**Improve text-to-speech by using Speech Synthesis Markup Language**

Speech Synthesis Markup Language (SSML) is an **XML-based markup language** that you can use to fine-tune your text to speech output attributes such as pitch, pronunciation, speaking rate, volume, and more. It gives you more control and flexibility than plain text input.

**Use case scenarios**

SSML is designed to give you flexibility in how you want your speech output to sound, and it provides different properties for how you can customize that output. You can use SSML to:

* [Define the input text structure](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-structure) that determines the structure, content, and other characteristics of your text to speech output. For example, you can use SSML to define a paragraph, a sentence, a break or a pause, or silence. You can wrap text with event tags, like a bookmark or viseme, that your application can process later. A viseme is the visual description of a phoneme, the individual speech sounds, in spoken language.
* [Choose the voice](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-voice), language, name, style, and role. You can use multiple voices in a single SSML document. You can also adjust the emphasis, speaking rate, pitch, and volume. SSML can also insert prerecorded audio, such as a sound effect or a musical note.
* [Control pronunciation](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-pronunciation) of the output audio. For example, you can use SSML with phonemes and a custom lexicon to improve pronunciation. You can also use SSML to define how a word or mathematical expression is pronounced.

You're billed for each character that's converted to speech, including punctuation. Although the SSML document itself isn't billable, the service counts optional elements that you use to adjust how the text is converted to speech, like phonemes and pitch, as billable characters.

**Ways to work with SSML**

SSML functionality is available in various tools that might fit your use case.

* [The audio content creation](https://aka.ms/audiocontentcreation) tool lets you author plain text and SSML in Speech Studio. You can listen to the output audio and adjust the SSML to improve speech synthesis. For more information, see [Speech synthesis with the Audio Content Creation tool](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/how-to-audio-content-creation).
* [The batch synthesis API](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/batch-synthesis) accepts SSML via the inputs property.
* [The Speech CLI](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/get-started-text-to-speech?pivots=programming-language-cli) accepts SSML via the spx synthesize --ssml SSML command line argument.
* [The Speech SDK](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/how-to-speech-synthesis#use-ssml-to-customize-speech-characteristics) accepts SSML via the "speak" SSML method across the different supported languages.

**Document structure**

Each SSML document is created with SSML elements or tags. These elements are used to adjust the voice, style, pitch, prosody, volume, and more.



Some examples of contents that are allowed in each element are described in the following list:

* **audio**: The body of the audio element can contain plain text or SSML markup that's spoken if the audio file is unavailable or unplayable. The audio element can also contain text and the following elements: audio, break, p, s, phoneme, prosody, say-as, and sub.
* **bookmark**: This element can't contain text or any other elements.
* **break**: This element can't contain text or any other elements.
* **emphasis**: This element can contain text and the following elements: audio, break, emphasis, lang, phoneme, prosody, say-as, and sub.
* **lang**: This element can contain all other elements except mstts:backgroundaudio, voice, and speak.
* **lexicon**: This element can't contain text or any other elements.
* **math**: This element can only contain text and MathML elements.
* **mstts:audioduration**: This element can't contain text or any other elements.
* **mstts:backgroundaudio**: This element can't contain text or any other elements.
* **mstts:embedding:** This element can contain text and the following elements: audio, break, emphasis, lang, phoneme, prosody, say-as, and sub.
* **mstts:express-as:** This element can contain text and the following elements: audio, break, emphasis, lang, phoneme, prosody, say-as, and sub.
* **mstts:silence**: This element can't contain text or any other elements.
* **mstts:viseme**: This element can't contain text or any other elements.
* **p**: This element can contain text and the following elements: audio, break, phoneme, prosody, say-as, sub, mstts:express-as, and s.
* **phoneme**: This element can only contain text and no other elements.
* **prosody**: This element can contain text and the following elements: audio, break, p, phoneme, prosody, say-as, sub, and s.
* **s**: This element can contain text and the following elements: audio, break, phoneme, prosody, say-as, mstts:express-as, and sub.
* **say-as**: This element can only contain text and no other elements.
* **sub**: This element can only contain text and no other elements.
* **speak**: The root element of an SSML document. This element can contain the following elements: mstts:backgroundaudio and voice.
* **voice**: This element can contain all other elements except mstts:backgroundaudio and speak.

