partial_images integer Optional Defaults to 0

The number of partial images to generate. This parameter is used for streaming responses that return partial images. Value must be between 0 and 3. When set to 0, the response will be a single image sent in one streaming event.

Note that the final image may be sent before the full number of partial images are generated if the full image is generated more quickly.

quality string or null Optional Defaults to auto

The quality of the image that will be generated.

- `auto` (default value) will automatically select the best quality for the given model.
- `high` , `medium` and `low` are supported for the GPT image models.
- `hd` and `standard` are supported for `dall-e-3` .
- `standard` is the only option for `dall-e-2` .

response_format string or null Optional Defaults to url

The format in which generated images with `dall-e-2` and `dall-e-3` are returned. Must be one of `url` or `b64_json` . URLs are only valid for 60 minutes after the image has been generated. This parameter isn't supported for the GPT image models, which always return base64-encoded images.

size string or null Optional Defaults to auto

The size of the generated images. Must be one of `1024x1024` , `1536x1024` (landscape), `1024x1536` (portrait), or `auto` (default value) for the GPT image models, one of `256x256` , `512x512` , or `1024x1024` for `dall-e-2` , and one of `1024x1024` , `1792x1024` , or `1024x1792` for `dall-e-3` .

stream boolean or null Optional Defaults to false

Generate the image in streaming mode. Defaults to `false` . See the Image generation guide for more information. This parameter is only supported for the GPT image models.

style string or null Optional Defaults to vivid

The style of the generated images. This parameter is only supported for `dall-e-3` . Must be one of `vivid` or `natural` . Vivid causes the model to lean towards generating hyper-real and

---

[Generate image] [Streaming]

Example request   node.js ⌄ ⧉

```node.js
1  import OpenAI from "openai";
2  import { writeFile } from "fs/promises";
3
4  const client = new OpenAI();
5
6  const img = await client.images.generate({
7    model: "gpt-image-1.5",
8    prompt: "A cute baby sea otter",
9    n: 1,
10   size: "1024x1024"
11 });
12
13 const imageBuffer = Buffer.from(img.data[0].b64_js
14 await writeFile("output.png", imageBuffer);
```

Response ⧉

```json
1  {
2    "created": 1713833628,
3    "data": [
4      {
5        "b64_json": "..."
6      }
7    ],
8    "usage": {
9      "total_tokens": 100,
10     "input_tokens": 50,
11     "output_tokens": 50,
12     "input_tokens_details": {
13       "text_tokens": 10,
14       "image_tokens": 40
15     }
16   }
17 }
```