

---

# AVPlayerItem Class Reference

Audio & Video



2010-05-15



Apple Inc.  
© 2010 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.  
1 Infinite Loop  
Cupertino, CA 95014  
408-996-1010

Apple, the Apple logo, iPhone, Objective-C, and QuickTime are trademarks of Apple Inc., registered in the United States and other countries.

iOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE**

**ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.**

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

**THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.**

**Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

# Contents

## AVPlayerItem Class Reference 5

---

Overview	5
Tasks	6
Creating a Player Item	6
Getting Information About an Item	6
Moving the Playhead	6
Information About Playback	7
Timing Information	7
Settings	7
Properties	7
asset	7
audioMix	8
error	8
forwardPlaybackEndTime	8
loadedTimeRanges	9
playbackBufferEmpty	9
playbackBufferFull	9
playbackLikelyToKeepUp	9
presentationSize	10
reversePlaybackEndTime	10
seekableTimeRanges	10
status	11
timedMetadata	11
tracks	11
videoComposition	11
Class Methods	12
playerItemWithAsset:	12
playerItemWithURL:	12
Instance Methods	13
currentTime	13
initWithAsset:	13
initWithURL:	14
seekToDate:	14
seekToTime:	15
seekToTime:toleranceBefore:toleranceAfter:	15
stepByCount:	16
Constants	16
AVPlayerItemStatus	16
Notifications	17
AVPlayerItemDidPlayToEndTimeNotification	17

**Document Revision History 19**

---

# AVPlayerItem Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSCopying NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AVFoundation.framework
<b>Availability</b>	Available in iOS 4.0 and later.
<b>Declared in</b>	AVPlayerItem.h

## Overview

An `AVPlayerItem` represents the presentation state of an asset that's played by an `AVPlayer` object, and lets you observe that state.

A object carries a reference to an `AVAsset` object and presentation settings for that asset, including track enabled state. If you need to inspect the media assets themselves, you should message the `AVAsset` object itself.

You can initialize a player item using an URL (`playerItemWithURL:` (page 12) and `initWithURL:` (page 14)); the resource types referenced by the URL may include, but aren't necessarily limited to, those with the following corresponding UTIs:

```
kUTTypeQuickTimeMovie, (.mov, .qt)
kUTTypeMPEG4 (.mp4)
@"public.3gpp" (.3gp, .3gpp)
kUTTypeMPEG4Audio (.m4a)
@"com.apple.coreaudio-format" (.caf)
@"com.microsoft.waveform-audio" (.wav)
@"public.aiff-audio" (.aif)
@"public.aifc-audio" (also .aif)
@"org.3gpp.adaptive-multi-rate-audio" (.amr)
```

If you want to play an asset more than once within a sequence of items, you must create independent instances of `AVPlayerItem` for each placement in the player's queue.

## Tasks

### Creating a Player Item

- [initWithURL:](#) (page 14)  
Prepares a player item with a given URL.
- + [playerItemWithURL:](#) (page 12)  
Returns a new player item, prepared to use a given URL.
- [initWithAsset:](#) (page 13)  
Initializes a new player item for a given asset.
- + [playerItemWithAsset:](#) (page 12)  
Returns a new player item for a given asset.

### Getting Information About an Item

- [asset](#) (page 7) *property*  
The underlying asset provided during initialization. (read-only)
- [tracks](#) (page 11) *property*  
An array of AVPlayerItemTrack objects. (read-only)
- [status](#) (page 11) *property*  
The status of the player item. (read-only)
- [loadedTimeRanges](#) (page 9) *property*  
The time ranges of the item that have been loaded. (read-only)
- [presentationSize](#) (page 10) *property*  
The size at which the visual portion of the item is presented by the player. (read-only)
- [timedMetadata](#) (page 11) *property*  
The timed metadata played most recently by the media stream. (read-only)
- [seekableTimeRanges](#) (page 10) *property*  
(read-only)
- [error](#) (page 8) *property*  
If the receiver's status is AVPlayerItemStatusFailed (page 17), this describes the error that caused the failure. (read-only)

### Moving the Playhead

- [stepByCount:](#) (page 16)  
Moves the player's current item's current time forward or backward by a specified number of steps.
- [seekToTime:](#) (page 15)  
Moves the playback cursor to a given time.
- [seekToTime:toleranceBefore:toleranceAfter:](#) (page 15)  
Moves the playback cursor within a specified time bound.

