# AVPlayerItem Class Reference

**Audio & Video**

**2010-05-15**

# Contents

# AVPlayerItem Class Reference

| | |
|---|---|
| **Inherits from** | NSObject |
| **Conforms to** | NSCopying<br>NSObject (NSObject) |
| **Framework** | /System/Library/Frameworks/AVFoundation.framework |
| **Availability** | Available in iOS 4.0 and later. |
| **Declared in** | 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(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(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(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(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(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(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(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 (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(nonatomic) CMTime reversePlaybackEndTime
```

**Discussion**

**Availability**
Available in iOS 4.0 and later.

**Declared In**
`AVPlayerItem.h`

## seekableTimeRanges

(read-only)

```
@property(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(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(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(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(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. |