AVAudioPlayer Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/AVFoundation.framework
Availability	Available in iOS 2.2 and later.
Declared in	AVAudioPlayer.h
Related sample code	AddMusic avTouch iPhoneExtAudioFileConvertTest Metronome oalTouch

Overview

An instance of the `AVAudioPlayer` class, called an audio player, provides playback of audio data from a file or memory.

Apple recommends that you use this class for audio playback unless you are playing audio captured from a network stream or require very low I/O latency. For an overview of audio technologies, see *Getting Started with Audio & Video* and "Using Audio" in *Multimedia Programming Guide*.

Using an audio player you can:

- Play sounds of any duration
- Play sounds from files or memory buffers
- Loop sounds
- Play multiple sounds simultaneously, one sound per audio player, with precise synchronization
- Control relative playback level and stereo positioning for each sound you are playing
- Seek to a particular point in a sound file, which supports such application features as fast forward and rewind
- Obtain data you can use for playback-level metering

The `AVAudioPlayer` class lets you play sound in any audio format available in iOS. You implement a delegate to handle interruptions (such as an incoming phone call) and to update the user interface when a sound has finished playing. The delegate methods to use are described in *AVAudioPlayerDelegate Protocol Reference*.

To play, pause, or stop an audio player, call one of its playback control methods, described in "Configuring and Controlling Playback."

This class uses the Objective-C declared properties feature for managing information about a sound—such as the playback point within the sound's timeline, and for accessing playback options—such as volume and looping. You also use a property (`playing`) to test whether or not playback is in

progress.

To configure an appropriate audio session for playback, refer to *AVAudioSession Class Reference* and *AVAudioSessionDelegate Protocol Reference*. To learn how your choice of file formats impacts the simultaneous playback of multiple sounds, refer to "iPhone Hardware and Software Audio Codecs" in *Multimedia Programming Guide*.

Tasks

Initializing an AVAudioPlayer Object

- `initWithContentsOfURL:error:`
- `initWithData:error:`

Configuring and Controlling Playback

- `play`
- `playAtTime:`
- `pause`
- `stop`
- `prepareToPlay`

`playing` *property*
`volume` *property*
`pan` *property*
`numberOfLoops` *property*
`delegate` *property*
`settings` *property*

Managing Information About a Sound

`numberOfChannels` *property*
`duration` *property*
`currentTime` *property*
`deviceCurrentTime` *property*
`url` *property*
`data` *property*

Using Audio Level Metering

`meteringEnabled` *property*

- `averagePowerForChannel:`
- `peakPowerForChannel:`
- `updateMeters`

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

currentTime

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

```
@property NSTimeInterval currentTime
```

Discussion
If the sound is playing, *currentTime* is the offset of the current playback position, measured in seconds from the start of the sound. If the sound is not playing, *currentTime* is the offset of where playing starts upon calling the `play` method, measured in seconds from the start of the sound.

By setting this property you can seek to a specific point in a sound file or implement audio fast-forward and rewind functions.

Availability
Available in iOS 2.2 and later.

See Also
`@property deviceCurrentTime`
`@property duration`

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

data

The data object containing the sound associated with the audio player. (read-only)

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

Discussion
Returns `nil` if the audio player has no data (that is, if it was not initialized with an `NSData` object).

Availability
Available in iOS 2.2 and later.

See Also
`@property url`

Declared In
`AVAudioPlayer.h`

delegate

The delegate object for the audio player.

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

Discussion
The object that you assign to be an audio player's delegate becomes the target of the notifications described in *AVAudioPlayerDelegate Protocol Reference*. These notifications let you respond to decoding errors, audio interruptions (such as an incoming phone call), and playback completion.

Availability

Available in iOS 2.2 and later.

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

deviceCurrentTime

The time value, in seconds, of the audio output device. (read-only)

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

Discussion
The value of this property increases monotonically while an audio player is 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 that are 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—as described in the discussion for that method.

