI from "openai";

const openai = new OpenAI();

async function main() {

### Example request

1

2

4

# node.js∨

node.js∨

 $\Box$ 

币

凸

```
https://platform.openai.com/docs/api-reference/runs/step-object
```

assistant id string Required

The ID of the assistant to modify.

# Request body

#### model Optional

ID of the model to use. You can use the **List models** API to see all of your available models, or see our **Model overview** for descriptions of them.

#### name string or null Optional

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

#### description string or null Optional

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

#### instructions string or null Optional

The system instructions that the assistant uses. The maximum length is 32768 characters.

#### tools array Optional Defaults to []

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of

types code\_interpreter, retrieval, or function.

∨Show possible types

#### file ids array Optional Defaults to []

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.

#### metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

### **Returns**

The modified assistant object.

# Delete assistant Beta &

DELETE https://api.openai.com/v1/assistants/{assistant\_id}

```
const myUpdatedAssistant = await openai.beta.assistants.ur
7
       "asst_abc123",
8
       {
9
         instructions:
          "You are an HR bot, and you have access to files to answ
10
11
         name: "HR Helper",
12
         tools: [{ type: "retrieval" }],
13
         model: "gpt-4",
14
         file_ids: [
15
          "file-abc123",
16
          "file-abc456".
17
        ],
18
       }
19
      );
20
21
      console.log(myUpdatedAssistant);
22
     }
23
```

#### Response

main();

24



```
1
2
       "id": "asst abc123".
3
       "object": "assistant",
       "created_at": 1699009709,
4
5
       "name": "HR Helper",
6
       "description": null,
7
       "model": "gpt-4",
8
       "instructions": "You are an HR bot, and you have access to fi
9
10
        {
         "type": "retrieval"
11
12
13
14
       "file ids": [
15
        "file-abc123".
16
        "file-abc456"
17
18
       "metadata": {}
19
    }
```

Response

Delete an assistant.

# **Path parameters**

assistant\_id string Required
The ID of the assistant to delete.

### **Returns**

Deletion status

# Delete assistant file Beta &

DELETE https://api.openai.com/v1/assistants/{assistant\_id}/files/{file\_i d}

Delete an assistant file.

# Path parameters

assistant\_id string Required

The ID of the assistant that the file belongs to.

file\_id string Required

The ID of the file to delete.

#### Returns

Deletion status

# The assistant object Beta &

Represents an assistant that can call the model and use tools.

id string

The identifier, which can be referenced in API endpoints.

object string

```
import OpenAI from "openai";
1
2
3
     const openai = new OpenAI();
4
     async function main() {
5
6
      const response = await openai.beta.assistants.del("asst_abc
7
8
      console.log(response);
9
    }
10 main();
```

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

币

 $\Box$ 

币

币

node.js∨

```
Example request

1 import OpenAI from "openai";
```

```
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const deletedAssistantFile = await openai.beta.assistants.file
6
       "asst_abc123",
7
       "file-abc123"
8
9
      console.log(deletedAssistantFile);
10
11
    main();
```

```
Response
```

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

```
The assistant object
```

```
1 {
2 "id": "asst_abc123",
3 "object": "assistant",
4 "created_at": 1698984975,
```

The object type, which is always assistant.

#### created\_at integer

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

#### name string or null

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

#### description string or null

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

#### model string

ID of the model to use. You can use the **List models** API to see all of your available models, or see our **Model overview** for descriptions of them.

#### instructions string or null

The system instructions that the assistant uses. The maximum length is 32768 characters.

#### tools array

A list of tool enabled on the assistant. There can be a maximum of 128 tools per assistant. Tools can be of types code\_interpreter , retrieval , or function . ~Show possible types

#### file\_ids array

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.

#### metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

# The assistant file object Beta &

A list of Files attached to an assistant.

#### id string

The identifier, which can be referenced in API endpoints.

#### object string

The object type, which is always assistant.file .

#### created\_at integer

```
5
       "name": "Math Tutor",
6
       "description": null,
7
       "model": "gpt-4",
8
       "instructions": "You are a personal math tutor. When asked a
9
       "tools": [
10
        {
         "type": "code_interpreter"
11
12
13
       "file_ids": [],
14
15
       "metadata": {}
16
    }
```

The assistant file object

```
ð
```

```
1 {
2 "id": "file-abc123",
3 "object": "assistant.file",
4 "created_at": 1699055364,
5 "assistant_id": "asst_abc123"
6 }
```

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

#### assistant\_id string

The assistant ID that the file is attached to.

