Advances in AVFoundation Playback

Waiting, looping, switching, widening, optimizing

Session 503

Sam Bushell Media Systems Architect

AVFoundation

File Playback

Network Playback

Video Processing

Metadata

Audio Mixing

Alternate Audio

AVFoundation

Photo Capture

Video Capture

Export

Subtitles

Editing

Video Effects

File Playback

Network Playback

Video Processing

Metadata

Audio Mixing

Alternate Audio

AVFoundation

Photo Capture

Video Capture

Export

Subtitles

Editing

Video Effects

Kinds of Playback

Kinds of Playback

Local File

file:///.../example.MOV

Kinds of Playback

Local File

file:///.../example.MOV

Progressive Download

https://example.com/example.MOV

Kinds of Playback

Local File	file:///.../example.MOV	
Progressive Download	https://example.com/example.MOV	
HTTP Live Streaming	master playlist	

Kinds of Playback

Local File	file:///.../example.MOV	
Progressive Download	https://example.com/example.MOV	
HTTP Live Streaming	master playlist	video 6Mbit playlist video 4Mbit playlist video 2Mbit playlist audio stereo playlist audio surround playlist

Kinds of Playback

Local File	file:///.../example.MOV		
Progressive Download	https://example.com/example.MOV		
HTTP Live Streaming	master playlist	video 6Mbit playlist	 segments
		video 4Mbit playlist	 segments
		video 2Mbit playlist	 segments
		audio stereo playlist	 segments
		audio surround playlist	 segments

Overview

Overview

Automatic waiting for buffering

Overview

Automatic waiting for buffering

Simple way to loop playback

Overview

Automatic waiting for buffering

Simple way to loop playback

Playback refinements

Overview

Automatic waiting for buffering

Simple way to loop playback

Playback refinements

Wide color

Overview

Automatic waiting for buffering

Simple way to loop playback

Playback refinements

Wide color

Best practices for being awesome

Buffering

Please wait...

Media Playback Over the Internet

Playback is at the mercy of the network!

- Start too soon → playback may stall
- Start too late → user unhappy
- Start when likely to keep up → just right

AVPlayerItem Buffering State Properties

Existing

```
playbackLikelyToKeepUp
playbackBufferFull
playbackBufferEmpty
```

For progressive-download playback, in iOS 9

- Wait until `playbackLikelyToKeepUp` or `playbackBufferFull` before setting `AVPlayer.rate`

For HLS, rules are simpler

- Set `AVPlayer.rate` and it will automatically wait for buffering before playback begins

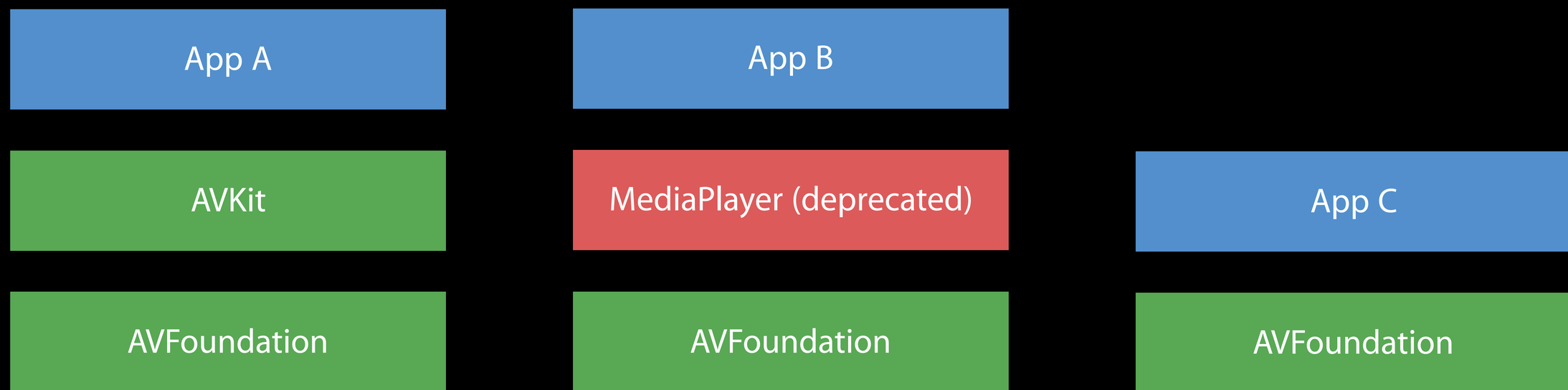
AVPlayer in iOS 10 / macOS Sierra / tvOS 10

NEW

Same rules for progressive and HLS

- Set `AVPlayer.rate` when user clicks play
- Automatically waits to buffer to avoid stalling

If network drops and playback stalls, playback will automatically resume when buffered


App A
App B
AVKit
MediaPlayer (deprecated)
App C
AVFoundation
AVFoundation
AVFoundation

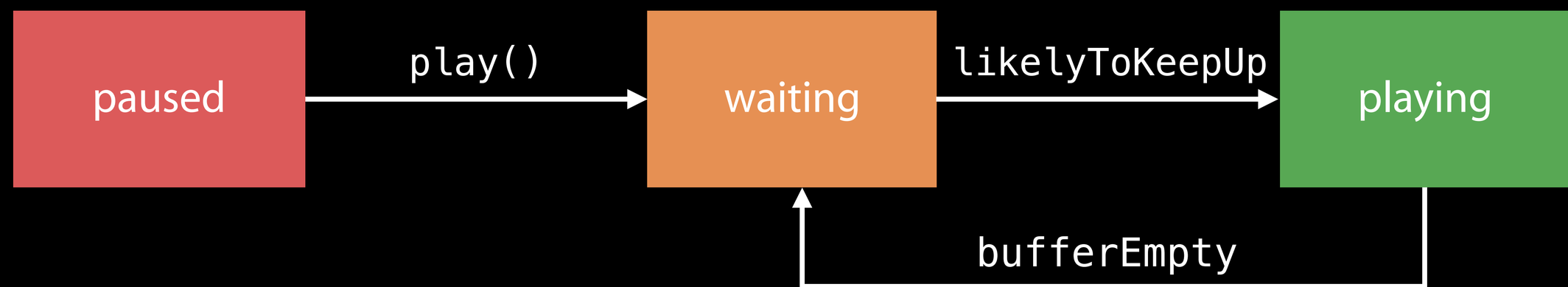
Autoplay or Autowait?

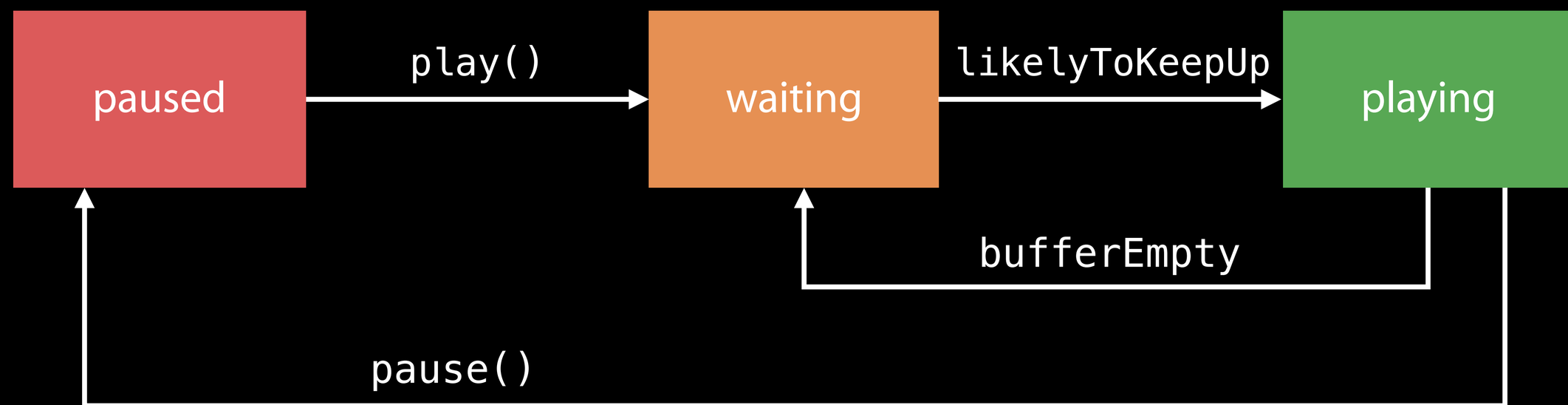
automaticallyWaitsToMinimizeStalling

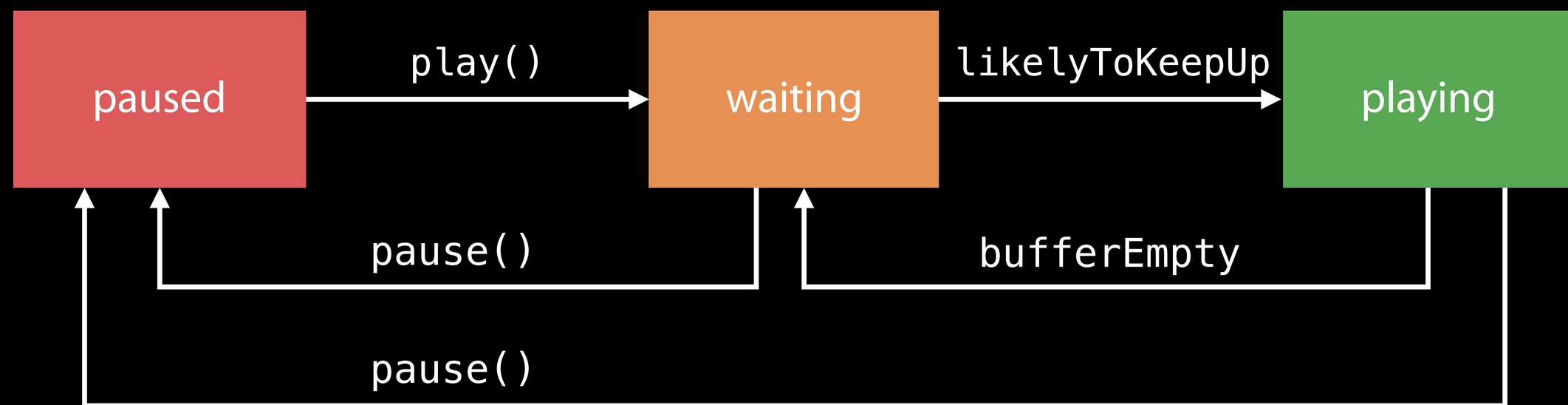

paused
waiting
playing

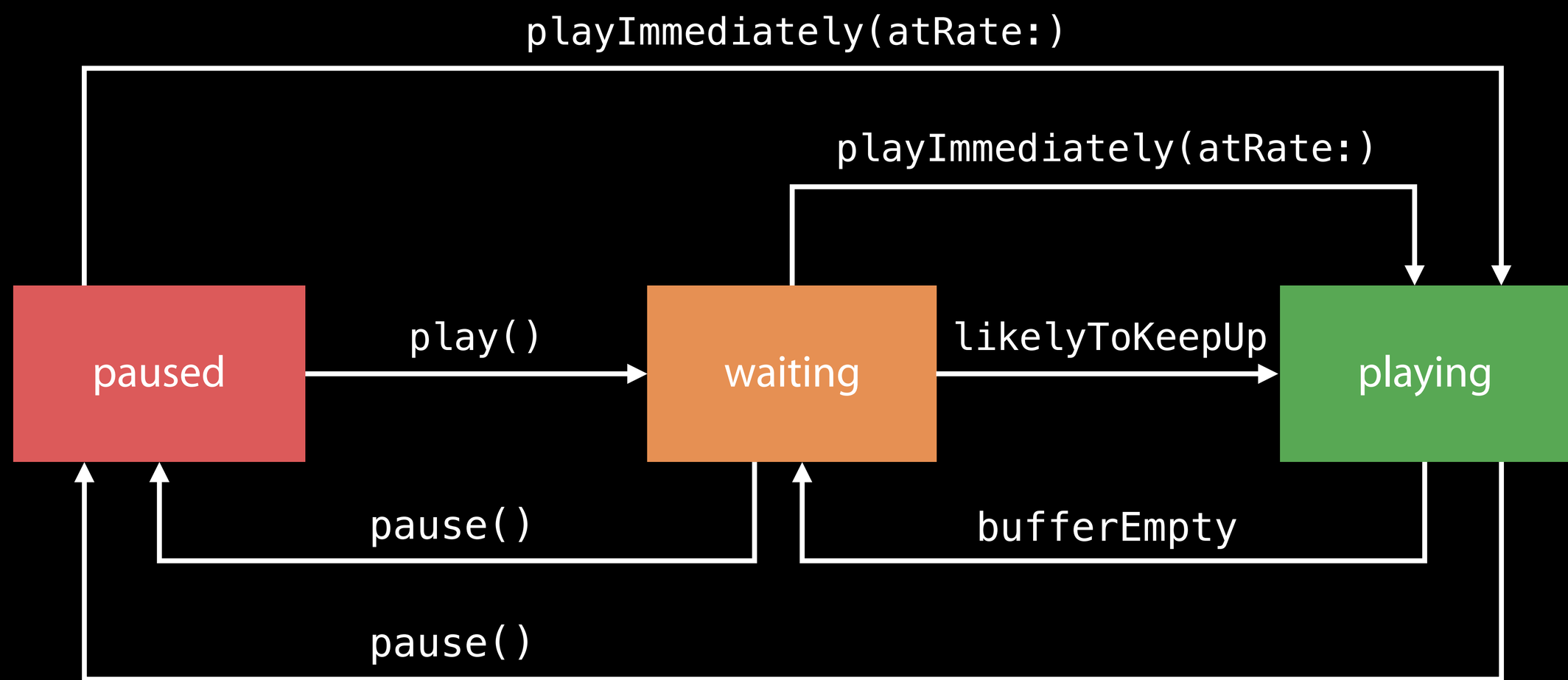

paused
play()
waiting
playing


paused
play()
waiting
likelyToKeepUp
playing


paused
play()
waiting
likelyToKeepUp
playing
bufferEmpty


paused
play()
waiting
likelyToKeepUp
playing
bufferEmpty
pause()


paused
play()
waiting
likelyToKeepUp
playing
pause()
bufferEmpty
pause()


playImmediately(atRate:)
playImmediately(atRate:)
paused
play()
waiting
likelyToKeepUp
playing
pause()
bufferEmpty
pause()

