

AVAudioPlayer Class Reference

사운드 콘텐츠 응용

Dept. of Multimedia Science, Sookmyung Women's University, prof. JongWoo Lee

Index

- Overview
- Tasks
 - Initializing a AVAudioPlayer Object
 - Configuring and Controlling Playback
 - Managing Information About a Sound
 - Using Audio Level Metering

Overview

AVAudioPlayer

- providing playback of audio data from a file or memory
- playing sound in any audio format available in iOS

Using an audio player you can:

- Play sounds of any duration
- Play sounds from files or memory buffers
- Loop sounds
- Play multiple sounds simultaneously
- Control relative playback level and stereo positioning for each sound you are playing
- Seek to a particular point in a sound file as fast forward and rewind
- Obtain data using for playback-level metering

Initializing an AVAudioPlayer Object

Initializing a AVAudioPlayer Object

initWithContentsOfURL:error:

Initializing an audio player for playing a designated sound file

```
- (id) initWithContentsOfURL:(NSURL *)url error:(NSError **)outError
```

- Parameters

url	<ul style="list-style-type: none">identifying the sound file to play (The audio data must be in a format supported by Core Audio.)
outError	<ul style="list-style-type: none">Pass in the address of a nil-initialized NSError object.If an error occurs, upon return the <code>NSError</code> object describes the error.<code>NULL</code> : if you do not want error information

- Return Value

On success : initializing `AVAudioPlayer` object

`nil` : the `outError` parameter contains a code that describes the problem

- Availability

Available in iOS 2.2 and later.

Initializing a AVAudioPlayer Object

initWithData:error:

Initializing an audio player for playing a designated memory buffer

```
- (id) initWithData:(NSData *)data error:(NSError **)outError
```

- Parameters

data	<ul style="list-style-type: none">A block of data containing a sound to play. (The audio data must be in a format supported by Core Audio.)
outError	<ul style="list-style-type: none">Pass in the address of a nil-initialized NSError object.If an error occurs, upon return the <code>NSError</code> object describes the error.<code>NULL</code> : if you do not want error information

- Return Value

On success : initializing `AVAudioPlayer` object

`nil` : the `outError` parameter contains a code that describes the problem

- Availability

Available in iOS 2.2 and later.

Configuring and Controlling Playback

Configuring and Controlling Playback

play

Playing a sound asynchronously

– (BOOL)play

- **Return Value**

YES : success

NO : failure

- **Discussion**

Calling this method implicitly calls the `prepareToPlay` method (if the audio player is not already prepared to play.)

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

playAtTime:

Playing a sound asynchronously (starting at a specified point in the audio output device's timeline)

```
- (BOOL)playAtTime:(NSTimeInterval)time
```

- **Parameters**

time	<p>The number of seconds to delay playback, relative to the audio output device's current time.</p> <pre>NSTimeInterval playbackDelay = 3.0; // must be ≥ 0 [myAudioPlayer playAtTime: myAudioPlayer.deviceCurrentTime + playbackDelay];</pre> <p>Important: The value(the time parameter) must be greater than or equal to the value of the audio player's <code>deviceCurrentTime</code> property.</p>
-------------	---

- **Return Value**

YES : success

NO : failure.

- **Availability**

Available in iOS 4.0 and later.

Configuring and Controlling Playback

pause

pauses playback; sound remains ready to resume playback from where it left off.

```
- (void)pause
```

- **Discussion**

Calling `pause` leaves the audio player prepared to play (it does not release the audio hardware that was acquired upon calling `play` or `prepareToPlay`)

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

stop

Stops playback and undoes the setup needed for playback.

```
- (void)stop
```

- **Discussion**

- Calling this method(or allowing a sound to finish playing) undoes the setup performed upon calling the `play` or `prepareToPlay` methods.
- Not reset the value of the `currentTime` property to 0
- If calling `stop` during playback and then calling `play`, playback resumes at the point where it left off.

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

prepareToPlay

Prepares the audio player for playback by preloading its buffers.

```
- (BOOL)prepareToPlay
```

- **Discussion**

- Preloading buffers and acquiring the audio hardware needed for playback
- To undo this setup : call the `stop` method, or allow a sound to finish playing

- **Return Value**

YES : success
NO : failure.

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

playing (read-only)

A Boolean value that indicates whether the audio player is playing (YES) or not (NO).

```
@property(readonly, getter=isplaying) BOOL playing
```

- **Discussion**

- To find out when playback has stopped, use the [audioPlayerDidFinishPlaying:successfully:](#) delegate method.

Important:

Do not poll this property (do not use it inside of a loop) in an attempt to discover when playback has stopped.

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

volume

The playback gain for the audio player, ranging from 0.0 through 1.0.

```
@property float volume
```

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

pan

The audio player's stereo pan position.

```
@property float pan
```

- **Discussion**

By setting this property you can position a sound in the stereo field.

-1.0 full left

0.0 : center

1.0 : full right

- **Availability**

Available in iOS 4.0 and later.

Configuring and Controlling Playback

numberOfLoops

The number of times a sound will return to the beginning, upon reaching the end, to repeat playback.

```
@property NSInteger numberOfLoops
```

- **Discussion**

- 0 : default, playing the sound once.