- [seekToDate:](#) (page 14)  
Moves the playback cursor to a given date.

## Information About Playback

- [playbackLikelyToKeepUp](#) (page 9) *property*  
Indicates whether the item will likely play through without stalling (read-only)
- [playbackBufferEmpty](#) (page 9) *property*  
Indicates whether playback has consumed all buffered media and that playback will stall or end. (read-only)
- [playbackBufferFull](#) (page 9) *property*  
Indicates whether the internal media buffer is full and that further I/O is suspended. (read-only)

## Timing Information

- [currentTime](#) (page 13)  
Returns the current time of the item.
- [forwardPlaybackEndTime](#) (page 8) *property*  
The time at which forward playback ends.
- [reversePlaybackEndTime](#) (page 10) *property*  
The time at which reverse playback ends.

## Settings

- [audioMix](#) (page 8) *property*  
The audio mix parameters to be applied during playback.
- [videoComposition](#) (page 11) *property*  
The video composition settings to be applied during playback.

## Properties

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

### **asset**

The underlying asset provided during initialization. (read-only)

```
@property(nonatomic, readonly) AVAsset *asset
```

### **Discussion**

#### **Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**audioMix**

The audio mix parameters to be applied during playback.

```
@property(n nonatomic, copy) AVAudioMix *audioMix
```

**Discussion****Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**error**

If the receiver's status is [AVPlayerItemStatusFailed](#) (page 17), this describes the error that caused the failure. (read-only)

```
@property(n nonatomic, readonly) NSError *error
```

**Discussion**

The value of this property is an error that describes what caused the receiver to no longer be able to be played.

If the receiver's status is not [AVPlayerItemStatusFailed](#) (page 17), the value of this property is `nil`.

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**forwardPlaybackEndTime**

The time at which forward playback ends.

```
@property(n nonatomic) CMTime forwardPlaybackEndTime
```

**Discussion****Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h



## loadedTimeRanges

The time ranges of the item that have been loaded. (read-only)

```
@property(n nonatomic, readonly) NSArray *loadedTimeRanges
```

### Discussion

The array contains `NSValue` objects containing a `CMTimeRange` value.

### Availability

Available in iOS 4.0 and later.

### Declared In

AVPlayerItem.h

## playbackBufferEmpty

Indicates whether playback has consumed all buffered media and that playback will stall or end. (read-only)

```
@property(n nonatomic, readonly, getter=isPlaybackBufferEmpty) BOOL playbackBufferEmpty
```

### Discussion

### Availability

Available in iOS 4.0 and later.

### Declared In

AVPlayerItem.h

## playbackBufferFull

Indicates whether the internal media buffer is full and that further I/O is suspended. (read-only)

```
@property(n nonatomic, readonly, getter=isPlaybackBufferFull) BOOL playbackBufferFull
```

### Discussion

### Availability

Available in iOS 4.0 and later.

### Declared In

AVPlayerItem.h

## playbackLikelyToKeepUp

Indicates whether the item will likely play through without stalling (read-only)

```
@property(n nonatomic, readonly, getter=isPlaybackLikelyToKeepUp) BOOL  
    playbackLikelyToKeepUp
```

**Discussion**

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

## presentationSize

The size at which the visual portion of the item is presented by the player. (read-only)

```
@property (n nonatomic, readonly) CGSize presentationSize;
```

**Discussion**

You can scale the presentation size to fit within the bounds of a player layer using its `videoGravity` property.