AVPlayer.rate

Might not mean what you thought it meant

AVPlayer.rate

Might not mean what you thought it meant

`AVPlayer.rate`

the app's requested playback rate

AVPlayer.rate

Might not mean what you thought it meant

`AVPlayer.rate`

the app's requested playback rate

`AVPlayerItem.timebase.rate`

the rate at which playback is actually occurring

AVPlayer.rate

Might not mean what you thought it meant

`AVPlayer.rate`

the app's requested playback rate

`AVPlayerItem.timebase.rate`

the rate at which playback is actually occurring

`AVPlayer.timeControlStatus`

Paused, WaitingToPlayAtSpecifiedRate, Playing

NEW

AVPlayer.rate

Might not mean what you thought it meant

`AVPlayer.rate`
the app's requested playback rate

`AVPlayerItem.timebase.rate`
the rate at which playback is actually occurring

`AVPlayer.timeControlStatus` NEW
Paused, WaitingToPlayAtSpecifiedRate, Playing

`AVPlayer.reasonForWaitingToPlay` NEW

AVPlayer.rate

Might not mean what you thought it meant

waiting

`AVPlayer.rate`
the app's requested playback rate

`AVPlayerItem.timebase.rate`
the rate at which playback is actually occurring

`AVPlayer.timeControlStatus` NEW
Paused, WaitingToPlayAtSpecifiedRate, Playing

`AVPlayer.reasonForWaitingToPlay` NEW

AVPlayer.rate

Might not mean what you thought it meant

waiting

`AVPlayer.rate` the app's requested playback rate	`1.0`
`AVPlayerItem.timebase.rate` the rate at which playback is actually occurring	
`AVPlayer.timeControlStatus` Paused, WaitingToPlayAtSpecifiedRate, Playing (NEW)	
`AVPlayer.reasonForWaitingToPlay` (NEW)	

AVPlayer.rate

Might not mean what you thought it meant

waiting

`AVPlayer.rate` the app's requested playback rate	`1.0`
`AVPlayerItem.timebase.rate` the rate at which playback is actually occurring	`0.0`
`AVPlayer.timeControlStatus` NEW Paused, WaitingToPlayAtSpecifiedRate, Playing	
`AVPlayer.reasonForWaitingToPlay` NEW	

AVPlayer.rate

Might not mean what you thought it meant

waiting

`AVPlayer.rate` the app's requested playback rate	`1.0`
`AVPlayerItem.timebase.rate` the rate at which playback is actually occurring	`0.0`
`AVPlayer.timeControlStatus` Paused, WaitingToPlayAtSpecifiedRate, Playing (NEW)	`WaitingToPlayAtSpecifiedRate`
`AVPlayer.reasonForWaitingToPlay` (NEW)	

AVPlayer.rate

Might not mean what you thought it meant

waiting

`AVPlayer.rate` the app's requested playback rate	`1.0`
`AVPlayerItem.timebase.rate` the rate at which playback is actually occurring	`0.0`
`AVPlayer.timeControlStatus` Paused, WaitingToPlayAtSpecifiedRate, Playing	NEW `WaitingToPlayAtSpecifiedRate`
`AVPlayer.reasonForWaitingToPlay`	NEW `WaitingToMinimizeStallsReason`

Demo

Autowait

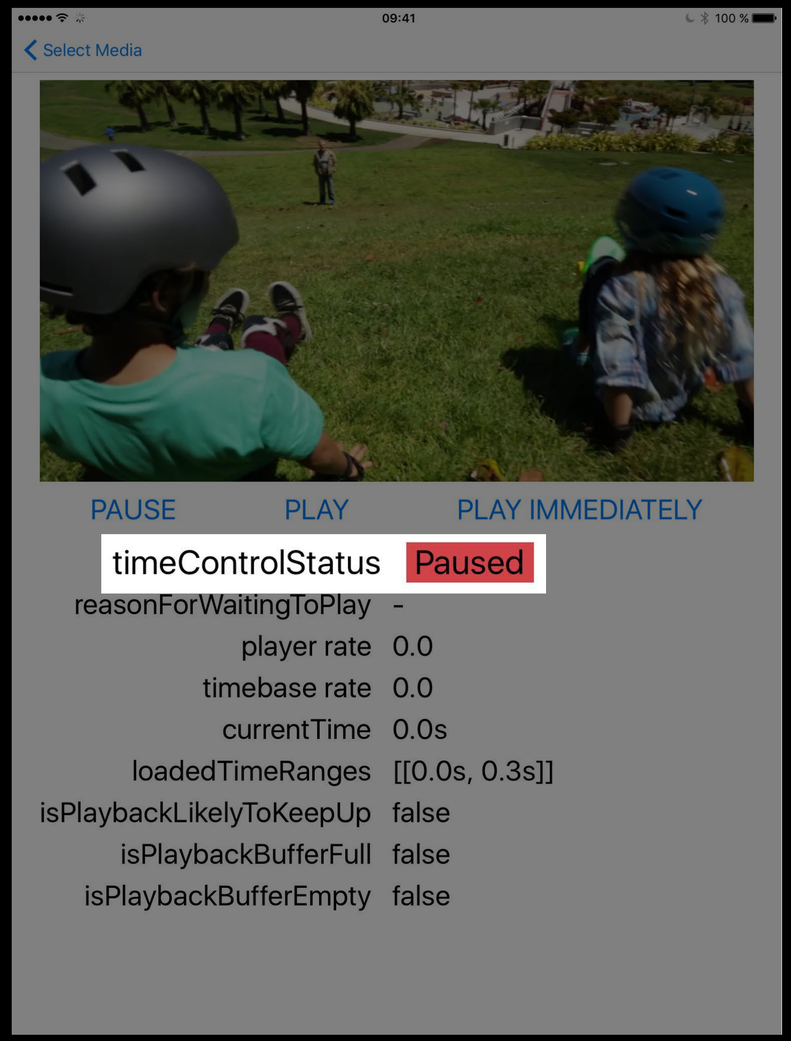
Moritz Wittenhagen

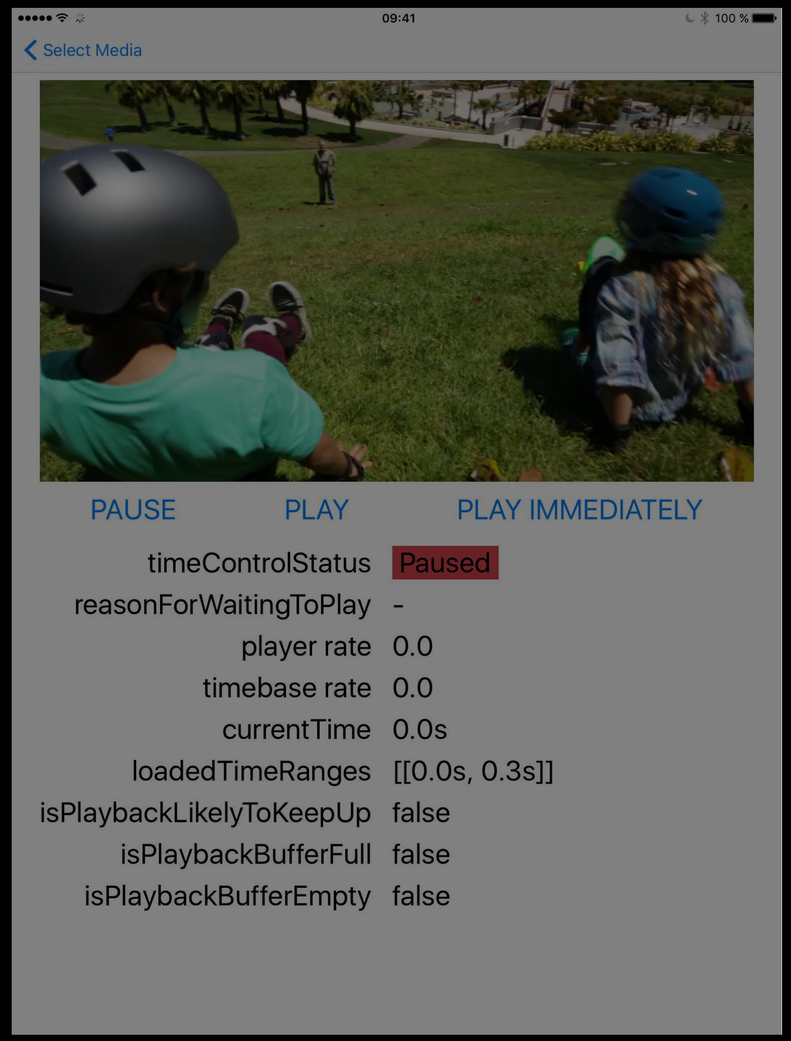
09:41 100 %

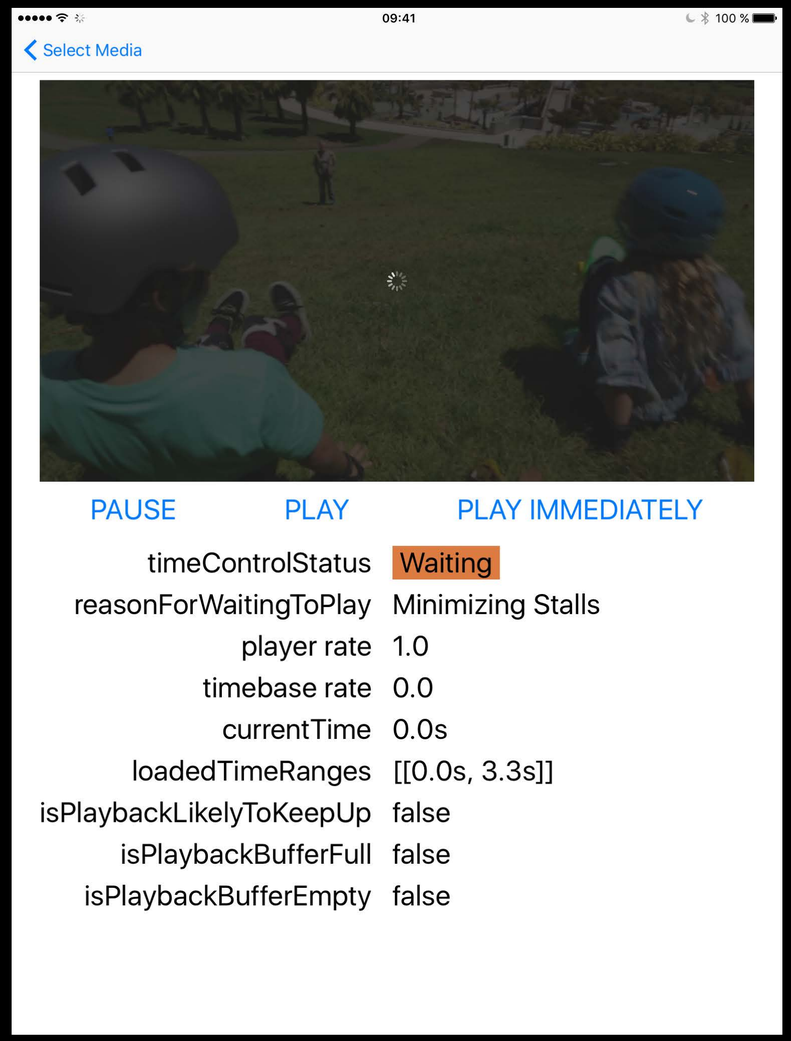
Select Media

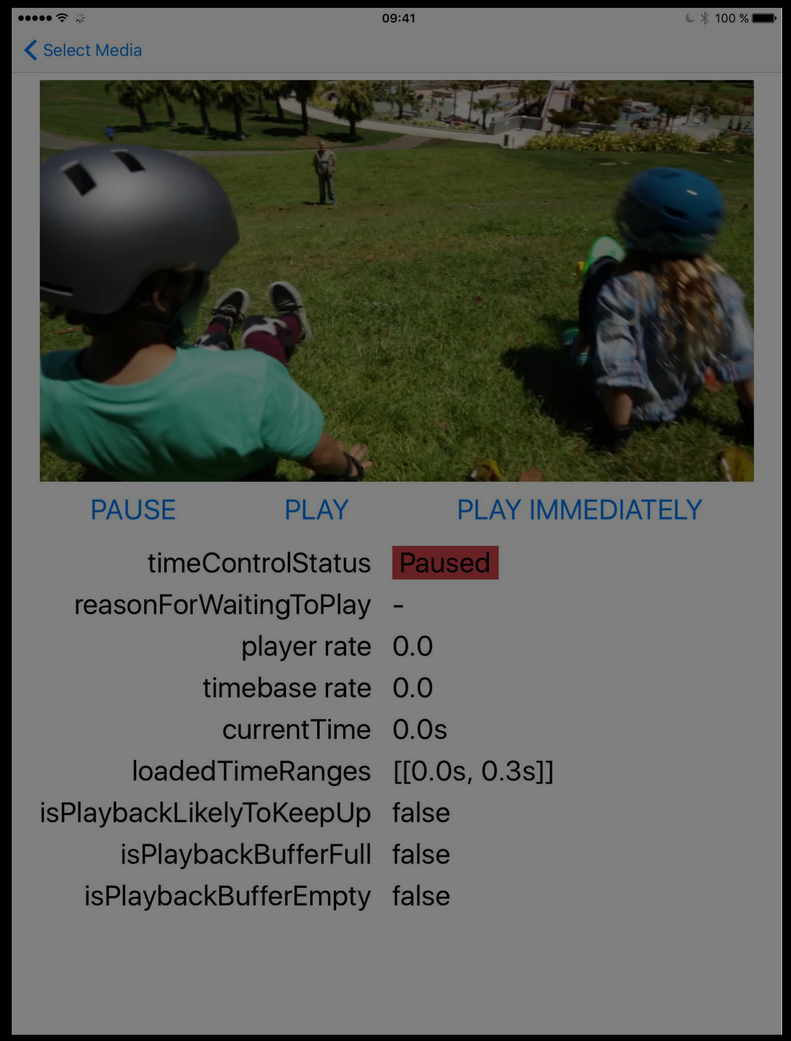
PAUSE PLAY PLAY IMMEDIATELY

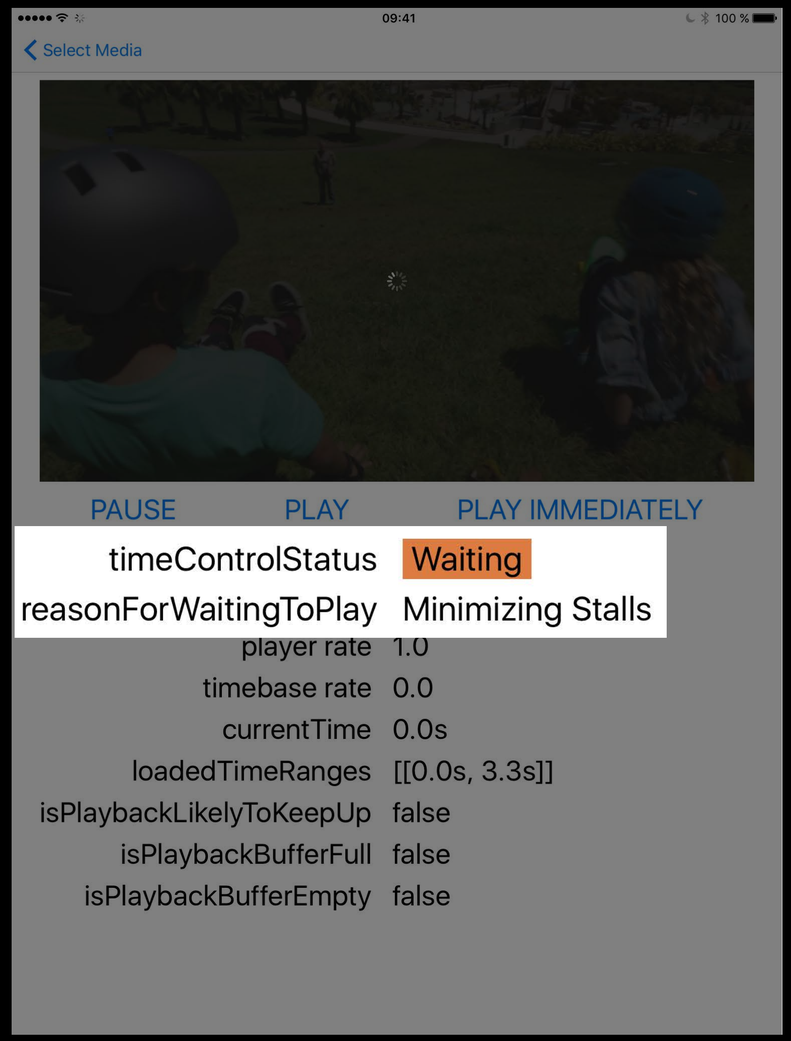
timeControlStatus	Paused
reasonForWaitingToPlay	-
player rate	0.0
timebase rate	0.0
currentTime	0.0s
loadedTimeRanges	[[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp	false
isPlaybackBufferFull	false
isPlaybackBufferEmpty	false