**Special characters**

* To use the characters **&,** **<**, and **>** within the SSML element's value or text, you must use the entity format. Specifically, you must use **&amp;** in place of **&**, use **&lt;** in place of **<**, and use **&gt;** in place of **>**. Otherwise, the SSML isn't parsed correctly.
* For example, specify **green &amp; yellow** instead of **green & yellow**. The following SSML is parsed as expected:

A screenshot of a computer code

Description automatically generated

Double or single quotation marks must enclose the attribute values. For example**, <prosody volume="90">** and **<prosody volume='90'>** are well-formed, valid elements, but **<prosody volume=90>** isn't recognized.

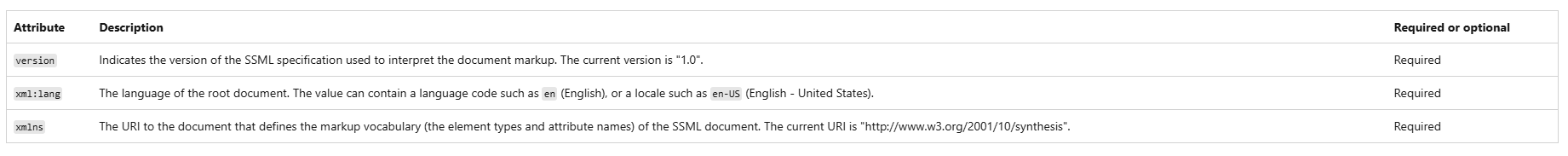
**Speak root element**

The **speak** element contains information such as version, language, and the markup vocabulary definition. The speak element is the root element that's required for all SSML documents. You must specify the default language within the **speak** element, whether or not the language is adjusted elsewhere such as within the [lang](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-voice#adjust-speaking-languages) element.

Here's the syntax for the speak element:

A close-up of a computer

Description automatically generated



The speak element must contain at least one [voice element](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-voice#use-voice-elements).

**Single voice example**

This example uses the **en-US-AvaNeural** voice.

A screenshot of a computer

Description automatically generated

**Add a break**

Use the **break** element to override the default behavior of breaks or pauses between words. Otherwise, the Speech service automatically inserts pauses.

Usage of the **break** element's attributes are described in the following table.

A white background with black text

Description automatically generated

Here are more details about the strength attribute.

A screenshot of a phone

Description automatically generated

**Break examples**

The following three ways all add 750 ms breaks.

A screenshot of a computer

Description automatically generated

**Add silence**

Use the **mstts:silence** element to insert pauses before or after text, or between two adjacent sentences.

One of the differences between **mstts:silence** and **break** is that a **break** element can be inserted anywhere in the text. Silence only works at the beginning or end of input text or at the boundary of two adjacent sentences.

The silence setting is applied to all input text within its enclosing **voice** element. To reset or change the silence setting again, you must use a new **voice** element with either the same voice or a different voice.

Usage of the **mstts:silence** element's attributes are described in the following table.

A close-up of a white background

Description automatically generated

**mstts silence examples**

In this example, mstts:silence is used to add 200 ms of silence between two sentences.

A screenshot of a computer

Description automatically generated

**Specify paragraphs and sentences**

The **p** and **s** elements are used to denote paragraphs and sentences, respectively. In the absence of these elements, the Speech service automatically determines the structure of the SSML document.

**Paragraph and sentence examples**

The following example defines two paragraphs that each contain sentences. In the second paragraph, the Speech service automatically determines the sentence structure, since they aren't defined in the SSML document.

A screenshot of a computer

Description automatically generated

**Bookmark element**

You can use the **bookmark** element in SSML to reference a specific location in the text or tag sequence. Then you use the Speech SDK and subscribe to the **BookmarkReached** event to get the offset of each marker in the audio stream. The **bookmark** element isn't spoken.



**Bookmark examples**

As an example, you might want to know the time offset of each flower word in the following snippet:

A screenshot of a computer code

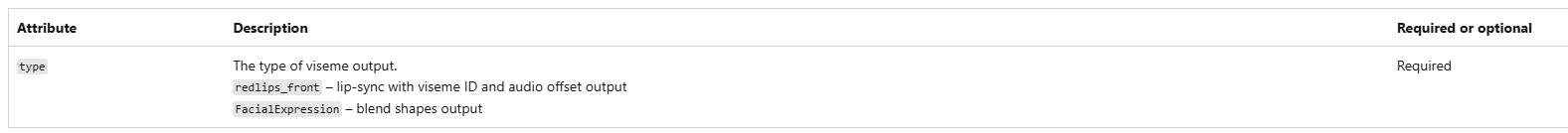
Description automatically generated

**Viseme element**

A **viseme** is the visual description of a phoneme in spoken language. It defines the position of the face and mouth while a person is speaking. You can use the **mstts:viseme**element in SSML to request viseme output.