- positive integer value : times

(For example, 1 : in a total of two plays of the sound)

- any negative integer value : loop the sound until you call the `stop` method

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

delegate

The delegate object for the audio player.

```
@property(assign) id<AVAudioPlayerDelegate> delegate
```

- **Discussion**

To respond to decoding errors, audio interruptions (such as an incoming phone call), and playback completion

- **Availability**

Available in iOS 2.2 and later.

Configuring and Controlling Playback

settings (read-only)

settings dictionary, containing information about the sound associated with the player

```
@property(readonly) NSDictionary *settings
```

- **Discussion**

settings dictionary contains keys for the following information about the player's associated sound:

- Channel layout ([AVChannelLayoutKey](#))
- Encoder bit rate ([AVEncoderBitRateKey](#))
- Audio data format ([AVFormatIDKey](#))
- Channel count ([AVNumberOfChannelsKey](#))
- Sample rate ([AVSampleRateKey](#))

(The settings keys are described in [AV Foundation Audio Settings Constants](#).)

- **Availability**

Available in iOS 4.0 and later.

Managing Information About a Sound

Managing Information About a Sound

numberOfChannels (read-only)

The number of audio channels in the sound associated with the audio player.

```
@property(readonly) NSInteger numberOfChannels
```

- **Availability**

Available in iOS 2.2 and later.

Managing Information About a Sound

duration (read-only)

Returns the total duration, in seconds, of the sound associated with the audio player.

```
@property(readonly) NSTimeInterval duration
```

- **Availability**

Available in iOS 2.2 and later.

Managing Information About a Sound

currentTime

The playback point, in seconds, within the timeline of the sound associated with the audio player.

```
@property NSTimeInterval currentTime
```

- **Discussion**

- If playing, `currentTime` is the offset of the current playback position(measured in seconds from the start of the sound)
- If not playing, `currentTime` is the offset of where playing starts upon calling the `play` method(measured in seconds from the start of the sound)
- For seeking to a specific point in a sound file or implement audio fast-forward and rewind functions

- **Availability**

Available in iOS 2.2 and later.

Managing Information About a Sound

`deviceCurrentTime` (read-only)

The time value, in seconds, of the audio output device.

```
@property(readonly) NSTimeInterval deviceCurrentTime
```

- **Discussion**

- The value increases monotonically. (while playing or paused)
- If more than one audio player is connected to the audio output device, device time continues incrementing as long as at least one of the players is playing or paused.
- If the audio output device has no connected audio players (either playing or paused), device time reverts to 0.
- Use this property to indicate “now” when calling the `playAtTime:` instance method.
- By configuring multiple audio players to play at a specified offset from `deviceCurrentTime`, you can perform precise synchronization.

- **Availability**

Available in iOS 4.0 and later.

Managing Information About a Sound

`url` (read-only)

The URL for the sound associated with the audio player.

```
@property(readonly) NSURL *url
```

- **Discussion**

`nil` : if the audio player was not initialized with a URL.

- **Availability**

Available in iOS 2.2 and later.

Managing Information About a Sound

data (read-only)

The data object containing the sound associated with the audio player.

```
@property(readonly) NSData *data
```

- **Discussion**

`nil` : if the audio player has no data (if it was not initialized with an `NSData` object).

- **Availability**

Available in iOS 2.2 and later.

Using Audio Level Metering

Using Audio Level Metering

meteringEnabled

A Boolean value that indicates the audio-level metering on/off state for the audio player.

```
@property(getter=isMeteringEnabled) BOOL meteringEnabled
```

- **Discussion**

- NO : default
- setting this property to YES, before using metering for an audio player
- ex)

```
[self.player setMeteringEnabled: YES];
```

- **Availability**

Available in iOS 2.2 and later.

Using Audio Level Metering

- averagePowerForChannel:

Returns the average power for a given channel, in decibels, for the sound being played.

```
- (float)averagePowerForChannel:(NSUInteger)channelNumber
```

- **Parameters**

channelNumber	<ul style="list-style-type: none">• The audio channel whose average power value you want to obtain.• Channel numbers : zero-indexed.• A monaural signal, or the left channel of a stereo signal, has 0 (channel number).
----------------------	--

- **Return Value**

- 0 dB : full scale, or maximum power;
- -160 dB : minimum power (near silence).

- **Discussion**

To obtain a current average power value, you must call the [updateMeters](#) method before calling this method.

- **Availability**

Available in iOS 2.2 and later.

Using Audio Level Metering

- peakPowerForChannel:

Returns the peak power for a given channel, in decibels, for the sound being played.

```
- (float)peakPowerForChannel:(NSUInteger)channelNumber
```

- **Parameters**

channelNumber	<ul style="list-style-type: none">• The audio channel whose peak power value you want to obtain.• Channel numbers : zero-indexed.• A monaural signal, or the left channel of a stereo signal, has 0 (channel number).
----------------------	---

- **Return Value**

- 0 dB : full scale, or maximum power
- -160 dB : minimum power (near silence).

- **Discussion**

To obtain a current peak power value, you must call the [updateMeters](#) method before calling this method.

- **Availability**

Available in iOS 2.2 and later.

Using Audio Level Metering

- updateMeters

Refreshes the average and peak power values for all channels of an audio player.

```
- (void)updateMeters
```

- **Discussion**

To obtain current audio power values, you must call this method (before calling [averagePowerForChannel:](#) or [peakPowerForChannel:](#))

- **Availability**

Available in iOS 2.2 and later.