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

`speed` number Optional
Defaults to 1
The speed of the generated audio. Select a value from `0.25` to `4.0`. `1.0` is the default.

`stream_format` string
Optional Defaults to audio
The format to stream the audio in. Supported formats are `sse` and `audio`. `sse` is not supported for `tts-1` or `tts-1-hd`.

Returns

The audio file content or a stream of audio events.

Create transcription

`POST` `https://api.openai.com/v1/audio/transcriptions`

Default | Streaming | Logprobs | Word timestamps

Example request javascript

riptions

Transcribes audio into the input language.

Request body

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

model string Required
ID of the model to use. The options are `gpt-4o-transcribe`, `gpt-4o-mini-transcribe`, and `whisper-1` (which is powered by our open source Whisper V2 model).

chunking_strategy
"auto" or object Optional
Controls how the audio is cut into chunks. When set to `"auto"`, the server first normalizes loudness and then uses voice activity detection (VAD) to choose boundaries. `server_vad` object can be provided to tweak VAD detection parameters manually. If unset, the audio is transcribed as a single block.

Show possible types

Example request javascript

```javascript
import OpenAI from "openai";

const openai = new OpenAI();

async function main() {
  const transcription = await openai.audio
    file: fs.createReadStream("audio.mp3")
    model: "gpt-4o-transcribe",
    response_format: "json",
    include: ["logprobs"]
  });

  console.log(transcription);
}
main();
```

Response

```
{
  "text": "Hey, my knee is hurting and I w
  "logprobs": [
    { "token": "Hey", "logprob": -1.041529
    { "token": ",", "logprob": -9.805982e-
    { "token": " my", "logprob": -0.002297
    {
      "token": " knee",
      "logprob": -4.7159858e-5,
      "bytes": [32, 107, 110, 101, 101]
    },
    { "token": " is", "logprob": -0.043909
    {
      "token": " hurting",
```

include[] array Optional
Additional information to include in the transcription response. `logprobs` will return the log probabilities of the tokens in the response to understand the model's confidence in the transcription. `logprobs` only works with response_format set to `json` and only with the models `gpt-4o-transcribe` and `gpt-4o-mini-transcribe`.

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

prompt string Optional
An optional text to guide the model's style or continue a previous audio segment. The prompt should match the audio language.

response_format string
Optional Defaults to json
The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, or `vtt`. For `gpt-4o-transcribe` and `gpt-4o-mini-transcribe`, the only supported format is `json`.

stream boolean or null Optional
Defaults to false

If set to true, the model response data will be streamed to the client as it is generated using server-sent events. See the Streaming section of the Speech-to-Text guide for more information.

Note: Streaming is not supported for the `whisper-1` model and will be ignored.

temperature number Optional
Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. If set to 0, the model will use log probability to automatically increase the temperature until certain thresholds are hit.

timestamp_granularities[]
array Optional
Defaults to segment

The timestamp granularities to populate for this transcription. `response_format` must be set `verbose_json` to use timestamp granularities. Either or both of these options are supported: `word`, or `segment`. Note: There is no additional

latency for segment timestamps, but generating word timestamps incurs additional latency.

Returns

The transcription object, a verbose transcription object or a stream of transcript events.

Create translation

`POST https://api.openai.com/v1/audio/translations`

Translates audio into English.

Request body

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

`model` string or "whisper-1" Required
ID of the model to use. Only `whisper-1` (which is powered by our open source Whisper V2

Example request javascript

```javascript
import fs from "fs";
import OpenAI from "openai";

const openai = new OpenAI();

async function main() {
    const translation = await openai.audio
        file: fs.createReadStream("speech.
        model: "whisper-1",
    });

    console.log(translation.text);
}
main();
```

Response

```
{
  "text": "Hello, my name is Wolfgang and I
```

model) is currently available.

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

response_format string Optional Defaults to json
The format of the output, in one of these options: `json`, `text`, `srt`, `verbose_json`, or `vtt`.

temperature number Optional Defaults to 0
The sampling temperature, between 0 and 1. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. If set to 0, the model will use log probability to automatically increase the temperature until certain thresholds are hit.

```
3 }
```

Returns

The translated text.