### Threads Beta &

Create threads that assistants can interact with.

Related guide: Assistants

# Create thread Beta &

POST https://api.openai.com/v1/threads

Create a thread.

# Request body

messages array Optional

A list of **messages** to start the thread with.

√Show properties

#### metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### Returns

A thread object.

### Retrieve thread Beta &

GET https://api.openai.com/v1/threads/{thread\_id}

Retrieves a thread.

# **Path parameters**

thread\_id string Required

The ID of the thread to retrieve.

#### Returns

```
Empty
                                                   Messages
Example request
                                                         \Box
                                           node.js ∨
     import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const emptyThread = await openai.beta.threads.create();
7
8
      console.log(emptyThread);
9
     }
10
     main();
                                                         句
Response
1 {
2
     "id": "thread_abc123",
3
     "object": "thread",
     "created at": 1699012949.
     "metadata": {}
6 }
```

Example request

1 2 3

4 5

7

8

9 10

> 11 } 12

);

main();

import OpenAI from "openai";

const openai = new OpenAI();

const myThread = await openai.beta.threads.retrieve(

async function main() {

"thread\_abc123"

console.log(myThread);

node.js ∨

 $\Box$ 

The thread object matching the specified ID.

#### Response

```
凸
```

```
1 {
    "id": "thread_abc123",
    "object": "thread",
    "created_at": 1699014083,
    "metadata": {}
6 }
```

# Modify thread Beta ₽

POST https://api.openai.com/v1/threads/{thread\_id}

Modifies a thread.

# Path parameters

thread\_id string Required

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

# Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maxium of 512 characters long.

### Returns

The modified thread object matching the specified ID.

Delete thread Beta &

DELETE https://api.openai.com/v1/threads/{thread\_id}

Delete a thread.

### Path parameters

thread\_id string Required

```
Example request
```

```
node.is ∨
```



```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const updatedThread = await openai.beta.threads.update(
7
       "thread_abc123",
8
       {
9
        metadata: { modified: "true", user: "abc123" },
10
       }
11
      );
12
13
      console.log(updatedThread);
14
    }
15
     main();
```



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

```
Example request
```



```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
5
     async function main() {
6
      const response = await openai.beta.threads.del("thread_abc
```

API Reference - OpenAI API

The ID of the thread to delete.

### **Returns**

Deletion status

# The thread object

Represents a thread that contains messages.

#### id string

The identifier, which can be referenced in API endpoints.

#### object string

The object type, which is always thread .

#### created\_at integer

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

#### metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maxium of 512 characters long.

# Messages Beta €

Create messages within threads

Related guide: Assistants

#### Create message Beta ₽

POST https://api.openai.com/v1/threads/{thread\_id}/messages

Create a message.

# Path parameters

thread\_id string Required

The ID of the thread to create a message for.

# Request body

role string Required

```
console.log(response);
9
   }
10 main();
```

```
Response
```

```
2
     "id": "thread_abc123",
     "object": "thread.deleted",
     "deleted": true
5 }
```

#### The thread object

```
1 {
    "id": "thread_abc123",
    "object": "thread",
    "created_at": 1698107661,
    "metadata": {}
6 }
```

#### Example request

```
node.js ∨
```



凸

D

```
1
     import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const threadMessages = await openai.beta.threads.message
7
       "thread_abc123",
8
       { role: "user", content: "How does AI work? Explain it in sim
9
10
11
      console.log(threadMessages);
12
13
14
```

The role of the entity that is creating the message. Currently only user is supported.

content string Required

The content of the message.

file\_ids array Optional Defaults to []

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.

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### Returns

A message object.

# List messages Beta &

GET https://api.openai.com/v1/threads/{thread\_id}/messages

Returns a list of messages for a given thread.

# Path parameters

thread id string Required

The ID of the thread the messages belong to.

# **Query parameters**

limit integer Optional Defaults to 20

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

order string Optional Defaults to desc

Sort order by the created\_at timestamp of the objects. asc for ascending order and desc for descending order.

after string Optional

A cursor for use in pagination. after is an object ID that defines your place in the list. For instance, if you make a list request and

```
main();
```

Response

```
2
       "id": "msg_abc123",
3
       "object": "thread.message",
4
       "created_at": 1699017614,
5
       "thread_id": "thread_abc123",
       "role": "user",
7
       "content": [
8
9
         "type": "text",
10
         "text": {
11
           "value": "How does AI work? Explain it in simple terms.",
12
           "annotations": []
13
         }
14
        }
15
16
       "file_ids": [],
       "assistant_id": null,
17
18
       "run_id": null,
       "metadata": {}
```

```
Example request
```

20 }

```
node.js ∨
```



凸

```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const threadMessages = await openai.beta.threads.message
7
       "thread_abc123"
8
      );
9
10
      console.log(threadMessages.data);
11
    }
12
    main();
```

```
Response
```



receive 100 objects, ending with obj\_foo, your subsequent call can include after=obj\_foo in order to fetch the next page of the list.