The viseme setting is applied to all input text within its enclosing **voice** element. To reset or change the viseme setting again, you must use a new **voice** element with either the same voice or a different voice.

Usage of the **viseme** element's attributes are described in the following table.



**Viseme examples**

This SSML snippet illustrates how to request blend shapes with your synthesized speech.

A screenshot of a computer

Description automatically generated

**Customize voice and sound with SSML**

 You can also use multiple voices in a single SSML document, and adjust the emphasis, speaking rate, pitch, and volume. In addition, SSML features the ability to insert prerecorded audio, such as a sound effect or a musical note.

**Use voice elements**

At least one voice element must be specified within each SSML [speak](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-structure#speak-root-element) element. This element determines the voice that's used for text to speech.

You can include multiple voice elements in a single SSML document. Each voice element can specify a different voice. You can also use the same voice multiple times with different settings, such as when you [change the silence duration](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-structure#add-silence) between sentences.

The following table describes the usage of the voice element's attributes:

A screenshot of a computer

Description automatically generated

**Single voice example**

This example uses the en-US-AvaMultilingualNeural voice.

A screenshot of a computer code

Description automatically generated

**Multiple voices example**

Within the speak element, you can specify multiple voices for text to speech output. These voices can be in different languages. For each voice, the text must be wrapped in a voice element.

This example alternates between the en-US-AvaMultilingualNeural and en-US-AndrewMultilingualNeural voices. The neural multilingual voices can speak different languages based on the input text.

A screenshot of a computer code

Description automatically generated

**Custom neural voice example**

This example uses a custom voice named **my-custom-voice**.

A screenshot of a computer

Description automatically generated

**Audio effect example**

You use the effect attribute to optimize the auditory experience for scenarios such as cars and telecommunications. The following SSML example uses the effect attribute with the configuration in car scenarios.

A screenshot of a computer code

Description automatically generated

**Use speaking styles and roles**

By default, neural voices have a neutral speaking style. You can adjust the speaking style, style degree, and role at the sentence level.

The following table describes the usage of the **mstts:express-as** element's attributes:

A close up of text

Description automatically generated

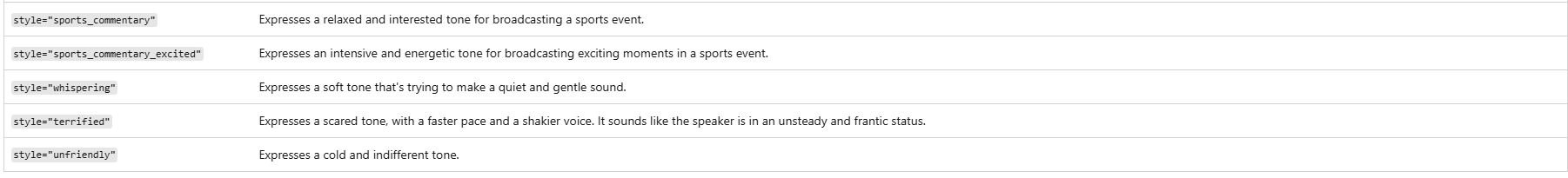
The following table describes each supported **style** attribute:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



The following table has descriptions of each supported **role** attribute:

A screenshot of a computer

Description automatically generated

**Style and degree example**

You use the mstts:express-as element to express emotions like cheerfulness, empathy, and calm. You can also optimize the voice for different scenarios like customer service, newscast, and voice assistant.

The following SSML example uses the <mstts:express-as> element with a sad style degree of 2.

A screenshot of a computer program

Description automatically generated

**Role example**

Apart from adjusting the speaking styles and style degree, you can also adjust the role parameter so that the voice imitates a different age and gender. For example, a male voice can raise the pitch and change the intonation to imitate a female voice, but the voice name isn't changed.

This SSML snippet illustrates how the role attribute is used to change the role-play for**zh-CN-XiaomoNeural**.

A screenshot of a computer

Description automatically generated

**Custom neural voice style example**

You can train your custom neural voice to speak with some preset styles such as cheerful, sad, and whispering. You can also [train a custom neural voice](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/professional-voice-train-voice?tabs=multistyle#train-your-custom-neural-voice-model) to speak in a custom style as determined by your training data. To use your custom neural voice style in SSML, specify the style name that you previously entered in Speech Studio.

This example uses a custom voice named **my-custom-voice**. The custom voice speaks with the cheerful preset style and style degree of 2, and then with a custom style named **my-custom-style** and style degree of 0.01.