The transcription object (JSON)

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

logprobs array

The log probabilities of the tokens in the transcription. Only returned with the models `gpt-4o-transcribe` and `gpt-4o-mini-transcribe` if `logprobs` is added to the `include` array.

Show properties

text string

The transcribed text.

usage object

Token usage statistics for the request.

Show possible types

OBJECT The transcription object (JSON)


```
{
  "text": "Imagine the wildest idea that y
  "usage": {
    "type": "tokens",
    "input_tokens": 14,
    "input_token_details": {
      "text_tokens": 10,
      "audio_tokens": 4
    },
    "output_tokens": 101,
    "total_tokens": 115
  }
}
```


The transcription object (Verbose JSON)

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

duration number
The duration of the input audio.

language string
The language of the input audio.

segments array
Segments of the transcribed text and their corresponding details.

Show properties

text string
The transcribed text.

usage object
Usage statistics for models billed by audio input duration.

Show properties

words array
Extracted words and their corresponding timestamps.

Show properties

OBJECT The transcription object (Verbose JS…


```
{
  "task": "transcribe",
  "language": "english",
  "duration": 8.470000267028809,
  "text": "The beach was a popular spot on
  "segments": [
    {
      "id": 0,
      "seek": 0,
      "start": 0.0,
      "end": 3.319999933242798,
      "text": " The beach was a popular sp
      "tokens": [
        50364, 440, 7534, 390, 257, 3743,
      ],
      "temperature": 0.0,
      "avg_logprob": -0.2860786020755768,
      "compression_ratio": 1.2363636493682
      "no_speech_prob": 0.0098597947508096
    },
    ...
  ],
  "usage": {
    "type": "duration",
    "seconds": 9
  }
}
```


Stream Event (speech.audio.delta)

Emitted for each chunk of audio data generated during speech synthesis.

audio string
A chunk of Base64-encoded audio data.

type string
The type of the event. Always `speech.audio.delta`.

OBJECT Stream Event (speech.audio.delta)

```
{
  "type": "speech.audio.delta",
  "audio": "base64-encoded-audio-data"
}
```

Stream Event (speech.audio.done)

Emitted when the speech synthesis is complete and all audio has been streamed.

type string
The type of the event. Always `speech.audio.done`.

usage object
Token usage statistics for the request.

Show properties

OBJECT Stream Event (speech.audio.done)

```
{
  "type": "speech.audio.done",
  "usage": {
    "input_tokens": 14,
    "output_tokens": 101,
    "total_tokens": 115
  }
}
```

Stream Event (transcript.text.delta)

Emitted when there is an additional text delta. This is also the first event emitted when the transcription starts. Only emitted when you create a transcription with the `Stream` parameter set to `true`.

`delta` string
The text delta that was additionally transcribed.

`logprobs` array
The log probabilities of the delta. Only included if you create a transcription with the `include[]` parameter set to `logprobs`.

Show properties

`type` string
The type of the event. Always `transcript.text.delta`.

OBJECT Stream Event (transcript.text.delta)

```
{
  "type": "transcript.text.delta",
  "delta": " wonderful"
}
```

Stream Event (transcript.text.done)

Emitted when the transcription is complete. Contains the complete transcription text. Only emitted when you create a transcription with the `Stream` parameter set to `true`.

logprobs array
The log probabilities of the individual tokens in the transcription. Only included if you create a transcription with the `include[]` parameter set to `logprobs`.

Show properties

text string
The text that was transcribed.

type string
The type of the event. Always `transcript.text.done`.

usage object
Usage statistics for models billed by token usage.

Show properties

OBJECT Stream Event (transcript.text.done)

```
{
  "type": "transcript.text.done",
  "text": "I see skies of blue and clouds
  "usage": {
    "type": "tokens",
    "input_tokens": 14,
    "input_token_details": {
      "text_tokens": 10,
      "audio_tokens": 4
    },
    "output_tokens": 31,
    "total_tokens": 45
  }
}
```

Images