#### before string Optional

A cursor for use in pagination. before is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, 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 objects.

# List message files Beta &

GET https://api.openai.com/v1/threads/{thread\_id}/messages/{message} e\_id}/files

Returns a list of message files.

# Path parameters

#### thread id string Required

The ID of the thread that the message and files belong to.

```
9
         "role": "user",
10
         "content": [
11
          {
12
            "type": "text",
13
            "text": {
14
             "value": "How does AI work? Explain it in simple terms
15
             "annotations": []
16
           }
17
          }
18
         ],
19
         "file_ids": [],
20
         "assistant_id": null,
21
         "run_id": null,
22
         "metadata": {}
23
        },
24
25
         "id": "msg_abc456",
         "object": "thread.message",
26
27
         "created_at": 1699016383,
28
         "thread_id": "thread_abc123",
         "role": "user",
29
30
         "content": [
31
            "type": "text",
32
33
            "text": {
34
             "value": "Hello, what is AI?",
35
             "annotations": []
36
37
          }
38
         ],
39
         "file_ids": [
40
          "file-abc123"
41
42
         "assistant_id": null,
43
         "run_id": null,
44
         "metadata": {}
45
        }
46
47
       "first_id": "msg_abc123",
       "last_id": "msg_abc456",
48
49
       "has_more": false
50 }
```

```
Example request
```

```
node.js∨
```



```
import OpenAI from "openai";
1
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const messageFiles = await openai.beta.threads.messages.f
6
       "thread_abc123",
7
       "msg_abc123"
8
      );
      console.log(messageFiles);
```

message\_id string Required

The ID of the message that the files belongs to.

# 10 } 11 12 main();

# **Query parameters**

limit integer Optional Defaults to 20

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

order string Optional Defaults to desc

Sort order by the created\_at timestamp of the objects. asc for ascending order and desc for descending order.

after string Optional

A cursor for use in pagination. after is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj\_foo, your subsequent call can include after=obj\_foo in order to fetch the next page of the list.

before string Optional

A cursor for use in pagination. before is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj\_foo, your subsequent call can include before=obj\_foo in order to fetch the previous page of the list.

#### Response

```
2
      "object": "list",
3
      "data": [
        {
5
         "id": "file-abc123",
         "object": "thread.message.file",
6
7
         "created_at": 1699061776,
R
         "message_id": "msg_abc123"
9
        },
10
         "id": "file-abc123",
11
12
         "object": "thread.message.file",
13
         "created_at": 1699061776,
14
         "message_id": "msg_abc123"
15
16
      "first id": "file-abc123",
17
18
      "last id": "file-abc123",
19
      "has_more": false
20 }
```

Example request

11

12 } 13

main();

console.log(message);

### **Returns**

A list of message file objects.

# Retrieve message Beta &

GET https://api.openai.com/v1/threads/{thread\_id}/messages/{message\_id}

Retrieve a message.

# Path parameters

thread\_id string Required

The ID of the thread to which this message belongs.

https://platform.openai.com/docs/api-reference/runs/step-object

message\_id string Required

The ID of the message to retrieve.

### **Returns**

#### 1 import OpenAI from "openai"; 2 3 const openai = new OpenAI(); 4 5 async function main() { 6 const message = await openai.beta.threads.messages.retrie 7 "thread\_abc123", "msg\_abc123" 8 9 ); 10

node.js ∨

句

Response

The message object matching the specified ID.

```
1
       "id": "msg_abc123",
3
      "object": "thread.message",
4
       "created_at": 1699017614,
5
       "thread_id": "thread_abc123",
6
       "role": "user",
7
       "content": [
8
9
         "type": "text",
10
         "text": {
          "value": "How does AI work? Explain it in simple terms.",
11
12
          "annotations": []
13
         }
14
        }
15
      ],
16
      "file_ids": [],
       "assistant_id": null,
17
      "run_id": null,
18
       "metadata": {}
19
```

# Retrieve message file Beta &

GET https://api.openai.com/v1/threads/{thread\_id}/messages/{message\_id}/files/{file\_id}

Retrieves a message file.