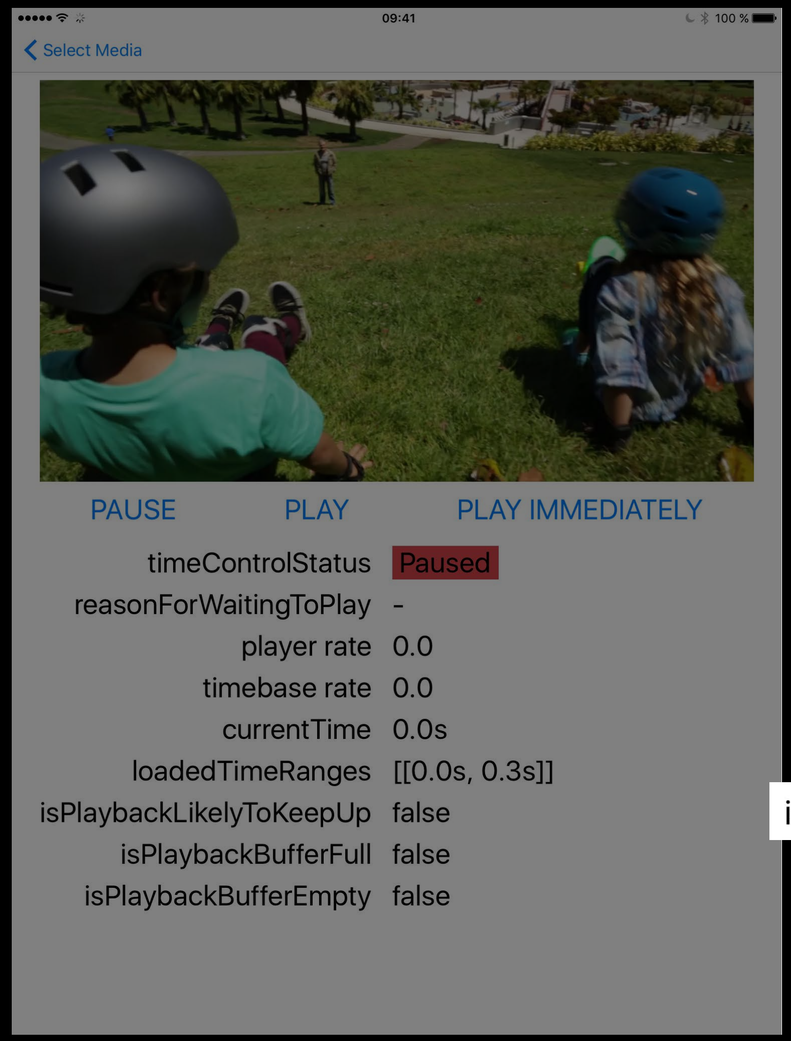

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

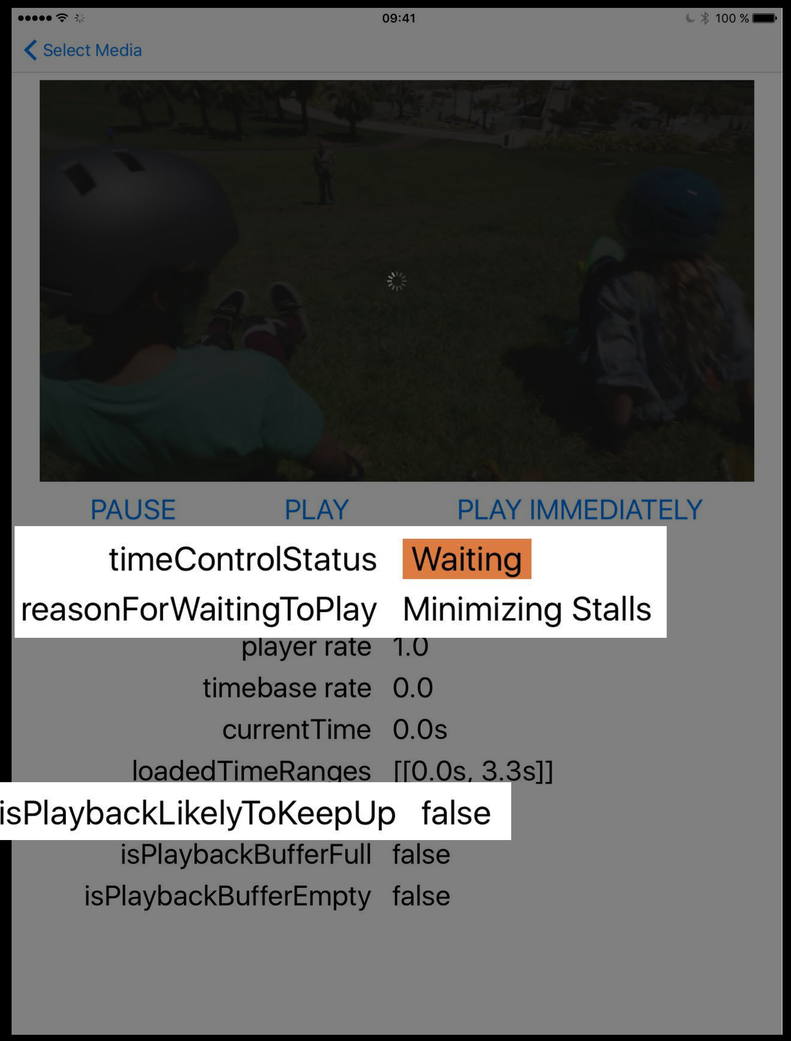

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

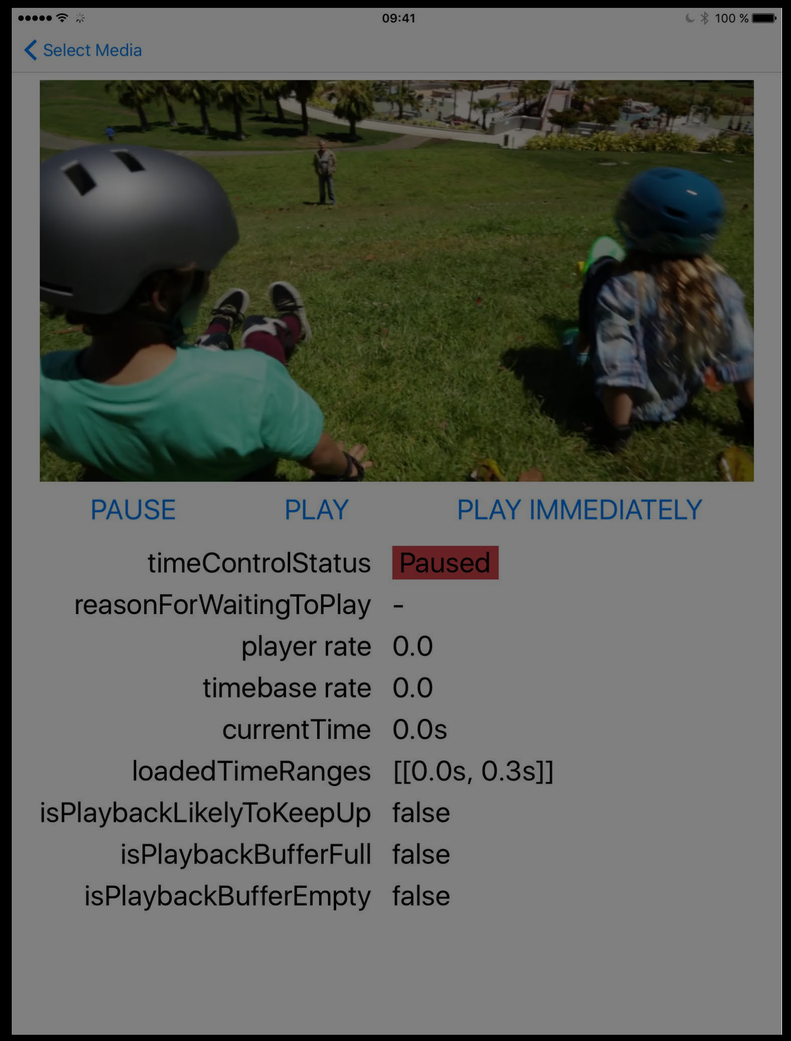

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

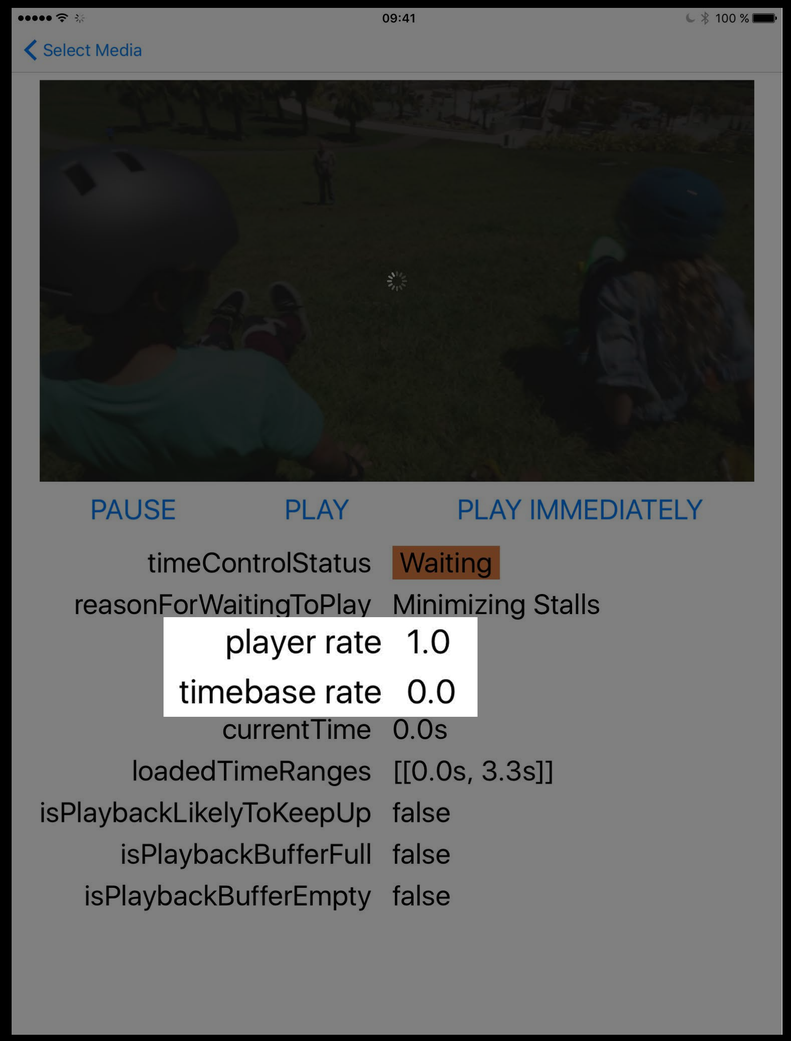

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

09:41 100 %

Select Media

PAUSE PLAY PLAY IMMEDIATELY

timeControlStatus	Paused
reasonForWaitingToPlay	-
player rate	0.0
timebase rate	0.0
currentTime	0.0s
loadedTimeRanges	[[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp	false
isPlaybackBufferFull	false
isPlaybackBufferEmpty	false

09:41 100 %

Select Media

PAUSE PLAY PLAY IMMEDIATELY

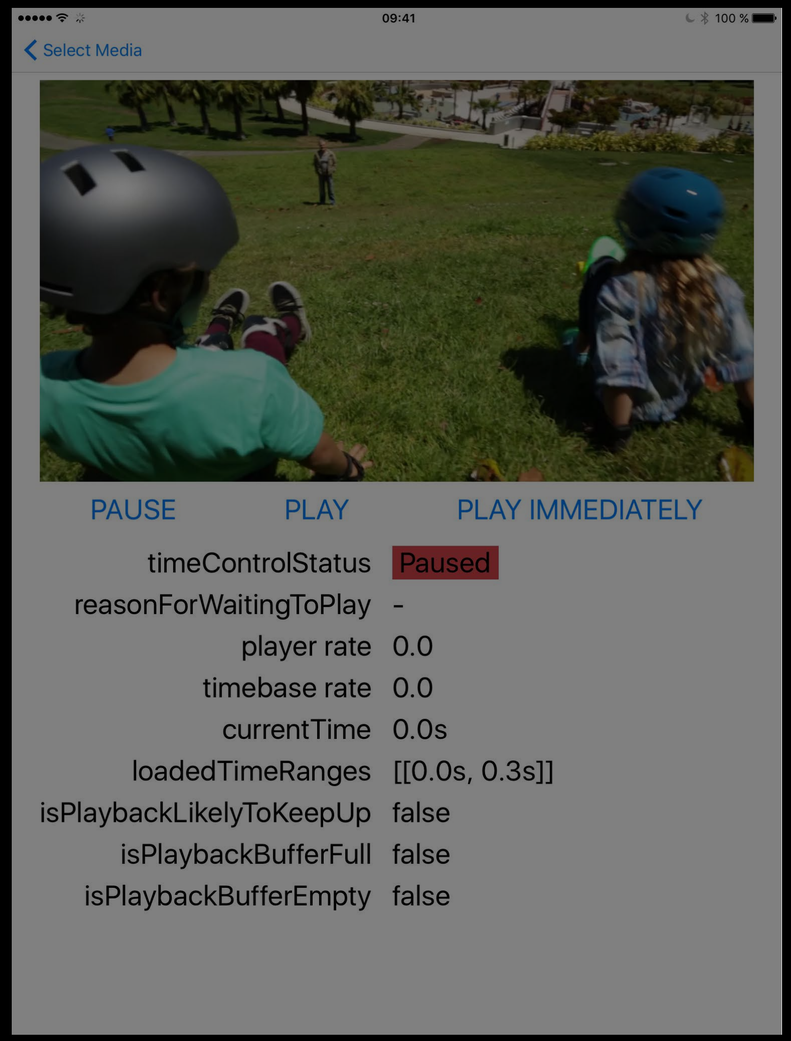
timeControlStatus	Waiting
reasonForWaitingToPlay	Minimizing Stalls
player rate	1.0
timebase rate	0.0
currentTime	0.0s
loadedTimeRanges	[[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp	false
isPlaybackBufferFull	false
isPlaybackBufferEmpty	false

