# Text-to-speech entity

A text-to-speech (TTS) entity enables Home Assistant to speak to you.

A text-to-speech entity is derived from the `homeassistant.components.tts.TextToSpeechEntity`.



## Properties #

**TIP**

Properties should always only return information from memory and not do I/O (like network requests).

| Name | Type | Default | Description |
|---|---|---|---|
| supported_languages | list[str] | **Required** | The supported languages of the TTS service. |
| default_language | str | **Required** | The default language of the TTS service. |
| supported_options | list[str] | None | The supported options like voice, emotions of the TTS service. |
| default_options | Mapping[str, Any] | None | The default options of the TTS service. |

## Methods

### Get supported voices

This method is used to return a list of supported voices for a language of a TTS service.

```
class MyTextToSpeechEntity(TextToSpeechEntity):
    """Represent a Text To Speech entity."""

    @callback
    def async_get_supported_voices(self, language: str) -> list[str] | None:
        """Return a list of supported voices for a language."""
```

### Generating TTS audio in 1-shot

This method takes a message and language as input and returns the TTS audio. It can be implemented as either synchronous or asynchronous and is mandatory to implement.

```
class MyTextToSpeechEntity(TextToSpeechEntity):
    """Represent a Text To Speech entity."""

    def get_tts_audio(
        self, message: str, language: str, options: dict[str, Any]
    ) -> TtsAudioType:
        """Load tts audio file from the engine."""

    async def async_get_tts_audio(
        self, message: str, language: str, options: dict[str, Any]
    ) -> TtsAudioType:
        """Load tts audio file from the engine."""
```

### Generating TTS audio with message streaming in

Large language models generate text in chunks. The TTS service can be called with a stream of text messages, and the TTS service will return the audio in chunks.

This method is optional. When not implemented, the TTS service will call the 1-shot method with the final message.

```
class MyTextToSpeechEntity(TextToSpeechEntity):
    """Represent a Text To Speech entity."""

    async def async_stream_tts_audio(
        self, request: TTSAudioRequest
    ) -> TTSAudioResponse:
        """Generate speech from an incoming message."""
```

The definition of the `TTSAudioRequest` and `TTSAudioResponse` objects are as follows:

```
@dataclass
class TTSAudioRequest:
    """Request to get TTS audio."""

    language: str
    options: dict[str, Any]
    message_gen: AsyncGenerator[str]


@dataclass
class TTSAudioResponse:
    """Response containing TTS audio stream."""

    extension: str
    data_gen: AsyncGenerator[bytes]
```

Edit this page

*Last updated on **May 19, 2025***

Previous « To-do list

Next Update »