Class

AVPlayerView

A view that displays content from a player and presents a native user interface to control playback.

macOS 10.9+

```
@MainActor
class AVPlayerView
```

Mentioned in

Implementing Trimming in a macOS Player

Overview

The player view supports several controls styles, ranging from no controls to controls matching the look of QuickTime Player. This makes it easy for you to tailor the presentation to best match your use of the player view. Regardless of the selected controls style, the player view always supports the following standard set of keyboard shortcuts to control playback:

- The Space bar plays and pauses playback.
- The right and left arrow keys step frame-by-frame through the video.
- JKL navigation:
- The J key rewinds. Press it multiple times to cycle through rewind speeds.
- The K key stops playback.
- The L key fast-forwards. Press it multiple times to cycle through fast-forward speeds.

The player view also makes it simple to add trimming capabilities to your player. Call the view's `beginTrimming(completionHandler:)` method to present a trimming UI that matches the QuickTime Player interface.

Topics

Customizing the user interface

`var controlsStyle: AVPlayerViewControlsStyle`

The player view's controls style.

`enum AVPlayerViewControlsStyle`

Constants that indicate which user interface controls the view displays.

`var showsFrameSteppingButtons: Bool`

A Boolean value that determines whether the player view displays frame stepping buttons.

`var showsSharingServiceButton: Bool`

A Boolean value that determines whether the player view displays a sharing service button.

`var showsFullScreenToggleButton: Bool`

A Boolean value that determines whether the player view displays a full-screen toggle button.

`var showsTimecodes: Bool`

A Boolean value that determines whether the player view displays timecodes, if available.

`var contentOverlayView: NSView?`

A view that adds additional custom views between the video content and the controls.

`var actionPopUpButtonMenu: NSMenu?`

An action pop-up button menu that the player view displays.

`var updatesNowPlayingInfoCenter: Bool`

A Boolean value that indicates whether the player view controller updates the Now Playing info center.

Customizing the video presentation

`var isReadyForDisplay: Bool`

A Boolean value that indicates whether the current player item's first video frame is ready for display.

`var videoBounds: NSRect`

The current size and position of the video image that displays within the player view's bounds.

`var videoGravity: AVLayerVideoGravity`

A value that determines how the player view displays video content within its bounds.

Configuring frame analysis

`var allowsVideoFrameAnalysis: Bool`

A Boolean value that indicates whether to perform video frame analysis.

`var videoFrameAnalysisTypes: AVVideoFrameAnalysisType`

`struct AVVideoFrameAnalysisType`

Constants that define the types of analysis a player view controller may perform on a paused video frame.

Configuring the playback speed

`var speeds: [AVPlaybackSpeed]`

A list of user-selectable playback speeds to show in the playback speed control.

`var selectedSpeed: AVPlaybackSpeed?`

The currently selected playback speed.

`func selectSpeed(AVPlaybackSpeed)`

Selects a specified playback speed.

`class AVPlaybackSpeed`

An object that represents a user-selectable playback speed in a playback user interface.

Configuring picture in picture

`var allowsPictureInPicturePlayback: Bool`

A Boolean value that determines whether the player view allows Picture in Picture playback.

```
var pictureInPictureDelegate: (any AVPlayerViewPictureInPictureDelegate
)?
```

The Picture in Picture delegate object.

```
protocol AVPlayerViewPictureInPictureDelegate
```

A protocol that defines the methods to implement to respond to Picture in Picture playback events.

Magnifying video

```
var allowsMagnification: Bool
```

A Boolean value that indicates whether the magnify gesture changes the video's view magnification.

```
var magnification: CGFloat
```

The factor by which the video's view is currently scaled.

```
func setMagnification(CGFloat, centeredAt: CGPoint)
```

Scales the video's view by a specified factor, and centers the result on a specified point.

Displaying the chapter and title

```
func flashChapterNumber(Int, chapterTitle: String?)
```

Displays the chapter number and title in the player view for a brief moment.

Trimming media

```
var canBeginTrimming: Bool
```

A Boolean value that indicates whether the player view can begin trimming.

```
func beginTrimming(completionHandler: ((AVPlayerViewTrimResult) -> Void
)?)
```

Puts the player view into trimming mode.

```
enum AVPlayerViewTrimResult
```

Constants that specify an action a user takes when trimming media in a player view.

Setting the player object

```
var player: AVPlayer?
```

The player instance that provides the media content for the view.

Setting the delegate object

`var delegate: (any AVPlayerViewDelegate)?`

The player view's delegate object.

`protocol AVPlayerViewDelegate`

A protocol that defines the methods to implement to participate in the player view's full-screen presentation life cycle.

High dynamic range

`var preferredDisplayDynamicRange: AVDisplayDynamicRange`

Describes how High Dynamic Range (HDR) video content renders.

`enum AVDisplayDynamicRange`

Describes how High Dynamic Range (HDR) video content renders.

Relationships

Inherits From

`NSView`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSAccessibilityElementProtocol`
`NSAccessibilityProtocol`
`NSAnimatablePropertyContainer`
`NSAppearanceCustomization`
`NSCoding`
`NSDraggingDestination`

```
NSObjectProtocol
NSStandardKeyBindingResponding
NSTouchBarProvider
NSUserActivityRestoring
NSUserInterfaceItemIdentification
Sendable
SendableMetatype
```

See Also

macOS playback and capture

Implementing Trimming in a macOS Player

Provide a QuickTime media-trimming experience in your macOS app.

`class AVCaptureView`

A view that displays standard user interface controls for capturing media data.