# Path parameters

thread\_id string Required

The ID of the thread to which the message and File belong.

message\_id string Required

The ID of the message the file belongs to.

file\_id string Required

The ID of the file being retrieved.

#### Returns

The message file object.

# Modify message Beta ₽

POST https://api.openai.com/v1/threads/{thread\_id}/messages/{message\_id}

Example request

20 }

node.js∨

 $\Box$ 

凸

```
1
     import OpenAI from "openai";
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const messageFile = await openai.beta.threads.messages.fil
6
       "thread_abc123",
7
       "msg_abc123",
8
       "file-abc123"
9
10
      console.log(messageFile);
11
12
13 main();
```

1 {
2 "id": "file-abc123",
3 "object": "thread.message.file",
4 "created\_at": 1699061776,
5 "message\_id": "msg\_abc123"
6 }

Example request

Response

node.js ∨



ð

Response

Modifies a message.

# **Path parameters**

thread\_id string Required

The ID of the thread to which this message belongs.

message id string Required

The ID of the message to modify.

# Request body

#### metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### Returns

The modified message object.

# The message object Beta ₽

Represents a message within a thread.

id string

The identifier, which can be referenced in API endpoints.

object string

The object type, which is always thread.message .

created\_at integer

```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
     async function main() {
5
6
      const message = await openai.beta.threads.messages.upda
7
       "thread_abc123",
8
       "msg_abc123",
9
       {
10
        metadata: {
11
          modified: "true",
12
          user: "abc123",
13
        },
14
       }
15
      }'
```

```
1
2
       "id": "msg_abc123",
3
       "object": "thread.message",
4
       "created_at": 1699017614,
5
       "thread_id": "thread_abc123",
6
       "role": "user",
7
       "content": [
8
9
         "type": "text",
10
         "text": {
11
          "value": "How does AI work? Explain it in simple terms.",
12
          "annotations": []
13
         }
14
        }
15
      ],
16
       "file_ids": [],
17
       "assistant id": null,
18
       "run_id": null,
19
       "metadata": {
20
        "modified": "true",
21
        "user": "abc123"
22
      }
23 }
```

```
The message object
```

```
1 {
2 "id": "msg_abc123",
3 "object": "thread.message",
4 "created_at": 1698983503,
5 "thread_id": "thread_abc123",
6 "role": "assistant",
7 "content": [
8 {
```

ð

凸

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

#### thread\_id string

The thread ID that this message belongs to.

#### role string

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

#### content array

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

∨Show possible types

#### assistant\_id string or null

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

#### run\_id string or null

If applicable, the ID of the **run** associated with the authoring of this message.

#### file\_ids array

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.

#### metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

# The message file object Beta &

A list of files attached to a message .

#### id string

The identifier, which can be referenced in API endpoints.

#### object string

The object type, which is always thread.message.file .

#### created\_at integer

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

#### message\_id string

The ID of the message that the File is attached to.

### Runs Beta &

Represents an execution run on a thread.

Related guide: Assistants

```
9
         "type": "text",
10
         "text": {
11
          "value": "Hi! How can I help you today?",
12
          "annotations": []
13
         }
14
        }
15
      ],
16
       "file_ids": [],
       "assistant_id": "asst_abc123",
17
       "run_id": "run_abc123",
18
19
       "metadata": {}
20 }
```

The message file object

```
币
```

```
1 {
2  "id": "file-abc123",
3  "object": "thread.message.file",
4  "created_at": 1698107661,
5  "message_id": "message_QLoltBbqwyAJEzITy4y9kOMM",
6  "file_id": "file-abc123"
7 }
```

# Create run Beta &

POST https://api.openai.com/v1/threads/{thread\_id}/runs

Create a run.

# Path parameters

thread\_id string Required The ID of the thread to run.

# **Request body**

assistant\_id string Required

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

#### model string or null 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.

#### instructions string or null Optional

Overrides the **instructions** of the assistant. This is useful for modifying the behavior on a per-run basis.

#### additional\_instructions string or null Optional

Appends additional instructions at the end of the instructions for the run. This is useful for modifying the behavior on a per-run basis without overriding other instructions.