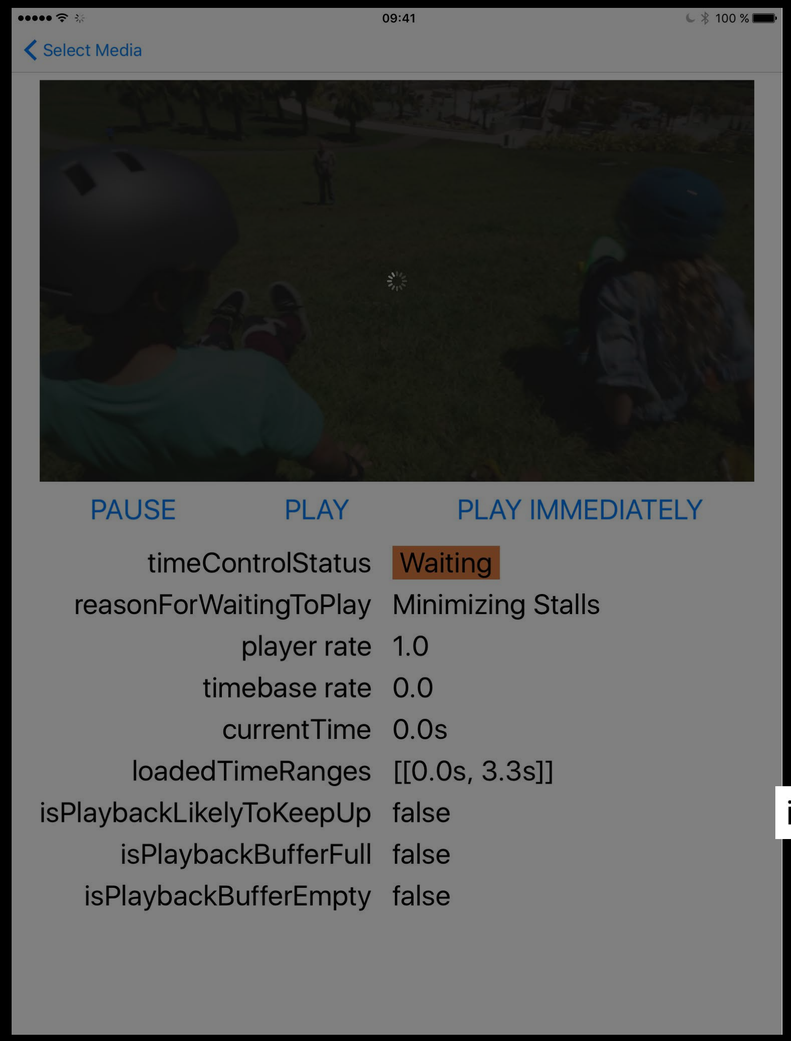
09:41 100 %

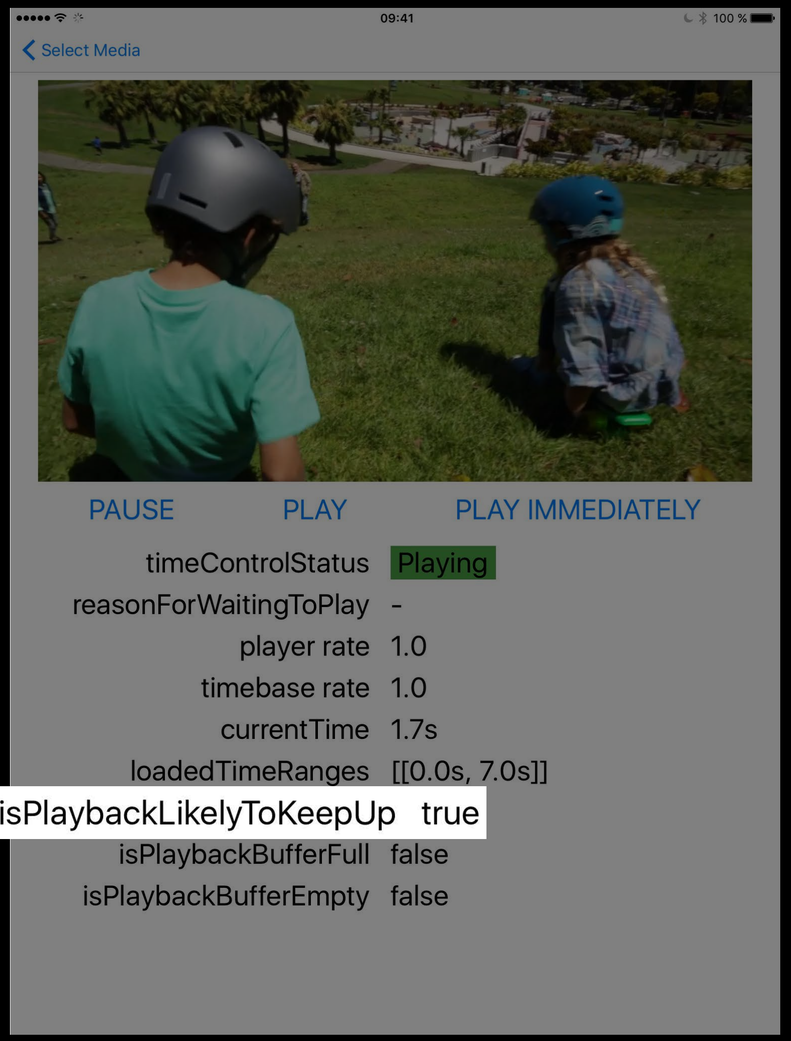
Select Media

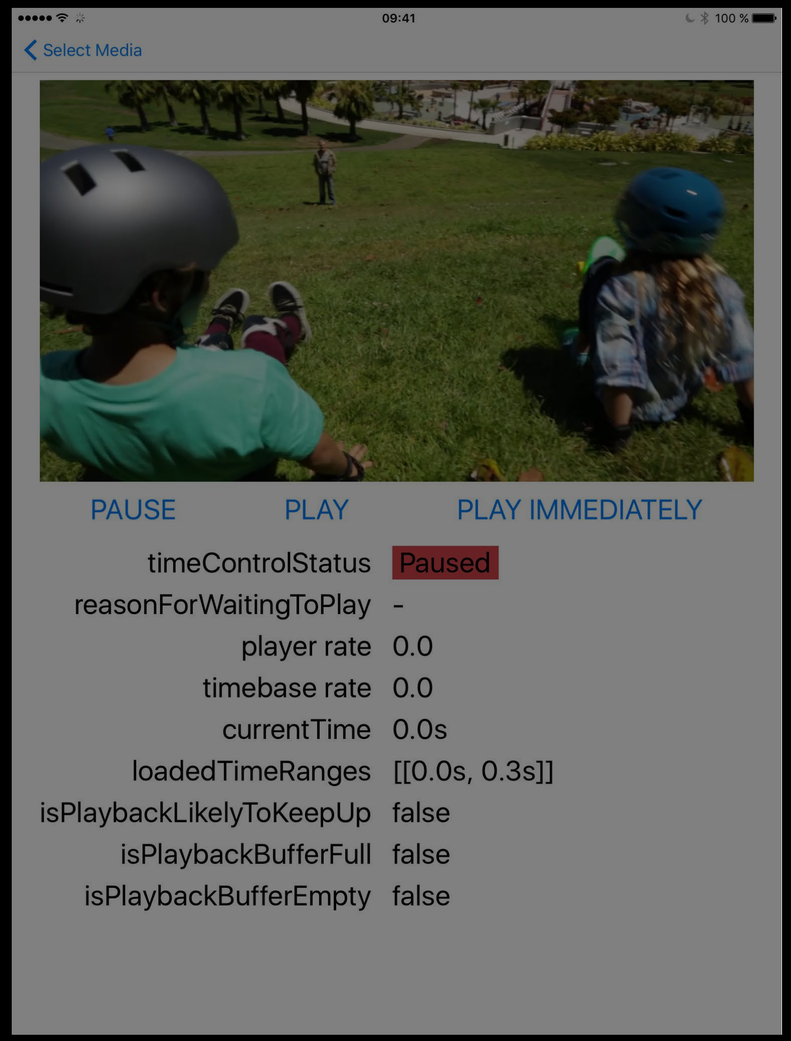
PAUSE PLAY PLAY IMMEDIATELY

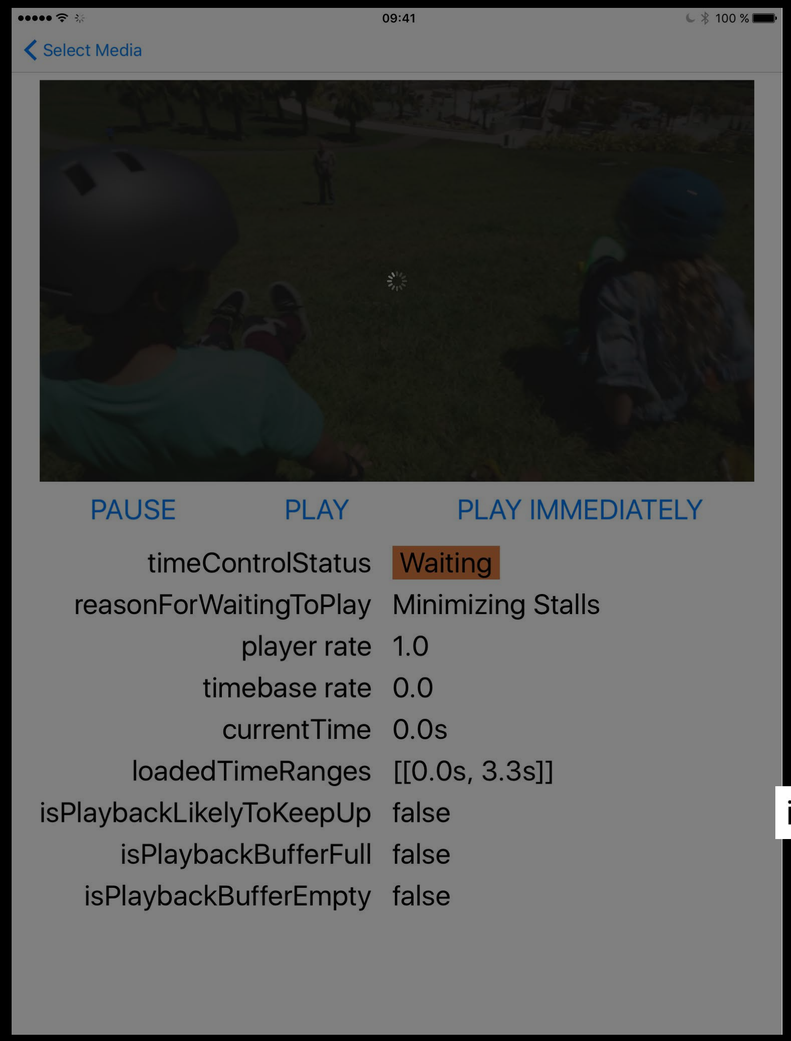
timeControlStatus	Playing
reasonForWaitingToPlay	-
player rate	1.0
timebase rate	1.0
currentTime	1.7s
loadedTimeRanges	[[0.0s, 7.0s]]
isPlaybackLikelyToKeepUp	true
isPlaybackBufferFull	false
isPlaybackBufferEmpty	false