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

## reversePlaybackEndTime

The time at which reverse playback ends.

```
@property(n nonatomic) CMTIME reversePlaybackEndTime
```

**Discussion**

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

## seekableTimeRanges

(read-only)

```
@property(n nonatomic, readonly) NSArray *seekableTimeRanges
```

**Discussion**

The array contains `NSValue` objects containing a `CMTimeRange` value.

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**status**

The status of the player item. (read-only)

@property(n nonatomic, readonly) AVPlayerItemStatus status

**Discussion**

For example, whether the item is playable. For possible values, see “[AVPlayerItemStatus](#)” (page 16).

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**timedMetadata**

The timed metadata played most recently by the media stream. (read-only)

@property(n nonatomic, readonly) NSArray \*timedMetadata

**Discussion**

The array contains instances of AVMetadataItem.

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**tracks**

An array of AVPlayerItemTrack objects. (read-only)

@property(n nonatomic, readonly) NSArray \*tracks

**Discussion**

This property can change dynamically during playback. You can observe it using key-value observing.

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

**videoComposition**

The video composition settings to be applied during playback.

```
@property(n nonatomic, copy) AVVideoComposition *videoComposition
```

**Discussion****Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

## Class Methods

### playerItemWithAsset:

Returns a new player item for a given asset.

```
+ (AVPlayerItem *)playerItemWithAsset:(AVAsset *)asset
```

**Parameters**

*asset*

An asset to play.

**Return Value**

A new player item, initialized to play *asset*.

**Discussion****Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

### playerItemWithURL:

Returns a new player item, prepared to use a given URL.

```
+ (AVPlayerItem *)playerItemWithURL:(NSURL *)URL
```

**Parameters**

*URL*

An URL.

**Return Value**

A new player item, prepared to use *URL*.

**Special Considerations**

This method immediately returns the item, but with the status [AVPlayerItemStatusUnknown](#) (page 16).

If the URL contains valid data that can be used by the player item, the status later changes to [AVPlayerItemStatusReadyToPlay](#) (page 17).

If the URL contains no valid data or otherwise can't be used by the player item, the status later changes to [AVPlayerItemStatusFailed](#) (page 17).

#### Availability

Available in iOS 4.0 and later.

#### See Also

[@property status](#) (page 11)

#### Declared In

AVPlayerItem.h

## Instance Methods

### currentTime

Returns the current time of the item.

```
- (CMTIME)currentTime
```

#### Return Value

The current time of the item.

#### Discussion

#### Availability

Available in iOS 4.0 and later.

#### Declared In

AVPlayerItem.h

### initWithAsset:

Initializes a new player item for a given asset.

```
- (id)initWithAsset:(AVAsset *)asset
```

#### Parameters

*asset*

An asset to play.

#### Return Value

The receiver, initialized to play *asset*.

#### Discussion

#### Availability

Available in iOS 4.0 and later.

#### Declared In

AVPlayerItem.h

## initWithURL:

Prepares a player item with a given URL.

```
- (id)initWithURL:(NSURL *)URL
```

### Parameters

*URL*

An URL.

### Return Value

The receiver, prepared to use *URL*.

### Special Considerations

This method immediately returns the item, but with the status [AVPlayerItemStatusUnknown](#) (page 16).

If the URL contains valid data that can be used by the player item, the status later changes to [AVPlayerItemStatusReadyToPlay](#) (page 17).

If the URL contains no valid data or otherwise can't be used by the player item, the status later changes to [AVPlayerItemStatusFailed](#) (page 17).

### Availability

Available in iOS 4.0 and later.

### See Also

[@property status](#) (page 11)

### Declared In

AVPlayerItem.h

## seekToDate:

Moves the playback cursor to a given date.

```
- (BOOL)seekToDate:(NSDate *)date
```

### Parameters

*date*

The date to which to move the playback cursor.

### Return Value

YES if the playhead was moved to *date*, otherwise NO.