#### tools array or null Optional

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

∨Show possible types

#### metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### **Returns**

A run object.

```
Example request
```

```
node.js ∨
```

```
Ð
```

```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const run = await openai.beta.threads.runs.create(
7
       "thread_abc123",
8
       { assistant_id: "asst_abc123" }
9
10
11
      console.log(run);
12
13
    main();
```



```
1
       "id": "run_abc123",
3
       "object": "thread.run",
       "created_at": 1699063290,
4
5
       "assistant_id": "asst_abc123",
6
       "thread_id": "thread_abc123",
       "status": "queued",
7
8
       "started_at": 1699063290,
9
       "expires_at": null,
10
       "cancelled_at": null,
11
       "failed_at": null,
12
       "completed_at": 1699063291,
13
       "last_error": null,
14
       "model": "gpt-4",
15
       "instructions": null,
16
       "tools": [
17
18
         "type": "code_interpreter"
19
        }
20
      ],
21
       "file_ids": [
22
        "file-abc123".
23
        "file-abc456"
24
25
       "metadata": {},
26
       "usage": null
27 }
```

# Create thread and run Beta &

POST https://api.openai.com/v1/threads/runs

Create a thread and run it in one request.

# Request body

assistant id string Required

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

thread object Optional

√Show properties

model string or null 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.

instructions string or null Optional

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

tools array or null Optional

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

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### **Returns**

A run object.

### List runs Beta &

GET https://api.openai.com/v1/threads/{thread\_id}/runs

Returns a list of runs belonging to a thread.

# **Path parameters**

```
Example request
```

node.js ∨

```
凸
```

```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const run = await openai.beta.threads.createAndRun({
7
        assistant_id: "asst_abc123",
8
        thread: {
9
         messages:
10
          { role: "user", content: "Explain deep learning to a 5 year
11
        ],
12
       },
13
      });
14
15
      console.log(run);
16
     }
17
18
     main();
```

### Response

币

```
1
2
       "id": "run_abc123",
3
       "object": "thread.run",
4
       "created_at": 1699076792,
5
       "assistant_id": "asst_abc123",
       "thread_id": "thread_abc123",
6
7
       "status": "queued",
8
       "started_at": null,
9
       "expires_at": 1699077392,
10
       "cancelled_at": null,
11
       "failed_at": null,
12
       "completed_at": null,
13
       "last_error": null,
14
       "model": "gpt-4",
15
       "instructions": "You are a helpful assistant.",
16
       "tools": [],
17
       "file_ids": [],
       "metadata": {}.
19
       "usage": null
20 }
```

# Example request

node.js ∨



```
1 import OpenAl from "openai";2
```

```
3 const openai = new OpenAI();
```

#### thread id string Required

The ID of the thread the run belongs to.

# **Query parameters**

limit integer Optional Defaults to 20

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

### order string Optional Defaults to desc

Sort order by the created\_at timestamp of the objects. asc for ascending order and desc for descending order.

#### after string Optional

A cursor for use in pagination. after is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj\_foo, your subsequent call can include after=obj\_foo in order to fetch the next page of the list.

#### before string Optional

A cursor for use in pagination. before is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, 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.

```
5  async function main() {
6   const runs = await openai.beta.threads.runs.list(
7   "thread_abc123"
8  );
9  
10   console.log(runs);
11  }
12  
13  main();
```

```
ð
```

```
1
2
       "object": "list",
3
       "data": [
4
5
         "id": "run_abc123",
6
         "object": "thread.run",
7
         "created_at": 1699075072,
8
         "assistant_id": "asst_abc123",
9
         "thread_id": "thread_abc123",
10
         "status": "completed",
11
         "started_at": 1699075072,
12
         "expires at": null,
13
         "cancelled_at": null,
14
         "failed_at": null,
15
         "completed_at": 1699075073,
16
         "last_error": null,
17
         "model": "gpt-3.5-turbo",
18
         "instructions": null.
         "tools": [
19
20
21
            "type": "code_interpreter"
22
          }
23
         ],
         "file ids": [
24
25
          "file-abc123".
          "file-abc456"
26
27
         ],
28
         "metadata": {},
29
         "usage": {
30
           "prompt_tokens": 123,
31
          "completion_tokens": 456,
32
          "total_tokens": 579
33
         }
34
        },
35
36
         "id": "run_abc456",
37
         "object": "thread.run",
38
         "created_at": 1699063290,
39
         "assistant_id": "asst_abc123",
         "thread_id": "thread_abc123",
40
41
         "status": "completed",
42
         "started_at": 1699063290,
43
         "expires_at": null,
44
         "cancelled_at": null,
```

```
45
         "failed_at": null,
46
         "completed_at": 1699063291,
47
         "last_error": null,
48
         "model": "gpt-3.5-turbo",
49
         "instructions": null.
50
         "tools": [
51
          {
52
            "type": "code_interpreter"
53
          }
54
         ],
55
         "file_ids": [
56
          "file-abc123".
57
          "file-abc456"
58
59
         "metadata": {},
60
         "usage": {
61
          "prompt_tokens": 123,
62
          "completion_tokens": 456,
63
          "total_tokens": 579
64
         }
65
        }
66
      ],
67
      "first_id": "run_abc123",
68
      "last_id": "run_abc456",
69
      "has more": false
70 }
```

# 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

thread id string Required

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

run id string Required

The ID of the run the run steps belong to.