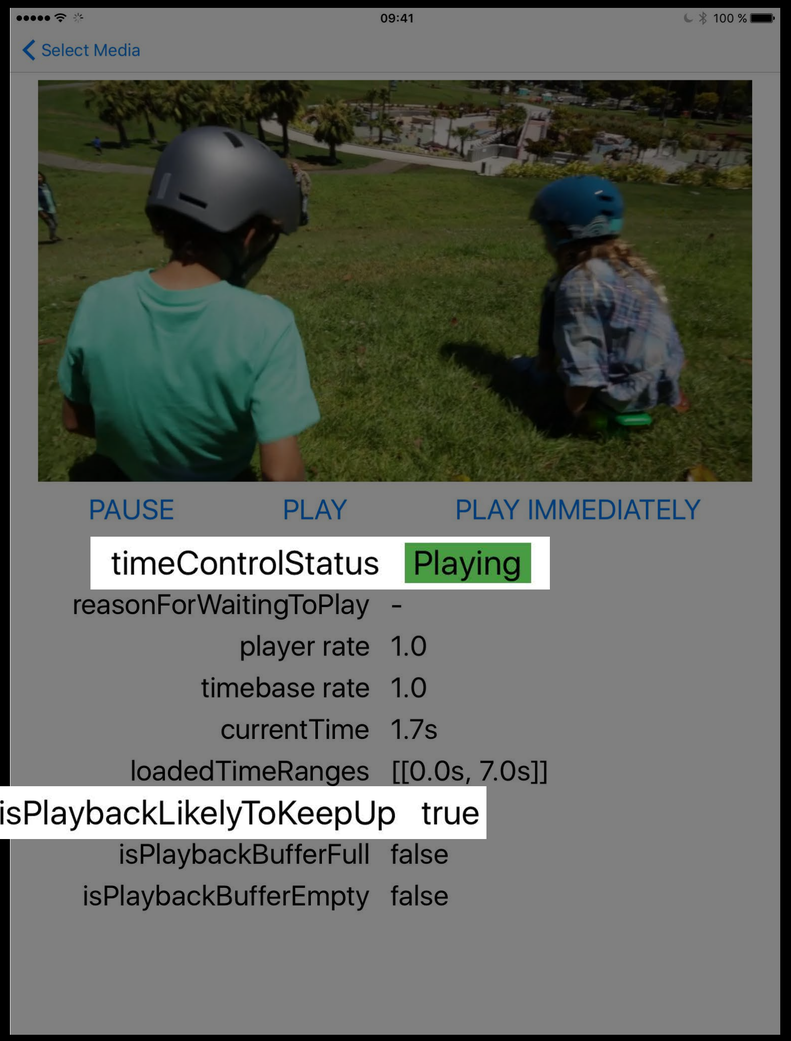

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

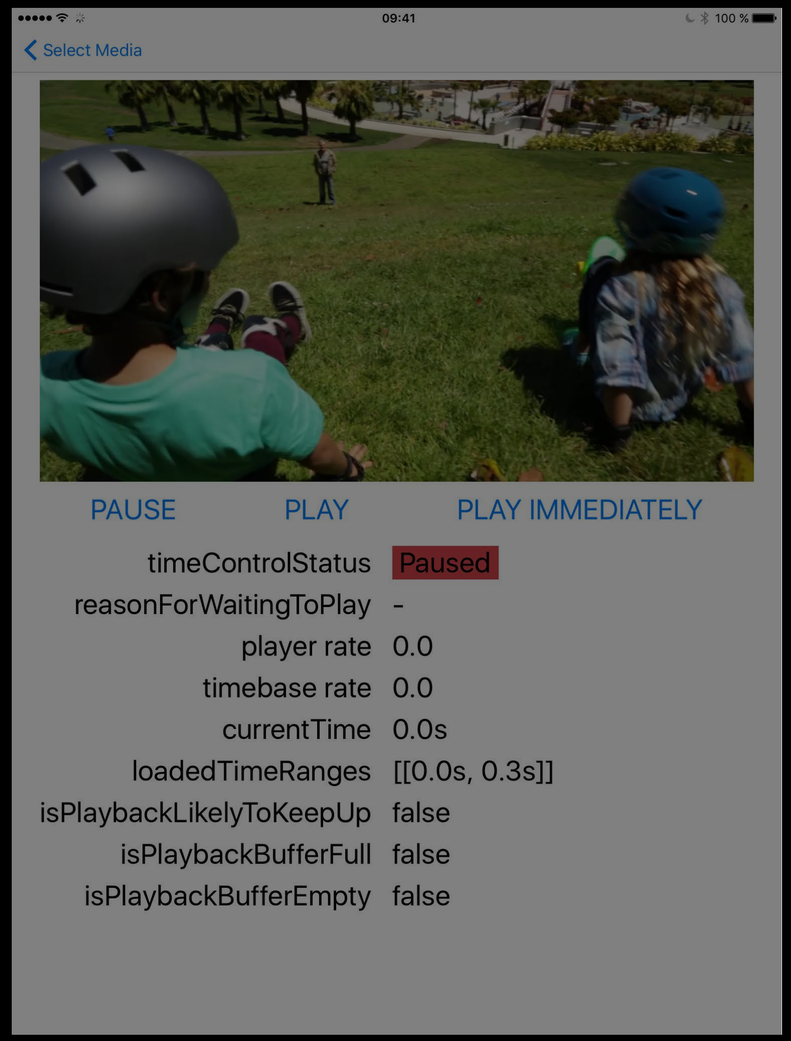

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

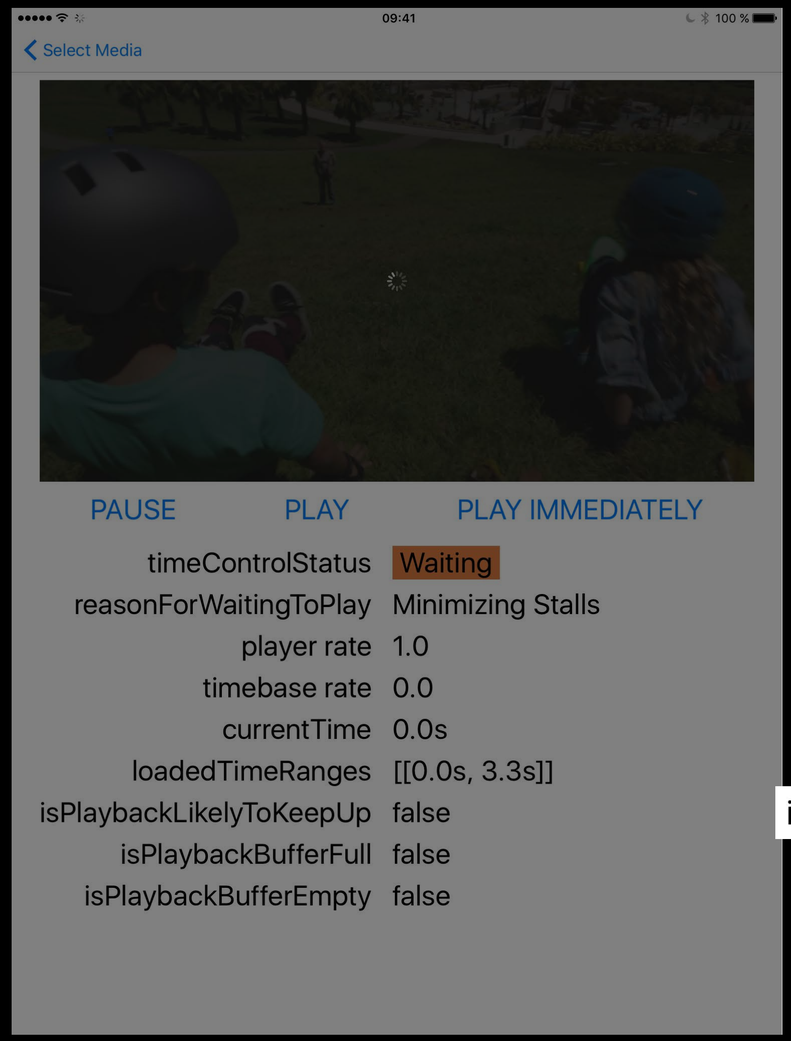

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Playing
reasonForWaitingToPlay -
player rate 1.0
timebase rate 1.0
currentTime 1.7s
loadedTimeRanges [[0.0s, 7.0s]]
isPlaybackLikelyToKeepUp true
isPlaybackBufferFull false
isPlaybackBufferEmpty false

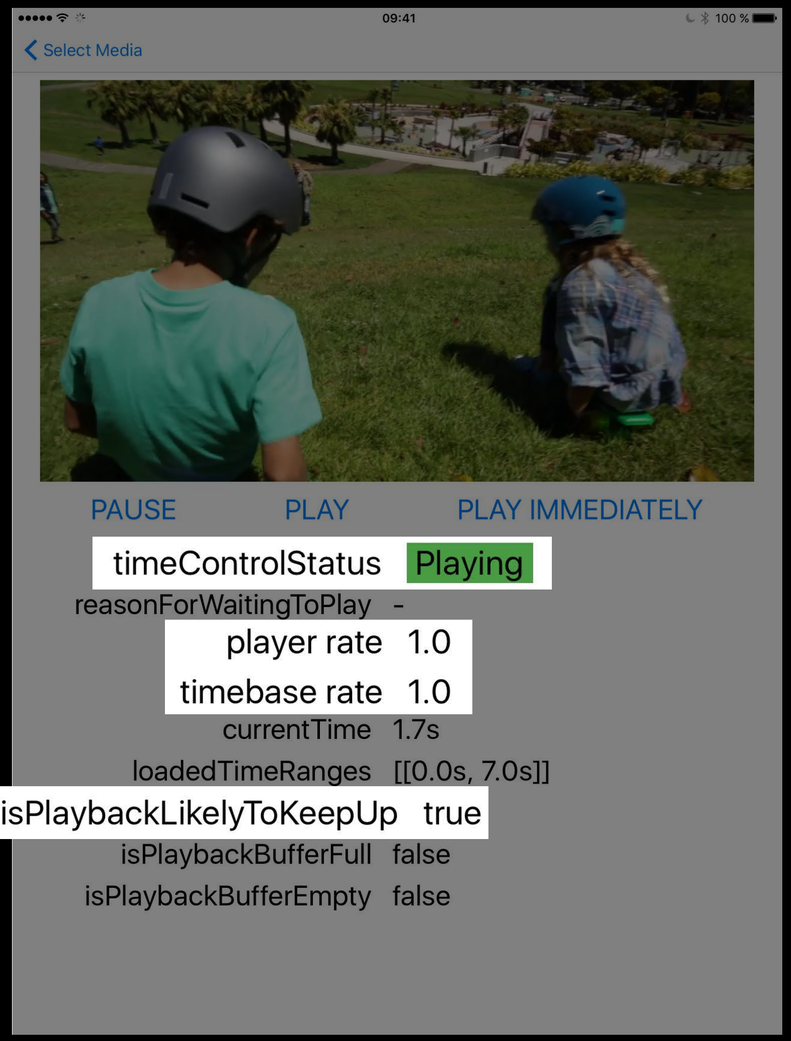

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Playing
reasonForWaitingToPlay -
player rate 1.0
timebase rate 1.0
currentTime 1.7s
loadedTimeRanges [[0.0s, 7.0s]]
isPlaybackLikelyToKeepUp true
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Paused
reasonForWaitingToPlay -
player rate 0.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 0.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false

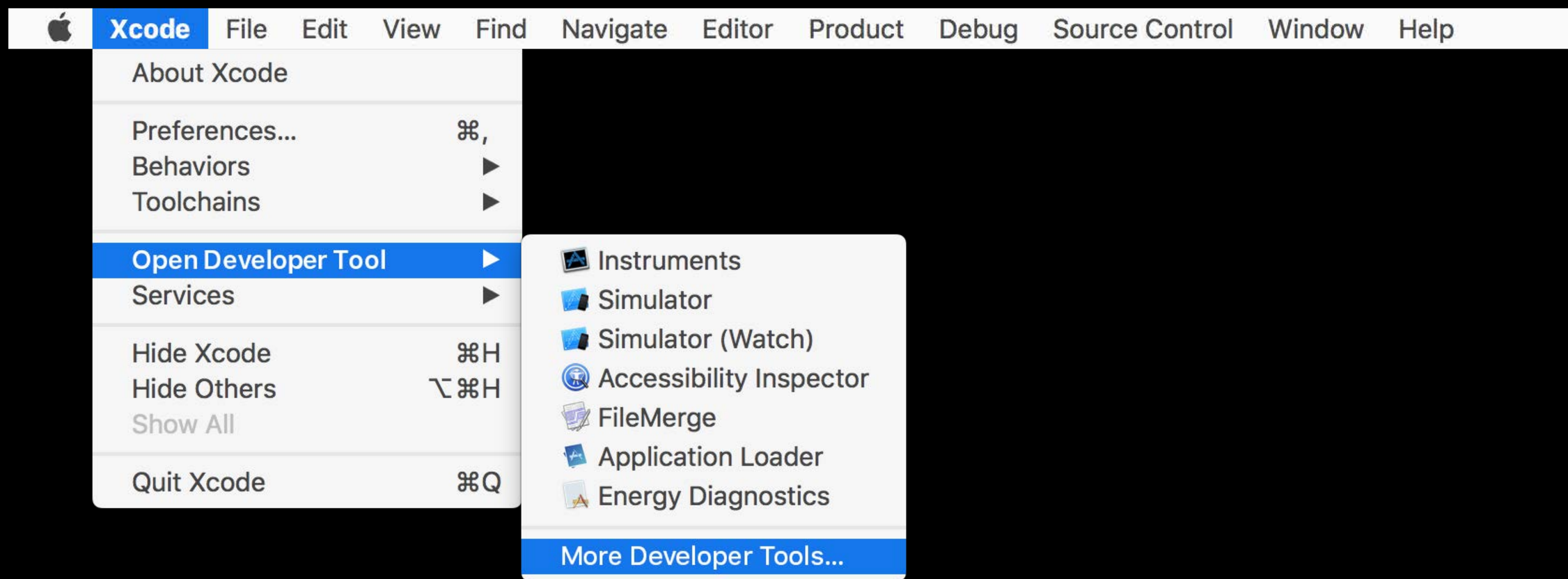

09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Waiting
reasonForWaitingToPlay Minimizing Stalls
player rate 1.0
timebase rate 0.0
currentTime 0.0s
loadedTimeRanges [[0.0s, 3.3s]]
isPlaybackLikelyToKeepUp false
isPlaybackBufferFull false
isPlaybackBufferEmpty false


09:41
100 %
Select Media
PAUSE
PLAY
PLAY IMMEDIATELY
timeControlStatus Playing
reasonForWaitingToPlay -
player rate 1.0
timebase rate 1.0
currentTime 1.7s
loadedTimeRanges [[0.0s, 7.0s]]
isPlaybackLikelyToKeepUp true
isPlaybackBufferFull false
isPlaybackBufferEmpty false

`rate` vs `timeControlStatus`

`AVPlayer.rate`	`AVPlayer.timeControlStatus`
`0.0`	paused
`1.0`	waiting
`1.0`	playing

Finding the Network Link Conditioner



Finding the Network Link Conditioner

Apple Inc. developer.apple.com/downloads/?name=for%20Xcod

https://developer.apple.com/downloads/?name=for%20Xcode

Developer | Platforms | Resources | Program | Support | Account

Downloads for Apple Developers

Hi, Sam Bushell

for Xcode

CATEGORIES

- OS X 123
- Developer Tools 324
- iOS 23
- Applications 11
- OS X Server 12

	Description	Release Date
–	Hardware IO Tools for Xcode 7.3	Mar 20, 2016
	This package includes additional hardware i/o tools formerly bundled in the Xcode installer. These tools include: Apple Bluetooth Guidelines Validation, Bluetooth Explorer, HomeKit Accessory Simulator, IO Registry Explorer, Network Link Conditioner.prefpane, PacketLogger and Printer Simulator. These graphics tools support running on OS X 10.11.	Hardware IO Tools for Xcode 7.3.dmg 12.1 MB
+	Command Line Tools (OS X 10.11) for Xcode 7.3	Mar 20, 2016

Cautions

Cautions

Enabled automatically if app linked on or after iOS 10, OSX 10.12, tvOS 10

- `AVPlayer.automaticallyWaitsToMinimizeStalling = true`

Cautions

Enabled automatically if app linked on or after iOS 10, OSX 10.12, tvOS 10

- `AVPlayer.automaticallyWaitsToMinimizeStalling = true`

Opt out if using `setRate(..., time:..., atHostTime:...)` to synchronize playback with external timeline

- `AVPlayer.automaticallyWaitsToMinimizeStalling = false`
- Otherwise, NSException

Cautions

Enabled automatically if app linked on or after iOS 10, OSX 10.12, tvOS 10

- `AVPlayer.automaticallyWaitsToMinimizeStalling = true`

Opt out if using `setRate(..., time:..., atHostTime:...)` to synchronize playback with external timeline

