Text to speech

Copy page

Learn how to turn text into lifelike spoken audio.

Overview

The Audio API provides a `speech` endpoint based on our TTS (text-to-speech) model. It comes with six built-in voices and can be used to:

- Narrate a written blog post
- Produce spoken audio in multiple languages
- Give realtime audio output using streaming

Here's an example of the `alloy` voice:

Our usage policies require you to provide a clear disclosure to end users that the TTS voice they are hearing is AI-generated and not a human voice.

Quickstart

The `speech` endpoint takes three key inputs: 1) model, 2) the text to be turned into audio, and 3) the voice you want to use in the output. Here's a simple request example:

Generate spoken audio from input text

python

```python
from pathlib import Path
from openai import OpenAI

client = OpenAI()
speech_file_path = Path(__file__).parent / "speech.mp3"
response = client.audio.speech.create(
    model="tts-1",
    voice="alloy",
    input="Today is a wonderful day to build something people love!",
)
response.stream_to_file(speech_file_path)
```

By default, the endpoint outputs an MP3 of the spoken audio, but you can configure it to output any supported format.

Audio quality

For realtime applications, the standard `tts-1` model provides the lowest latency, but at a lower quality than the `tts-1-hd` model. Because of the way the models generate the audio, `tts-1` may generate content with more static in certain situations. The audio may not have noticeable differences—it depends on the individual person listening and the listening device.

Voice options

Experiment with different voices (`alloy` , `ash` , `coral` , `echo` , `fable` , `onyx` , `nova` , `sage` , `shimmer`) to find a match for your desired tone and audience. Current voices are optimized for English.

Alloy

Ash

Coral

Echo

Fable

Onyx

Nova

Sage

Shimmer

Streaming realtime audio

The Speech API provides support for realtime audio streaming using chunk transfer encoding. This means the audio can be played before the full file is generated and made accessible.

```python
from openai import OpenAI

client = OpenAI()

response = client.audio.speech.create(
    model="tts-1",
    voice="alloy",
    input="Hello world! This is a streaming test.",
)

response.stream_to_file("output.mp3")
```

Supported output formats

The default response format is `mp3`, but other formats like `opus` and `wav` are available.

MP3: The default response format for general use cases.

Opus: For internet streaming and communication, low latency.

AAC: For digital audio compression, preferred by YouTube, Android, iOS.

FLAC: For lossless audio compression, favored by audio enthusiasts for archiving.

WAV: Uncompressed WAV audio, suitable for low-latency applications to avoid decoding overhead.

PCM: Similar to WAV but contains the raw samples in 24kHz (16-bit signed, low-endian), without the header.

Supported languages

The TTS model generally follows the Whisper model in terms of language support. Whisper supports the following languages and performs well, despite voices being optimized for English:

Afrikaans, Arabic, Armenian, Azerbaijani, Belarusian, Bosnian, Bulgarian, Catalan, Chinese, Croatian, Czech, Danish, Dutch, English, Estonian, Finnish, French, Galician, German, Greek, Hebrew, Hindi, Hungarian, Icelandic, Indonesian, Italian, Japanese, Kannada, Kazakh, Korean, Latvian, Lithuanian, Macedonian, Malay, Marathi, Maori, Nepali, Norwegian, Persian, Polish, Portuguese, Romanian, Russian, Serbian, Slovak, Slovenian, Spanish, Swahili, Swedish, Tagalog, Tamil, Thai, Turkish, Ukrainian, Urdu, Vietnamese, and Welsh.

You can generate spoken audio in these languages by providing input text in the language of your choice.

FAQ

How can I control the emotional range of the generated audio?

There's no direct mechanism to control the emotional output of the audio generated. Certain factors may influence the output audio, like capitalization or grammar, but our internal tests with these have yielded mixed results.

Can I create a custom copy of my own voice?

No, this is not something we support.

Do I own the outputted audio files?

Yes, like with all outputs from our API, the person who created them owns the output. You are still required to inform end users that they are hearing audio generated by AI and not a real person talking to them.