# **Query parameters**

limit integer Optional Defaults to 20

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

order string Optional Defaults to desc

Sort order by the created\_at timestamp of the objects. asc for ascending order and desc for descending order.

after string Optional

```
Example request
```

node.js∨



```
import OpenAI from "openai";
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const runStep = await openai.beta.threads.runs.steps.list(
6
       "thread abc123".
7
       "run_abc123"
8
9
      console.log(runStep);
10
     }
11
12 main();
```



```
1 {
2  "object": "list",
3  "data": [
4  {
5    "id": "step_abc123",
6    "object": "thread.run.step",
7    "created_at": 1699063291,
8    "run_id": "run_abc123",
9    "assistant_id": "asst_abc123",
```

A cursor for use in pagination. after is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, ending with obj\_foo, your subsequent call can include after=obj\_foo in order to fetch the next page of the list.

#### before string Optional

A cursor for use in pagination. before is an object ID that defines your place in the list. For instance, if you make a list request and receive 100 objects, 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.

### Retrieve run Beta &

GET https://api.openai.com/v1/threads/{thread\_id}/runs/{run\_id}

Retrieves a run.

# Path parameters

thread\_id string Required
The ID of the thread that was run.

run\_id string Required
The ID of the run to retrieve.

#### Returns

The run object matching the specified ID.

```
10
         "thread_id": "thread_abc123",
11
         "type": "message_creation",
12
         "status": "completed",
13
         "cancelled_at": null,
14
         "completed_at": 1699063291,
15
         "expired_at": null,
16
         "failed_at": null,
17
         "last_error": null,
18
         "step_details": {
19
          "type": "message_creation",
20
          "message_creation": {
21
           "message_id": "msg_abc123"
22
          }
23
         },
24
         "usage": {
25
          "prompt_tokens": 123,
          "completion_tokens": 456,
26
27
          "total_tokens": 579
28
        }
29
       }
30
31
      "first_id": "step_abc123",
32
      "last_id": "step_abc456",
33
      "has more": false
34 }
```

```
Example request
```

```
node.js∨
```

```
Д
```

```
import OpenAI from "openai";
1
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const run = await openai.beta.threads.runs.retrieve(
7
       "thread abc123".
8
       "run_abc123"
9
      );
10
11
      console.log(run);
12
     }
13
     main();
```



```
    {
    "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,
10
      "cancelled_at": null,
11
      "failed_at": null,
12
      "completed_at": 1699075073,
13
      "last_error": null,
14
      "model": "gpt-3.5-turbo",
15
      "instructions": null,
16
      "tools": [
17
18
         "type": "code_interpreter"
19
        }
20
      ],
21
      "file_ids": [
        "file-abc123".
22
        "file-abc456"
23
24
25
      "metadata": {},
26
      "usage": {
27
        "prompt_tokens": 123,
28
        "completion_tokens": 456,
        "total_tokens": 579
29
30
      }
31 }
```

Example request

Response

# 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**

thread\_id string Required

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

run\_id string Required

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

step\_id string Required

The ID of the run step to retrieve.

#### Returns

The run step object matching the specified ID.

```
1
     import OpenAI from "openai";
2
     const openai = new OpenAI();
3
4
     async function main() {
5
      const runStep = await openai.beta.threads.runs.steps.retriev
6
       "thread_abc123",
7
       "run_abc123",
8
       "step_abc123"
9
10
      console.log(runStep);
11
12
13
    main();
```

node.js ∨

ð

凸

```
1
      "id": "step_abc123",
2
3
      "object": "thread.run.step",
4
      "created_at": 1699063291,
5
      "run_id": "run_abc123",
6
      "assistant_id": "asst_abc123",
      "thread_id": "thread_abc123",
7
8
      "type": "message_creation",
9
      "status": "completed",
10
      "cancelled_at": null,
      "completed_at": 1699063291,
```

```
12
      "expired_at": null,
      "failed_at": null,
13
14
      "last_error": null,
15
      "step_details": {
       "type": "message_creation",
16
17
       "message_creation": {
18
        "message_id": "msg_abc123"
19
       }
20
      "usage": {
21
22
       "prompt_tokens": 123,
23
       "completion_tokens": 456,
24
       "total_tokens": 579
25
26 }
```

#### **Modify run** Beta ∂

POST https://api.openai.com/v1/threads/{thread\_id}/runs/{run\_id}

Modifies a run.