- `AVPlayer.automaticallyWaitsToMinimizeStalling = false`
- Otherwise, NSException

Never use the player rate to project currentTime into the future

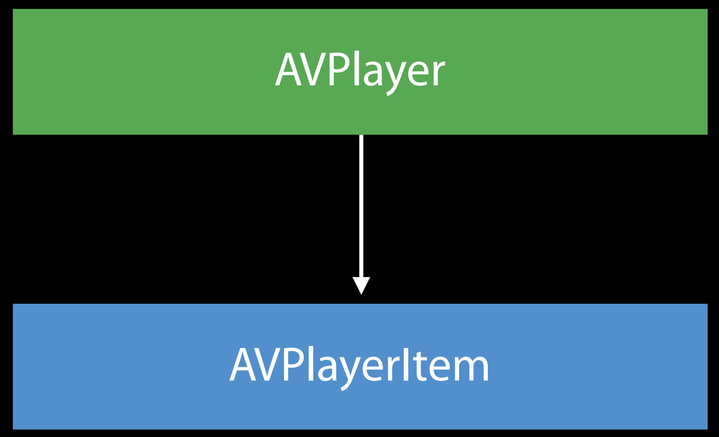
- Use currentItem's timebase rate for that instead

Looping

Made easier

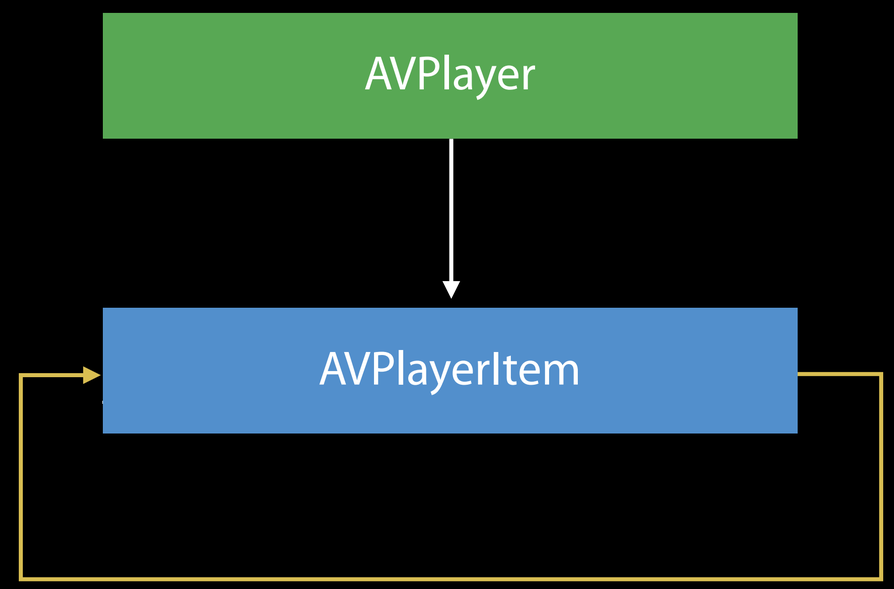
How Do You Loop an AVPlayerItem?

When end reached, rewind?



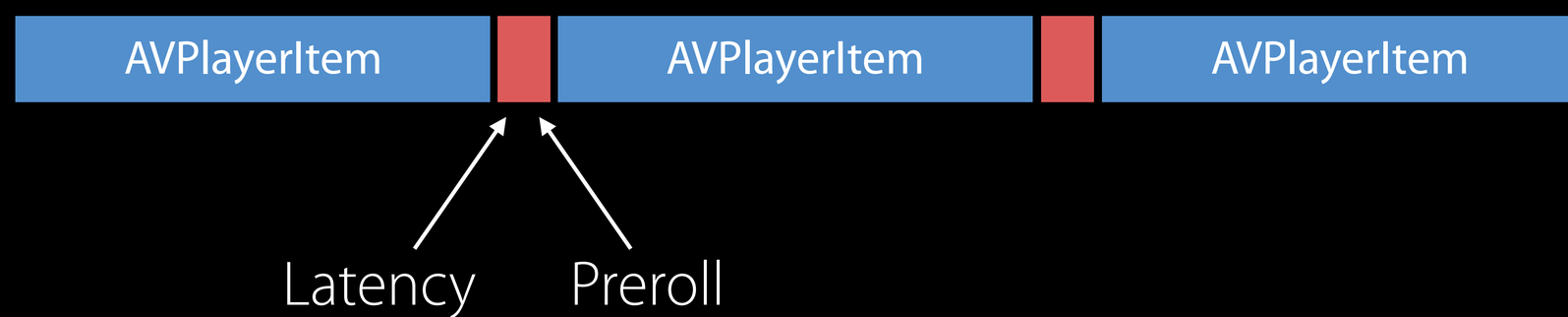
How Do You Loop an AVPlayerItem?

When end reached, rewind?



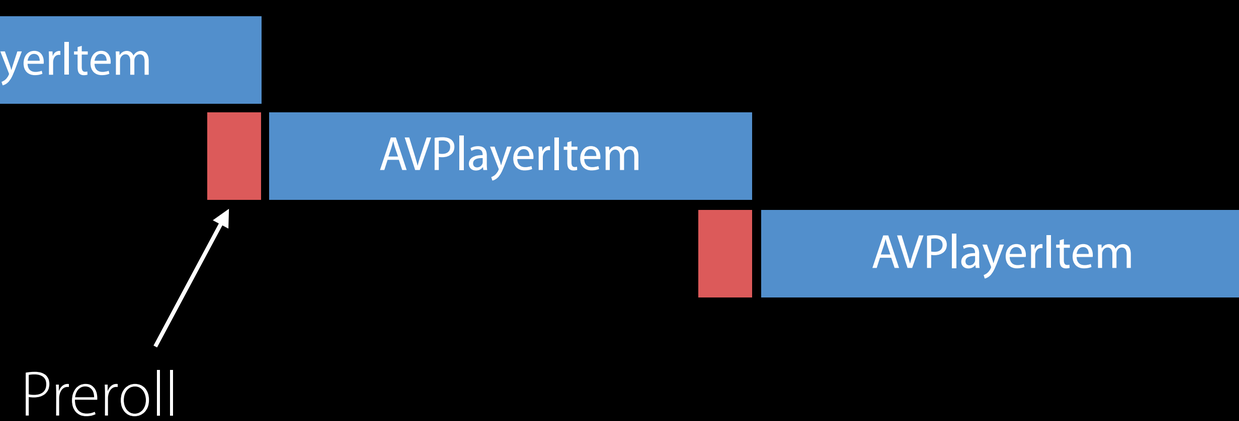
How Do You Loop an AVPlayerItem?

When end reached, rewind?

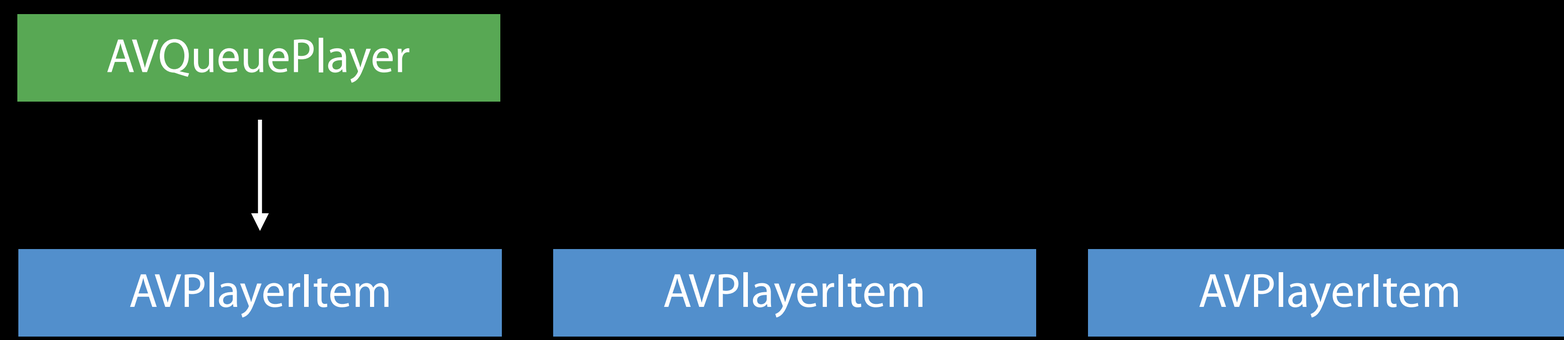


How Do You Loop an AVPlayerItem?

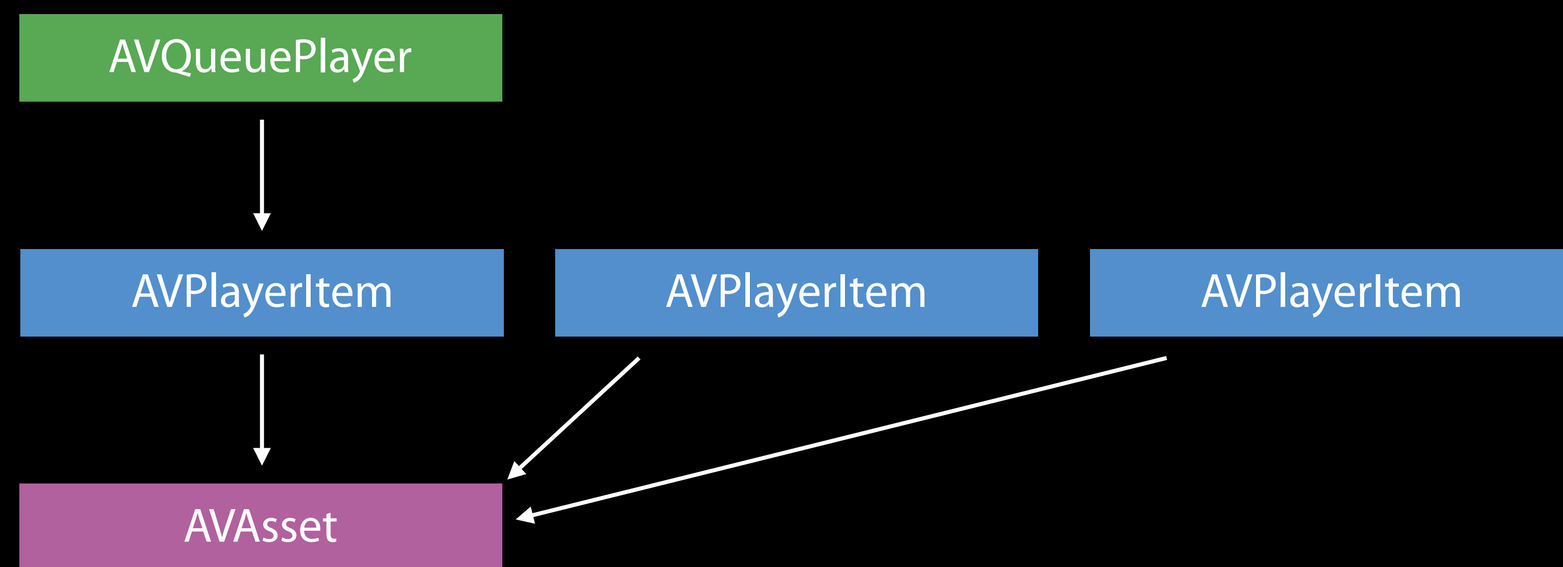
When end reached, rewind?



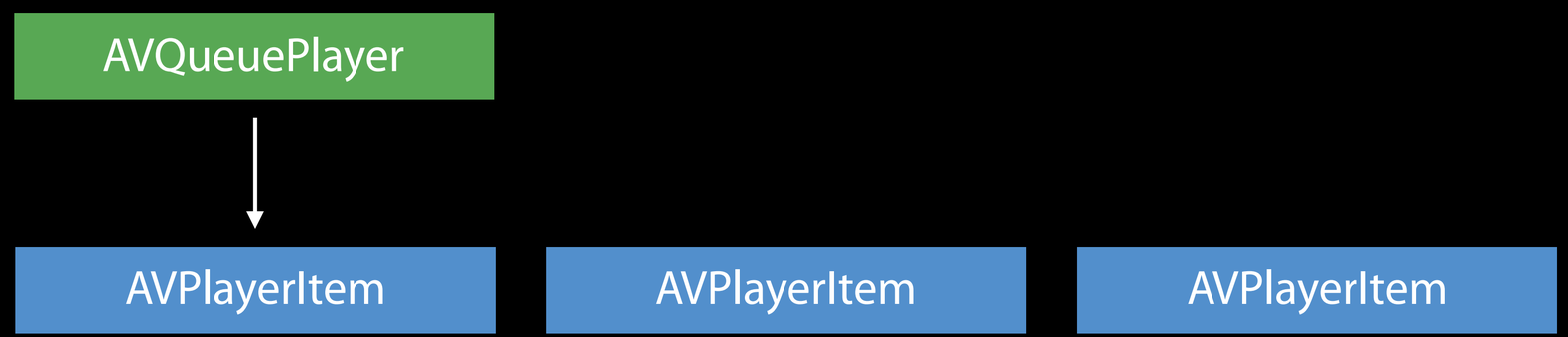
AVQueuePlayer



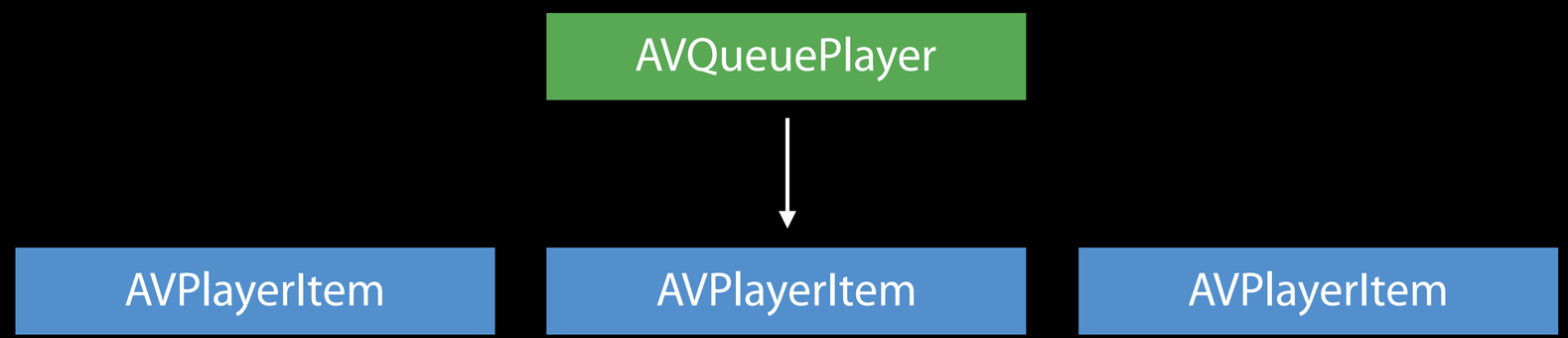
AVQueuePlayer



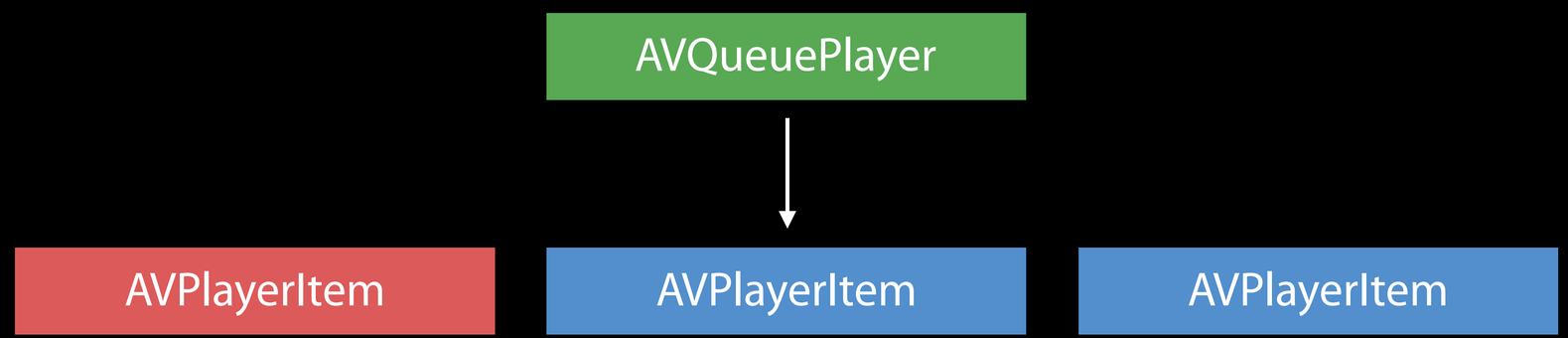
The "Treadmill"