A screenshot of a computer code

Description automatically generated

**Speaker profile ID**

You use the **mstts:ttsembedding** element to specify the **speakerProfileId** property for a [personal voice](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/personal-voice-overview). Personal voice is a custom neural voice that's trained on your own voice or your customer's voice. For more information, see [create a personal voice](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/personal-voice-create-voice).

The following SSML example uses the <mstts:ttsembedding> element with a voice name and speaker profile ID.

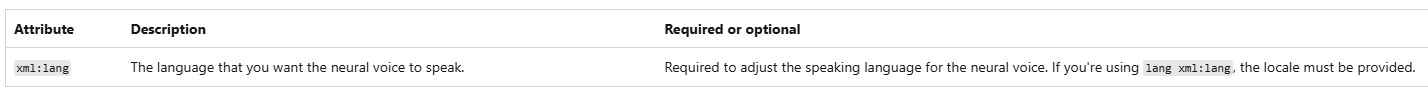
A screenshot of a computer

Description automatically generated

**Adjust speaking languages**

By default, multilingual voices can autodetect the language of the input text and speak in the language of the default locale of the input text without using SSML. Optionally, you can use the <lang xml:lang> element to adjust the speaking language for these voices to set the preferred accent such as en-GB for British English. You can adjust the speaking language at both the sentence level and word level. For information about the supported languages for multilingual voice, see [Multilingual voices with the lang element](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-synthesis-markup-voice#multilingual-voices-with-the-lang-element) for a table showing the <lang> syntax and attribute definitions.

The following table describes the usage of the **<lang xml:lang>** element's attributes:



**Note** : The <lang xml:lang> element is incompatible with the prosody and break elements. You can't adjust pause and prosody like pitch, contour, rate, or volume in this element. Non-multilingual voices don't support the <lang xml:lang> element by design.

**Multilingual voices with the lang element**

Use the [multilingual voices section](https://learn.microsoft.com/en-us/azure/ai-services/speech-service/language-support?tabs=tts#multilingual-voices) to determine which speaking languages the Speech service supports for each neural voice, as demonstrated in the following example table. If the voice doesn't speak the language of the input text, the Speech service doesn't output synthesized audio.

A close-up of a document

Description automatically generated

**Lang examples**

You must specify en-US as the default language within the speak element, whether or not the language is adjusted elsewhere. In this example, the primary language for en-US-AvaMultilingualNeural is en-US.

This SSML snippet shows how to use <lang xml:lang> to speak de-DE with the en-US-AvaMultilingualNeural neural voice.



Within the speak element, you can specify multiple languages including en-US for text to speech output. For each adjusted language, the text must match the language and be wrapped in a voice element. This SSML snippet shows how to use <lang xml:lang> to change the speaking languages to es-MX, en-US, and fr-FR.

A screenshot of a computer

Description automatically generated

**Adjust prosody**

You can use the prosody element to specify changes to pitch, contour, range, rate, and volume for the text to speech output. The prosody element can contain text and the following elements: audio, break, p, phoneme, prosody, say-as, sub, and s.

Because prosodic attribute values can vary over a wide range, the speech recognizer interprets the assigned values as a suggestion of what the actual prosodic values of the selected voice should be. Text to speech limits or substitutes values that aren't supported. Examples of unsupported values are a pitch of 1 MHz or a volume of 120.

The following table describes the usage of the prosody element's attributes:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Change speaking rate example**

This SSML snippet illustrates how the rate attribute is used to change the speaking rate to 30% greater than the default rate.

A screenshot of a computer code

Description automatically generated

**Change volume example**

This SSML snippet illustrates how the volume attribute is used to change the volume to 20% greater than the default volume.

A screenshot of a computer code

Description automatically generated

**Change pitch example**

This SSML snippet illustrates how the pitch attribute is used so that the voice speaks in a high pitch.

A screen shot of a computer code

Description automatically generated

**Change pitch contour example**

This SSML snippet illustrates how the contour attribute is used to change the contour.

A screenshot of a computer code

Description automatically generated

**Adjust emphasis**

You can use the optional emphasis element to add or remove word-level stress for the text. This element can only contain text and the following elements: audio, break, emphasis, lang, phoneme, prosody, say-as, sub, and voice.

A white background with black text

Description automatically generated

**Emphasis examples**

This SSML snippet demonstrates how you can use the emphasis element to add moderate level emphasis for the word "meetings."

