

[AVFAudio](#) / [AVSpeechSynthesizer](#)

## Class

# AVSpeechSynthesizer

An object that produces synthesized speech from text utterances and enables monitoring or controlling of ongoing speech.

iOS 7.0+ | iPadOS 7.0+ | Mac Catalyst 13.1+ | macOS 10.14+ | tvOS | visionOS 1.0+ | watchOS 2.0+

```
class AVSpeechSynthesizer
```

## Overview

To speak some text, create an [AVSpeechUtterance](#) instance that contains the text and pass it to [speak\(\\_:\)](#) on a speech synthesizer instance. You can optionally also retrieve an [AVSpeechSynthesisVoice](#) and set it on the utterance's [voice](#) property to have the speech synthesizer use that voice when speaking the utterance's text.

The speech synthesizer maintains a queue of utterances that it speaks. If the synthesizer isn't speaking, calling [speak\(\\_:\)](#) begins speaking that utterance either immediately or after pausing for its [preUtteranceDelay](#), if necessary. If the synthesizer is speaking, the synthesizer adds utterances to a queue and speaks them in the order it receives them.

After speech begins, you can use the synthesizer object to pause or stop speech. After pausing, you can resume the speech from its paused point or stop the speech entirely and remove all remaining utterances in the queue.

You can monitor the speech synthesizer by examining its [isSpeaking](#) and [isPaused](#) properties, or by setting a delegate that conforms to [AVSpeechSynthesizerDelegate](#). The delegate receives significant events as they occur during speech synthesis.

An `AVSpeechSynthesizer` also controls the route where the speech plays. For more information, see [Directing speech output](#).

### Note

The system doesn't automatically retain the speech synthesizer, so you need to manually retain it until speech concludes.

## Topics

### Controlling speech

```
func speak(AVSpeechUtterance)
```

Adds the utterance you specify to the speech synthesizer's queue.

```
func continueSpeaking() -> Bool
```

Resumes speech from its paused point.

```
func pauseSpeaking(at: AVSpeechBoundary) -> Bool
```

Pauses speech at the boundary you specify.

```
func stopSpeaking(at: AVSpeechBoundary) -> Bool
```

Stops speech at the boundary you specify.

```
enum AVSpeechBoundary
```

Specifies when to pause or stop speech.

### Inspecting a speech synthesizer

```
var isSpeaking: Bool
```

A Boolean value that indicates whether the speech synthesizer is speaking or is in a paused state and has utterances to speak.

```
var isPaused: Bool
```

A Boolean value that indicates whether a speech synthesizer is in a paused state.

### Managing the delegate

```
var delegate: (any AVSpeechSynthesizerDelegate)?
```

The delegate object for the speech synthesizer.

```
protocol AVSpeechSynthesizerDelegate
```

A delegate protocol that contains optional methods you can implement to respond to events that occur during speech synthesis.

## Directing speech output

```
var usesApplicationAudioSession: Bool
```

A Boolean value that specifies whether the app manages the audio session.

```
var mixToTelephonyUplink: Bool
```

A Boolean value that specifies whether to send synthesized speech to an active call.

```
var outputChannels: [AVAudioSessionChannelDescription]?
```

An array of audio session channels to route generated speech.

```
func write(AVSpeechUtterance, toBufferCallback: AVSpeechSynthesizer.
BufferCallback)
```

Generates speech for the utterance and invokes the callback with the audio buffer.

```
typealias BufferCallback
```

A type that defines a callback that receives a buffer of generated speech.

```
func write(AVSpeechUtterance, toBufferCallback: AVSpeechSynthesizer.
BufferCallback, toMarkerCallback: AVSpeechSynthesizer.MarkerCallback)
```

Generates audio buffers and associated metadata for storage or further speech synthesis processing.

```
typealias MarkerCallback
```

A type that defines a callback that receives speech markers.

## Enabling personal voices

```
class var personalVoiceAuthorizationStatus: AVSpeechSynthesizer.
PersonalVoiceAuthorizationStatus
```

Your app's authorization to use personal voices.

```
class let availableVoicesDidChangeNotification: NSNotification.Name
```

A notification that indicates a change in available voices for speech synthesis.

```
class func requestPersonalVoiceAuthorization(completionHandler: (
AVSpeechSynthesizer.PersonalVoiceAuthorizationStatus) -> Void)
```

Prompts the user to authorize your app to use personal voices.

enum PersonalVoiceAuthorizationStatus

An enumeration that models the personal voices authorization status.

---

# Relationships

## Inherits From

NSObject

## Conforms To

- CVarArg
- CustomDebugStringConvertible
- CustomStringConvertible
- Equatable
- Hashable
- NSObjectProtocol