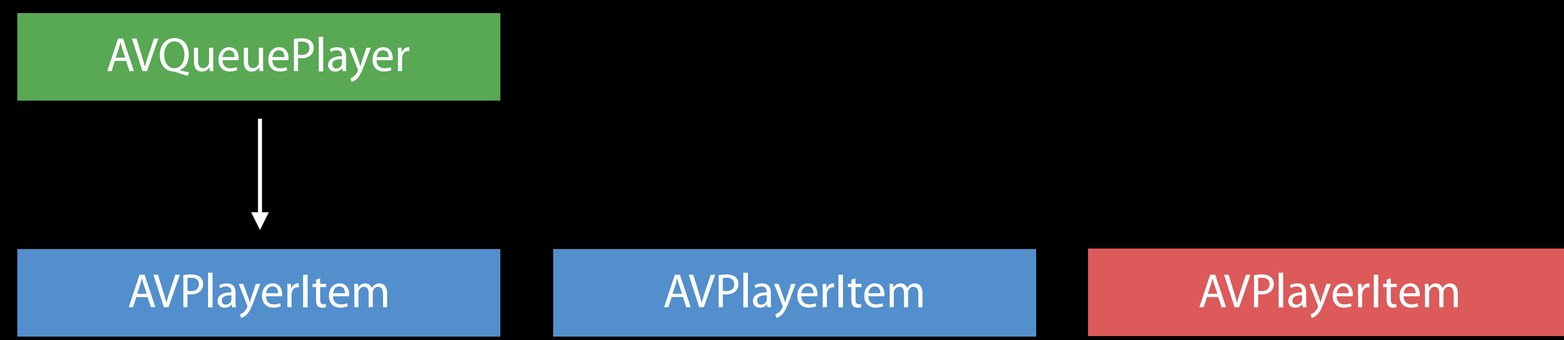
The "Treadmill"



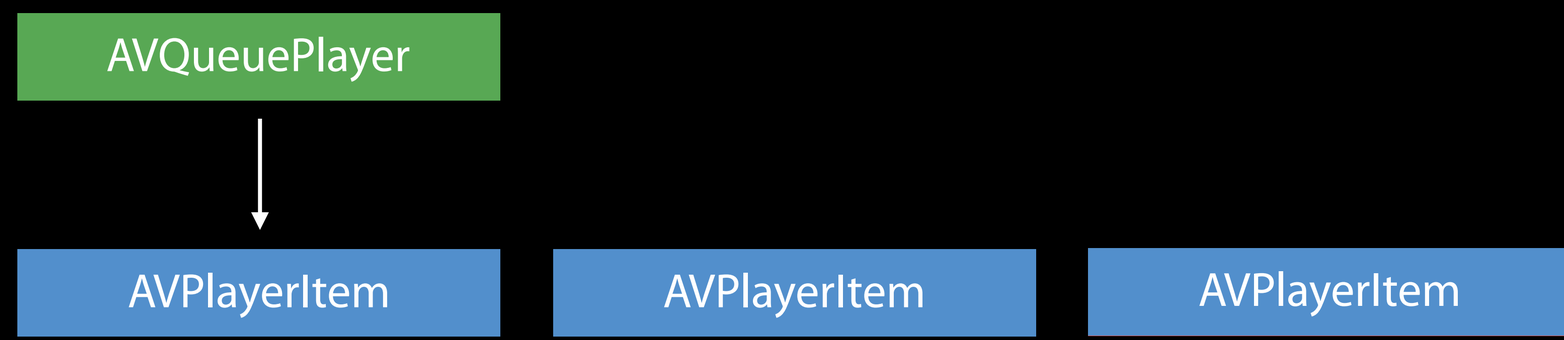
The "Treadmill"



The "Treadmill"



The "Treadmill"



```
// Looping using AVQueuePlayer
override func observeValue(forKeyPath keyPath: String?, of object: AnyObject?, change:
[NSKeyValueChangeKey : AnyObject]?, context: UnsafeMutablePointer<Void>?) {
    if context == &ObserverContexts.currentItem {
        guard let player = player else { return }
        if player.items().isEmpty {
            // Play queue emptied out due to bad player item. End looping.
        }
        else {
            if let itemRemoved = change?[.oldKey] as? AVPlayerItem {
                itemRemoved.seek(to: kCMTimeZero)
                stopObserving()
                player.insert(itemRemoved, after: nil)
                startObserving()
            }
        }
    }
    // else ...
}
```

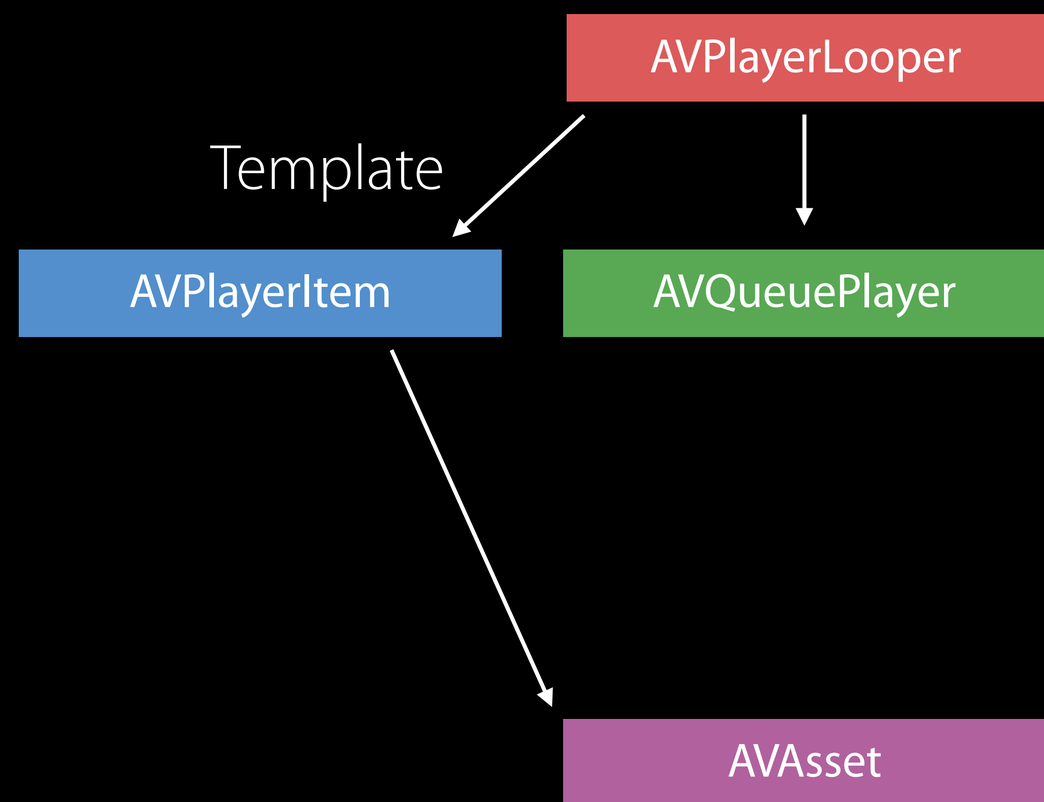
```
// Looping using AVQueuePlayer
override func observeValue(forKeyPath keyPath: String?, of object: AnyObject?, change:
[NSKeyValueChangeKey : AnyObject]?, context: UnsafeMutablePointer<Void>?) {
    if context == &ObserverContexts.currentItem {
        guard let player = player else { return }
        if player.items().isEmpty {
            // Play queue emptied out due to bad player item. End looping.
        }
        else {
            if let itemRemoved = change?[.oldKey] as? AVPlayerItem {
                itemRemoved.seek(to: kCMTimeZero)
                stopObserving()
                player.insert(itemRemoved, after: nil)
                startObserving()
            }
        }
    }
    // else ...
}
```

```
// Looping using AVQueuePlayer
override func observeValue(forKeyPath keyPath: String?, of object: AnyObject?, change:
[NSKeyValueChangeKey : AnyObject]?, context: UnsafeMutablePointer<Void>?) {
    if context == &ObserverContexts.currentItem {
        guard let player = player else { return }
        if player.items().isEmpty {
            // Play queue emptied out due to bad player item. End looping.
        }
        else {
            if let itemRemoved = change?[.oldKey] as? AVPlayerItem {
                itemRemoved.seek(to: kCMTimeZero)
                stopObserving()
                player.insert(itemRemoved, after: nil)
                startObserving()
            }
        }
    }
    // else ...
}
```

```
// Looping using AVQueuePlayer
override func observeValue(forKeyPath keyPath: String?, of object: AnyObject?, change:
[NSKeyValueChangeKey : AnyObject]?, context: UnsafeMutablePointer<Void>?) {
    if context == &ObserverContexts.currentItem {
        guard let player = player else { return }
        if player.items().isEmpty {
            // Play queue emptied out due to bad player item. End looping.
        }
        else {
            if let itemRemoved = change?[.oldKey] as? AVPlayerItem {
                itemRemoved.seek(to: kCMTimeZero)
                stopObserving()
                player.insert(itemRemoved, after: nil)
                startObserving()
            }
        }
    }
    // else ...
}
```

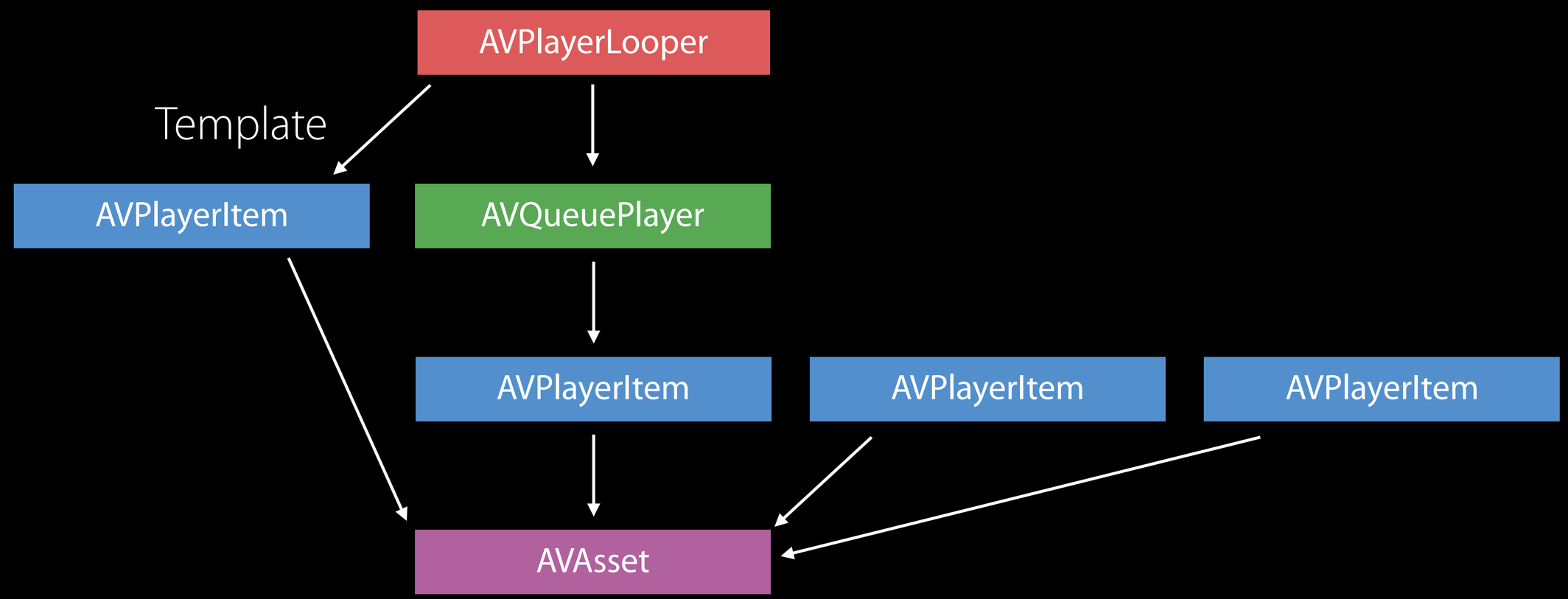
AVPlayerLooper

NEW



AVPlayerLooper

NEW



AVPlayerLooper

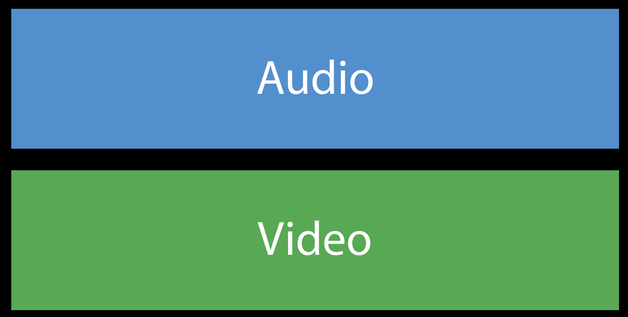
```
player = AVQueuePlayer()
playerLayer = AVPlayerLayer(player: player)
playerItem = AVPlayerItem(url: videoURL)
playerLooper = AVPlayerLooper(player: player, templateItem: playerItem)
player.play()
```