# Path parameters

thread id string Required The ID of the thread that was run.

run\_id string Required The ID of the run to modify.

# Request body

metadata map Optional

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maxium of 512 characters long.

#### Returns

The modified run object matching the specified ID.

```
1
```

Example request

```
node.js ∨
```

```
仚
```

```
import OpenAI from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
6
      const run = await openai.beta.threads.runs.update(
7
       "thread_abc123",
8
       "run_abc123",
9
10
        metadata: {
11
          user_id: "user_abc123",
12
        },
13
14
      );
15
16
      console.log(run);
17
     }
18
     main();
```



```
1
      "id": "run_abc123",
2
      "object": "thread.run",
3
4
      "created_at": 1699075072,
5
      "assistant_id": "asst_abc123",
6
      "thread_id": "thread_abc123",
7
      "status": "completed",
8
      "started_at": 1699075072,
9
      "expires_at": null,
10
      "cancelled_at": null,
11
      "failed_at": null,
12
      "completed_at": 1699075073,
      "last_error": null,
```

```
"model": "gpt-3.5-turbo",
      "instructions": null,
15
      "tools": [
16
17
       {
         "type": "code_interpreter"
18
19
20
      ],
21
      "file_ids": [
       "file-abc123",
22
       "file-abc456"
23
24
      ],
25
      "metadata": {
26
       "user_id": "user_abc123"
27
28
      "usage": {
29
        "prompt_tokens": 123,
30
       "completion_tokens": 456,
31
       "total_tokens": 579
32
33 }
```

# Submit tool outputs to run Beta &

POST https://api.openai.com/v1/threads/{thread\_id}/runs/{run\_id}/sub mit\_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**

thread\_id string Required

The ID of the thread to which this run belongs.

run\_id string Required

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

# Request body

tool\_outputs array Required

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

∨ Show properties

### **Returns**

The modified run object matching the specified ID.

```
Example request
```

```
1
     import OpenAl from "openai";
2
3
     const openai = new OpenAI();
4
5
     async function main() {
      const run = await openai.beta.threads.runs.submitToolOutpu
7
       "thread_abc123",
8
       "run_abc123",
9
10
        tool_outputs: [
11
12
           tool_call_id: "call_abc123",
13
           output: "28C",
14
         },
15
        ],
16
       }
17
      );
18
19
      console.log(run);
20
    }
21
    main();
```

node.js ∨

O

#### Response

```
    {
    "id": "run_abc123",
    "object": "thread.run",
    "created_at": 1699075592,
    "assistant_id": "asst_abc123",
```

D

```
6
       "thread_id": "thread_abc123",
7
       "status": "queued",
8
       "started_at": 1699075592,
9
       "expires_at": 1699076192,
10
      "cancelled_at": null,
11
       "failed_at": null,
12
       "completed_at": null,
13
       "last_error": null,
14
       "model": "gpt-4",
15
      "instructions": "You tell the weather.",
16
       "tools": [
17
       {
18
         "type": "function",
         "function": {
19
20
          "name": "get_weather",
21
          "description": "Determine weather in my location",
22
          "parameters": {
23
            "type": "object",
24
            "properties": {
25
             "location": {
26
               "type": "string",
27
               "description": "The city and state e.g. San Francisco,
28
             },
29
             "unit": {
30
              "type": "string",
31
               "enum": [
32
                "c",
33
                "f"
34
35
             }
36
            },
37
            "required": [
             "location"
38
39
40
          }
41
         }
42
       }
43
      ],
44
      "file_ids": [],
45
      "metadata": {},
46
       "usage": null
47 }
```

### Cancel a run Beta &

POST https://api.openai.com/v1/threads/{thread\_id}/runs/{run\_id}/canc

Cancels a run that is in\_progress.

# **Path parameters**

thread\_id string Required
The ID of the thread to which this run belongs.

```
Example request
```

```
node.js∨
```



```
import OpenAl from "openai";

const openai = new OpenAl();

async function main() {
 const run = await openai.beta.threads.runs.cancel(
 "thread_abc123",
 "run_abc123"
);
```

12 } 13

Response

24 }

18

"metadata": {},

The run object

main();

console.log(run);

10

11

run\_id string Required

The ID of the run to cancel.