Availability
Available in iOS 4.0 and later.

See Also
`@property currentTime`
– `playAtTime:`

Declared In
`AVAudioPlayer.h`

duration

Returns the total duration, in seconds, of the sound associated with the audio player. (read-only)

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

Availability
Available in iOS 2.2 and later.

See Also
`@property currentTime`

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

meteringEnabled

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

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

Discussion
The default value for the `meteringEnabled` property is off (Boolean `NO`). Before using metering for an audio player, you need to enable it by setting this property to `YES`. If `player` is an audio player instance variable of your controller class, you enable metering as shown here:

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

Availability
Available in iOS 2.2 and later.

See Also
- `averagePowerForChannel:`
- `peakPowerForChannel:`
- `updateMeters`

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

numberOfChannels

The number of audio channels in the sound associated with the audio player. (read-only)

```
@property(readonly) NSUInteger numberOfChannels
```

Availability
Available in iOS 2.2 and later.

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

numberOfLoops

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

```
@property NSInteger numberOfLoops
```

Discussion
A value of `0`, which is the default, means to play the sound once. Set a positive integer value to specify the number of times to return to the start and play again. For example, specifying a value of `1` results in a total of two plays of the sound. Set any negative integer value to loop the sound indefinitely until you call the `stop` method.

Availability
Available in iOS 2.2 and later.

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

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. A value of -1.0 is full left, 0.0 is center, and 1.0 is full right.

Availability
Available in iOS 4.0 and later.

Declared In
`AVAudioPlayer.h`

playing

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

```
@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 (that is, 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.

Related Sample Code
AddMusic
avTouch

Declared In
`AVAudioPlayer.h`

settings

The audio player's settings dictionary, containing information about the sound associated with the player. (read-only)

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

Discussion

An audio player's 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.

Declared In
`AVAudioPlayer.h`

url

The URL for the sound associated with the audio player. (read-only)

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

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

Availability
Available in iOS 2.2 and later.

See Also
`@property data`

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

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.

Related Sample Code
avTouch

Declared In
`AVAudioPlayer.h`

Instance Methods

averagePowerForChannel:

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

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

Parameters

channelNumber

The audio channel whose average power value you want to obtain. Channel numbers are zero-indexed. A monaural signal, or the left channel of a stereo signal, has channel number `0`.

Return Value

A floating-point representation, in decibels, of a given audio channel's current average power. A return value of 0 dB indicates full scale, or maximum power; a return value of -160 dB indicates minimum power (that is, near silence).

If the signal provided to the audio player exceeds ±full scale, then the return value may exceed 0 (that is, it may enter the positive range).

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.

See Also

`@property meteringEnabled`

`peakPowerForChannel:`

Declared In

`AVAudioPlayer.h`

initWithContentsOfURL:error:

Initializes and returns an audio player for playing a designated sound file.

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

Parameters

url

A URL identifying the sound file to play. The audio data must be in a format supported by Core Audio. See "Using Sound in iOS" in *iOS Application Programming Guide*.

outError

Pass in the address of a `nil`-initialized `NSError` object. If an error occurs, upon return the `NSError` object describes the error. If you do not want error information, pass in `NULL`.

Return Value

On success, an initialized `AVAudioPlayer` object. If `nil`, the *outError* parameter contains a code that describes the problem.

Availability

Available in iOS 2.2 and later.

See Also
– `initWithData:error:`

Related Sample Code
AddMusic
avTouch
iPhoneExtAudioFileConvertTest
Metronome
oalTouch

Declared In
`AVAudioPlayer.h`

initWithData:error:

Initializes and returns an audio player for playing a designated memory buffer.

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

Parameters
data
A block of data containing a sound to play. The audio data must be in a format supported by Core Audio. See “Using Sound in iOS” in *iOS Application Programming Guide*.

outError
Pass in the address of a `nil`-initialized `NSError` object. If an error occurs, upon return the `NSError` object describes the error. If you do not want error information, pass in `NULL`.

Return Value
On success, an initialized `AVAudioPlayer` object. If `nil`, the *outError* parameter contains a code that describes the problem.

Availability
Available in iOS 2.2 and later.

See Also
– `initWithContentsOfURL:error:`

Declared In
`AVAudioPlayer.h`

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.

See Also
– `play`
– `prepareToPlay`

– stop

Related Sample Code
avTouch

Declared In
AVAudioPlayer.h

peakPowerForChannel:

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

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

Parameters
channelNumber
The audio channel whose peak power value you want to obtain. Channel numbers are zero-indexed. A monaural signal, or the left channel of a stereo signal, has channel number 0.

Return Value
A floating-point representation, in decibels, of a given audio channel's current peak power. A return value of 0 dB indicates full scale, or maximum power; a return value of -160 dB indicates minimum power (that is, near silence).

If the signal provided to the audio player exceeds ±full scale, then the return value may exceed 0 (that is, it may enter the positive range).

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.

See Also
@property meteringEnabled
– averagePowerForChannel:

Declared In
AVAudioPlayer.h

play

Plays a sound asynchronously.

- (BOOL)play

Return Value
Returns YES on success, or NO on 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.

See Also

- pause
- playAtTime:
- prepareToPlay
- stop

Related Sample Code
AddMusic
AQOfflineRenderTest
iPhoneExtAudioFileConvertTest
Metronome
oalTouch

Declared In
AVAudioPlayer.h

playAtTime:

Plays a sound asynchronously, starting at a specified point in the audio output device's timeline.

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

Parameters
time
The number of seconds to delay playback, relative to the audio output device's current time. For example, to start playback three seconds into the future from the time you call this method, use code like this:

```
NSTimeInterval playbackDelay = 3.0;                 // must be ≥ 0
[myAudioPlayer playAtTime: myAudioPlayer.deviceCurrentTime + playbackDelay];
```

Important: The value that you provide to the *time* parameter must be greater than or equal to the value of the audio player's deviceCurrentTime property.

Return Value
YES on success, or NO on failure.

Discussion
Use this method to precisely synchronize the playback of two or more AVAudioPlayer objects. This code snippet shows the recommended way to do this:

```
// Before calling this method, instantiate two AVAudioPlayer objects and
// assign each of them a sound.

- (void) startSynchronizedPlayback {

    NSTimeInterval shortStartDelay = 0.01;            // seconds
    NSTimeInterval now = player.deviceCurrentTime;

    [player       playAtTime: now + shortStartDelay];
    [secondPlayer playAtTime: now + shortStartDelay];
```

```
    // Here, update state and user interface for each player, as appropriate
}
```

To learn about the virtual audio output device's timeline, read the description for the `deviceCurrentTime` property.

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

Availability
Available in iOS 4.0 and later.

See Also
– `pause`
– `play`
– `prepareToPlay`
– `stop`

Declared In
`AVAudioPlayer.h`

prepareToPlay

Prepares the audio player for playback by preloading its buffers.

```
- (BOOL)prepareToPlay
```

Return Value
Returns `YES` on success, or `NO` on failure.

Discussion
Calling this method preloads buffers and acquires the audio hardware needed for playback, which minimizes the lag between calling the `play` method and the start of sound output.

Calling the `stop` method, or allowing a sound to finish playing, undoes this setup.

Availability
Available in iOS 2.2 and later.

See Also
– `pause`
– `play`
– `stop`

Declared In
`AVAudioPlayer.h`

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.

The `stop` method does not reset the value of the `currentTime` property to `0`. In other words, if you call `stop` during playback and then call `play`, playback resumes at the point where it left off.

Availability
Available in iOS 2.2 and later.

See Also
- `pause`
- `play`
- `prepareToPlay`

Related Sample Code
oalTouch

Declared In
`AVAudioPlayer.h`

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.

See Also
`@property meteringEnabled`

Declared In
`AVAudioPlayer.h`