A screenshot of a computer

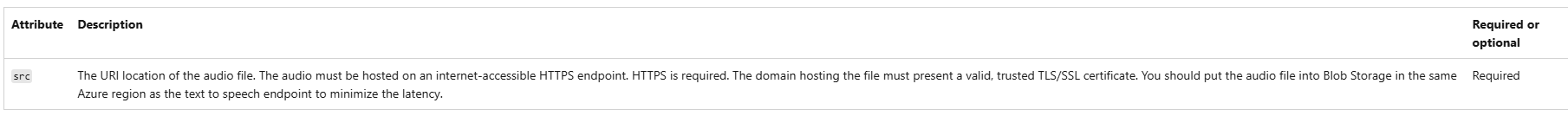
Description automatically generated

**Add recorded audio**

The audio element is optional. You can use it to insert prerecorded audio into an SSML document. The body of the audio element can contain plain text or SSML markup spoken if the audio file is unavailable or unplayable. The audio element can also contain text and the following elements: audio, break, p, s, phoneme, prosody, say-as, and sub.

Any audio included in the SSML document must meet these requirements:

* The audio file must be valid \**.mp3*, \**.wav*, \**.opus*, \**.ogg*, \**.flac*, or \**.wma* files.
* The combined total time for all text and audio files in a single response can't exceed 600 seconds.
* The audio must not contain any customer-specific or other sensitive information.



**Audio examples**

This SSML snippet illustrates how to use src attribute to insert audio from two .wav files.

A screenshot of a computer code

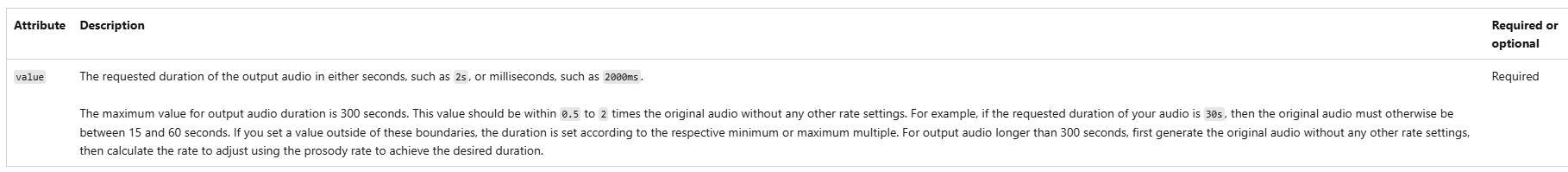
Description automatically generated

**Adjust the audio duration**

Use the mstts:audioduration element to set the duration of the output audio. Use this element to help synchronize the timing of audio output completion. The audio duration can be decreased or increased between 0.5 to 2 times the rate of the original audio. The original audio is the audio without any other rate settings. The speaking rate is slowed down or sped up accordingly based on the set value.

The audio duration setting applies to all input text within its enclosing voice element. To reset or change the audio duration setting again, you must use a new voice element with either the same voice or a different voice.

The following table describes the usage of the mstts:audioduration element's attributes:



**mstts audio duration examples**

In this example, the original audio is around 15 seconds. The mstts:audioduration element is used to set the audio duration to 20 seconds or 20s.

A screenshot of a computer

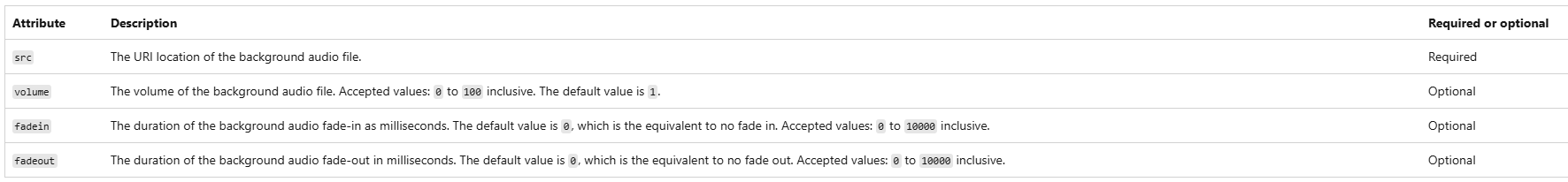
Description automatically generated

**Add background audio**

You can use the **mstts:backgroundaudio** element to add background audio to your SSML documents or mix an audio file with text to speech. With mstts:backgroundaudio, you can loop an audio file in the background, fade in at the beginning of text to speech, and fade out at the end of text to speech.

If the background audio provided is shorter than the text to speech or the fade out, it loops. If it's longer than the text to speech, it stops when the fade out is finished.

Only one background audio file is allowed per SSML document. You can intersperse audio tags within the voice element to add more audio to your SSML document.



**mstss backgroundaudio examples**