### Discussion

For playback content that is associated with a range of dates, this method moves the playhead to point within that range. This method will fail (return NO) if *date* is outside the range or if the content is not associated with a range of dates.

### Availability

Available in iOS 4.0 and later.

### See Also

- [seekToTime:](#) (page 15)

- [seekToDate:](#) (page 14)

**Declared In**

AVPlayerItem.h

**seekToTime:**

Moves the playback cursor to a given time.

- (void)seekToTime:(CMTime)*time*

**Parameters**

*time*

The time to which to move the playback cursor.

**Discussion**

The time seeked to may differ from the specified time for efficiency. For sample accurate seeking see [seekToTime:toleranceBefore:toleranceAfter:](#) (page 15).

**Availability**

Available in iOS 4.0 and later.

**See Also**

- [seekToTime:toleranceBefore:toleranceAfter:](#) (page 15)
- [seekToDate:](#) (page 14)

**Declared In**

AVPlayerItem.h

**seekToTime:toleranceBefore:toleranceAfter:**

Moves the playback cursor within a specified time bound.

- (void)seekToTime:(CMTime)*time* toleranceBefore:(CMTime)*toleranceBefore* toleranceAfter:(CMTime)*toleranceAfter*

**Parameters**

*time*

The time to which you would like to move the playback cursor.

*toleranceBefore*

The tolerance allowed before *time*.

*toleranceAfter*

The tolerance allowed after *time*.

**Discussion**

The time seeked to will be within the range [*time*-beforeTolerance, *time*+afterTolerance], and may differ from the specified time for efficiency. If you pass `kCMTimeZero` for both *toleranceBefore* and *toleranceAfter* (to request sample accurate seeking), you may incur additional decoding delay.

Passing `kCMTimePositiveInfinity` for both *toleranceBefore* and *toleranceAfter* is the same as messaging [seekToTime:](#) (page 15) directly.

**Availability**

Available in iOS 4.0 and later.

**See Also**

- [seekToTime:](#) (page 15)
- [seekToDate:](#) (page 14)

**Declared In**

AVPlayerItem.h

**stepByCount:**

Moves the player's current item's current time forward or backward by a specified number of steps.

```
- (void)stepByCount:(NSInteger)stepCount
```

**Parameters***stepCount*

The number of steps by which to move.

A positive number steps forward, a negative number steps backward.

**Discussion**

The size of each step depends on the receiver's enabled `AVPlayerItemTrack` objects (see [tracks](#) (page 11)).

**Availability**

Available in iOS 4.0 and later.

**Declared In**

AVPlayerItem.h

## Constants

**AVPlayerItemStatus**

Constants that represent the status of an item

```
enum {
    AVPlayerItemStatusUnknown,
    AVPlayerItemStatusReadyToPlay,
    AVPlayerItemStatusFailed
};
typedef NSInteger AVPlayerItemStatus;
```

**Constants**

AVPlayerItemStatusUnknown

The item's status is unknown.

Available in iOS 4.0 and later.

Declared in `AVPlayerItem.h`.



`AVPlayerItemStatusReadyToPlay`

The item is ready to play.

Available in iOS 4.0 and later.

Declared in `AVPlayerItem.h`.

`AVPlayerItemStatusFailed`

The item cannot be played.

Available in iOS 4.0 and later.

Declared in `AVPlayerItem.h`.

## Notifications

### **AVPlayerItemDidPlayToEndTimeNotification**

Posted when the item has played to its end time.

The notification's object is the item that finished playing.

**Important:** This notification may be posted on a different thread than the one on which the observer was registered.

#### **Availability**

Available in iOS 4.0 and later.

#### **Declared In**

`AVPlayerItem.h`



# Document Revision History

---

This table describes the changes to *AVPlayerItem Class Reference*.

Date	Notes
2010-05-15	New document that describes an object that represents the presentation state of an asset that's played by an AVPlayer object.

## REVISION HISTORY

### Document Revision History