#### Returns

The modified run object matching the specified ID.

```
1
2
       "id": "run_abc123",
3
      "object": "thread.run",
4
       "created_at": 1699076126,
5
       "assistant_id": "asst_abc123",
6
       "thread_id": "thread_abc123",
7
       "status": "cancelling",
8
       "started_at": 1699076126,
9
       "expires_at": 1699076726,
10
       "cancelled_at": null,
       "failed_at": null,
11
12
       "completed_at": null,
13
       "last_error": null,
14
       "model": "gpt-4",
       "instructions": "You summarize books.",
15
16
       "tools": [
17
        {
         "type": "retrieval"
18
19
        }
20
21
      "file_ids": [],
      "metadata": {},
       "usage": null
```

币

# The run object Beta ₽

Represents an execution run on a thread.

#### id string

The identifier, which can be referenced in API endpoints.

#### object string

The object type, which is always thread.run.

#### created\_at integer

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

#### thread\_id string

The ID of the thread that was executed on as a part of this run.

#### assistant id string

The ID of the assistant used for execution of this run.

status string

```
1
2
      "id": "run_abc123",
3
      "object": "thread.run",
4
      "created_at": 1698107661,
5
      "assistant_id": "asst_abc123",
6
      "thread_id": "thread_abc123",
7
      "status": "completed",
8
      "started_at": 1699073476,
9
      "expires_at": null,
      "cancelled_at": null,
10
11
      "failed_at": null,
12
      "completed_at": 1699073498,
13
      "last_error": null,
      "model": "gpt-4",
14
15
      "instructions": null,
      "tools": [{"type": "retrieval"}, {"type": "code_interpreter"}],
16
17
      "file_ids": [],
```

ð

```
The status of the run, which can be

19 "usage": {
either queued, in_progress, requires_action, cancelling, cancelled, faile0, coinstant_tokens": 123,
or expired.

21 "completion_tokens": 456,
22 "total_tokens": 579

required_action object or null

23 }

24 }
```

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

∨Show properties

#### last error object or null

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

∨ Show properties

#### expires\_at integer

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

#### started at integer or null

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

#### cancelled\_at integer or null

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

#### failed\_at integer or null

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

### completed\_at integer or null

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

#### model string

The model that the assistant used for this run.

#### instructions string

The instructions that the assistant used for this run.

#### tools array

The list of tools that the assistant used for this run.

∨Show possible types

### file\_ids array

The list of File IDs the assistant used for this run.

### metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### usage object or null

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 run step object Beta &

Represents a step in execution of a run.

#### id string

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

#### object string

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

#### created\_at integer

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

#### assistant id string

The ID of the assistant associated with the run step.

#### thread id string

The ID of the thread that was run.

#### run\_id string

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

#### type string

The type of run step, which can be either message\_creation or tool\_calls.

#### status string

The status of the run step, which can be either in\_progress, cancelled, failed, completed, or expired.

### step\_details object

The details of the run step.

∨Show possible types

#### last\_error object or null

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

∨Show properties

#### expired at integer or null

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

#### cancelled\_at integer or null

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

#### failed\_at integer or null

```
The run step object
```

```
1
2
      "id": "step_abc123",
3
      "object": "thread.run.step",
4
      "created_at": 1699063291,
5
      "run_id": "run_abc123",
6
      "assistant_id": "asst_abc123",
7
      "thread_id": "thread_abc123",
8
      "type": "message_creation",
9
      "status": "completed",
      "cancelled_at": null,
10
11
      "completed_at": 1699063291,
12
      "expired_at": null,
13
      "failed_at": null,
14
      "last_error": null,
15
      "step_details": {
16
       "type": "message_creation",
17
       "message_creation": {
18
         "message_id": "msg_abc123"
19
20
      },
      "usage": {
21
22
       "prompt_tokens": 123,
23
       "completion_tokens": 456,
24
       "total_tokens": 579
25
      }
26 }
```

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

#### completed\_at integer or null

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

#### metadata map

Set of 16 key-value pairs that can be attached to an object. This can be useful for storing additional information about the object in a structured format. Keys can be a maximum of 64 characters long and values can be a maximum of 512 characters long.

#### usage object or null

Usage statistics related to the run step. This value will be null while the run step's status is in\_progress .

∨Show properties