Demo

AVPlayerLooper

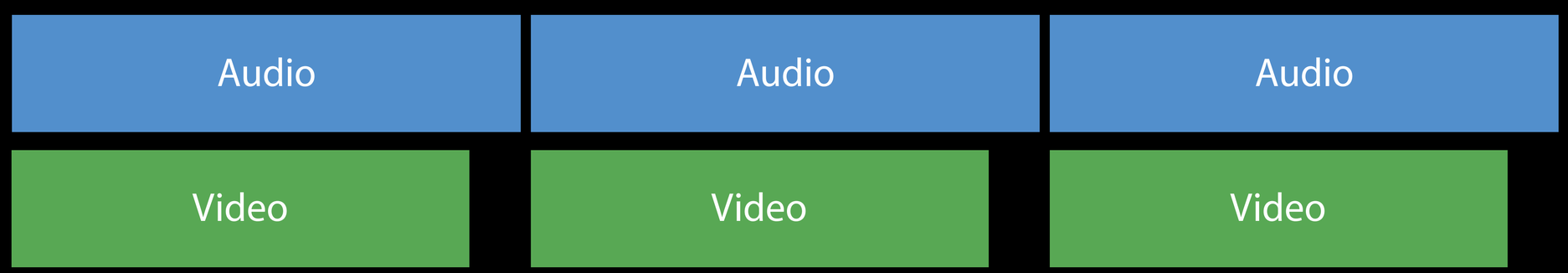
Optimizing Movies for Looping

Make sure audio and video tracks are same length



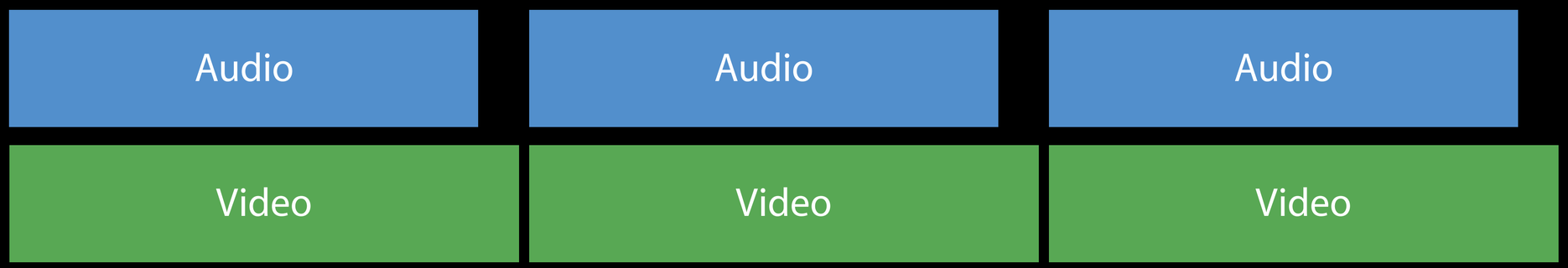
Optimizing Movies for Looping

Make sure audio and video tracks are same length



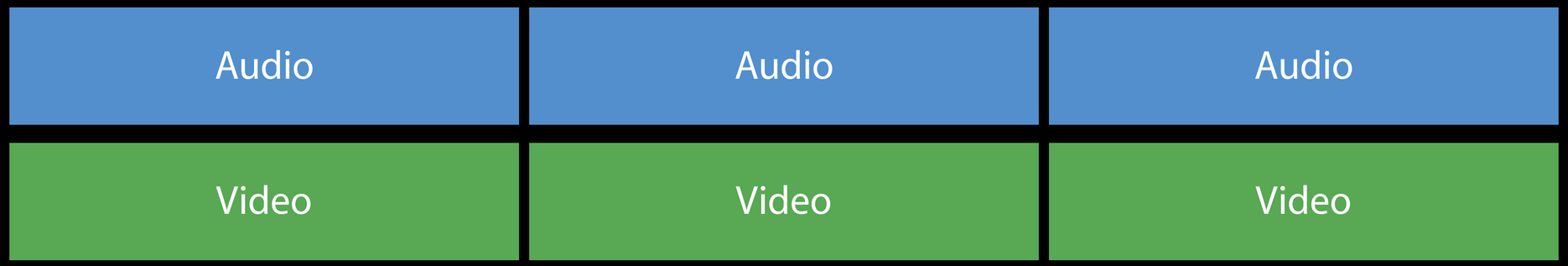
Optimizing Movies for Looping

Make sure audio and video tracks are same length



Optimizing Movies for Looping

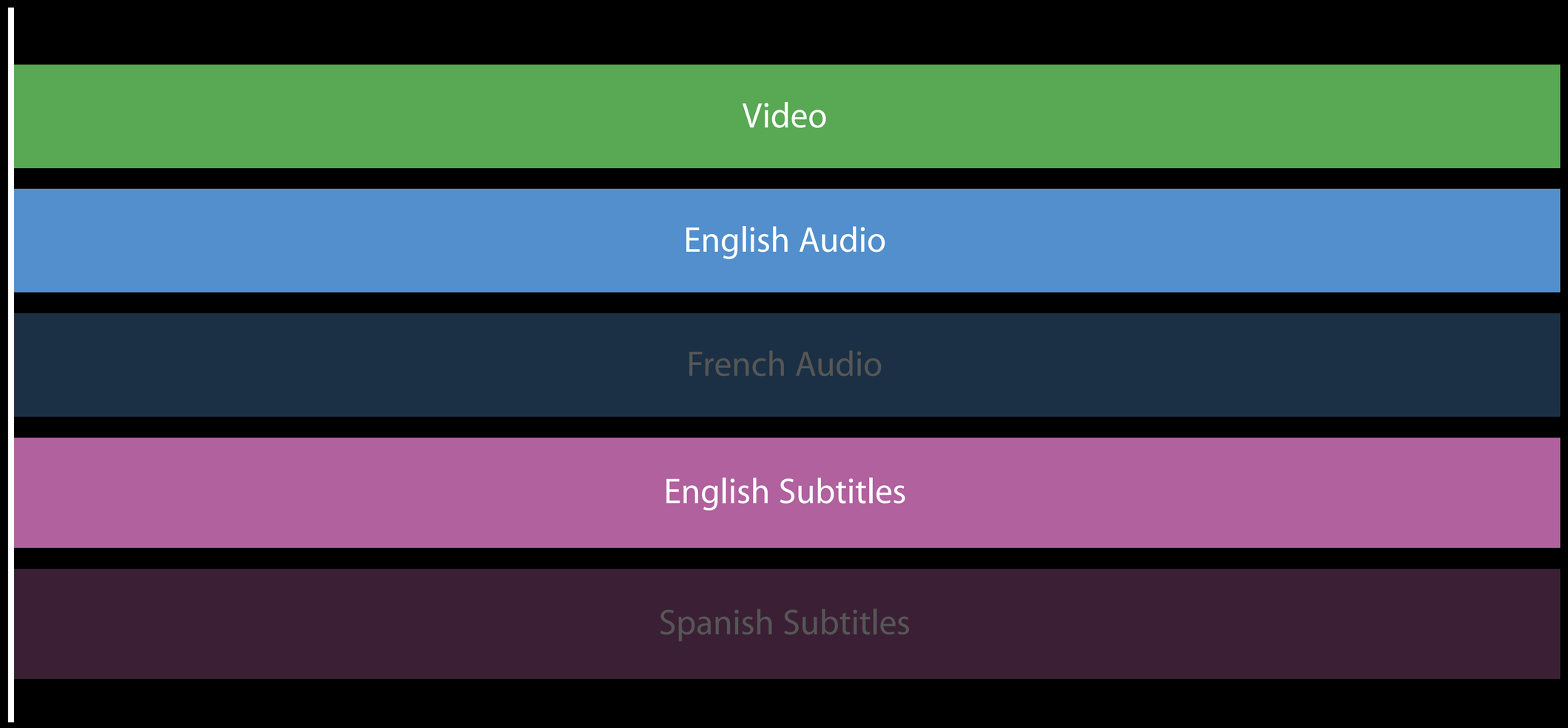
Make sure audio and video tracks are same length



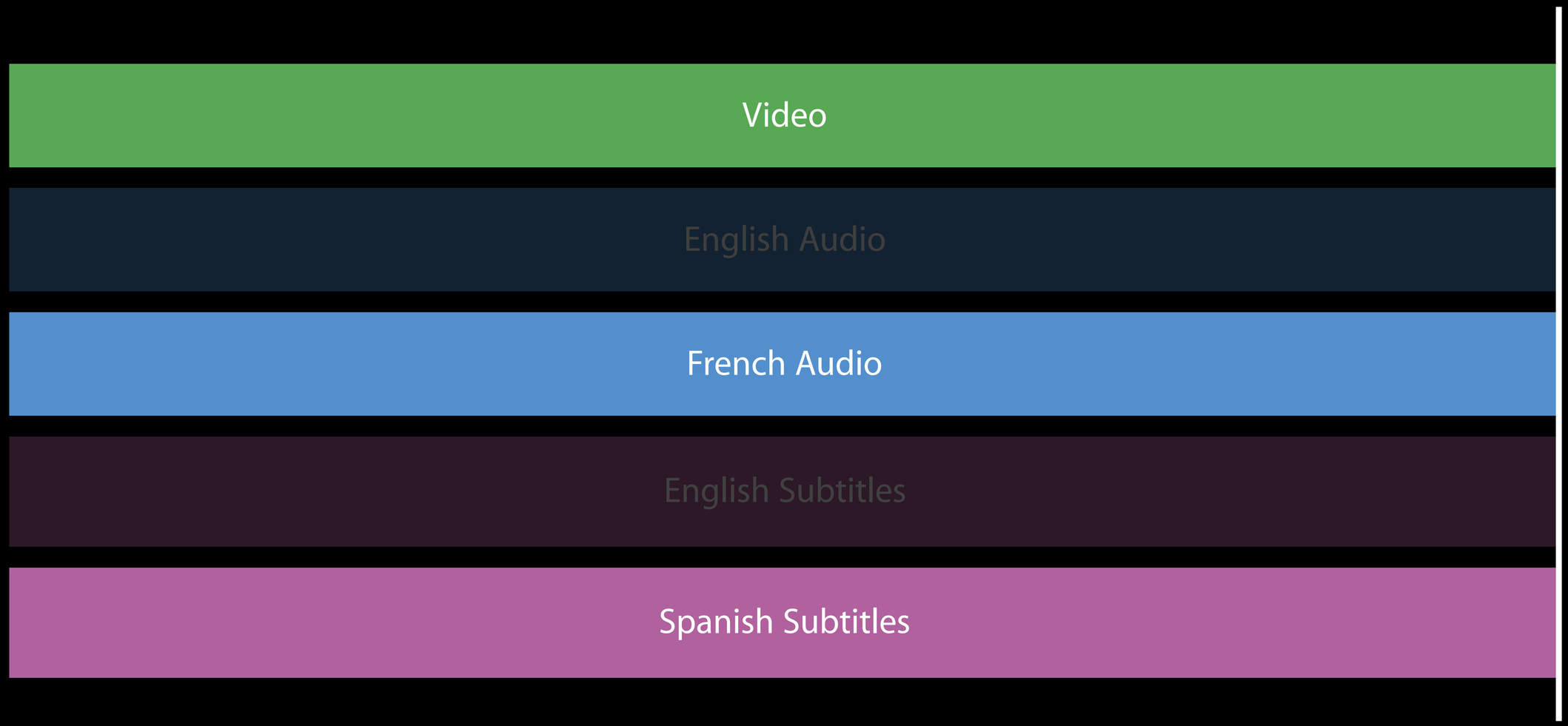
Playback Refinements

Under the hood

When Tracks Come and Go During Playback



When Tracks Come and Go During Playback



Some More Smoothness

Where once there were glitches

- Adding / Removing the only AVPlayerLayer on playing AVPlayer
- Changing subtitle language on playing AVPlayer
- Changing audio language on playing AVPlayer
- Manually disabling / enabling tracks on playing AVPlayer

Preparing for Wide Color Video

Color Space Tagging in Media Files

Color space information is part of the metadata of video tracks

	SD	HD	P3 D65
Color Primaries	Rec. 601	Rec. 709	P3 D65
Transfer Characteristics	Rec. 709	Rec. 709	Rec. 709
Y'CbCr Matrix	Rec. 601	Rec. 709	Rec. 709

Standard tag numbers defined in ISO/IEC 23001-8, "Coding Independent Code Points"

Detecting Wide Color Tags

NEW

Use `AVMediaCharacteristicUsesWideGamutColorSpace`

```
let wideGamutTracks = asset.tracks(
   withMediaCharacteristic:AVMediaCharacteristicUsesWideGamutColorSpace)

if wideGamutTracks.count > 0 {
   // use wide color aware processing
}
else {
   // use Rec 709 processing
}
```

Specifying Working Color Space

NEW

`AVPlayerItemVideoOutput, AVAssetReaderOutput` and `AVAssetWriterInput` now support color space specification in `outputSettings`

```
let exampleSettings =
   [AVVideoColorPropertiesKey:
      [AVVideoColorPrimariesKey:   AVVideoColorPrimaries_P3_D65,
       AVVideoTransferFunctionKey: AVVideoTransferFunction_ITU_R_709_2,
       AVVideoYCbCrMatrixKey:      AVVideoYCbCrMatrix_ITU_R_709_2]]
let videoOutput = AVPlayerItemVideoOutput(outputSettings:exampleSettings)
let readerOutput = AVAssetReaderOutput(outputSettings:exampleSettings)
let writerInput = AVAssetWriterInput(mediaType:AVMediaTypeVideo, outputSettings:exampleSettings)
```

Preserving Wide Color Space

NEW

Alternatively, ask `AVPlayerItemVideoOutput` or `AVAssetReaderOutput` to keep buffers in original color space via `AVVideoAllowWideColorKey`

```
let allowWideColorSettings = [AVVideoAllowWideColorKey:true]
let videoOutput = AVPlayerItemVideoOutput(outputSettings:allowWideColorSettings)
let readerOutput = AVAssetReaderOutput(outputSettings:allowWideColorSettings)
```

Video Composition

Specifying working color space

NEW

```
let videoComposition = AVMutableVideoComposition()
videoComposition.colorPrimaries        = AVVideoColorPrimaries_P3_D65
videoComposition.colorTransferFunction = AVVideoTransferFunction_ITU_R_709_2
videoComposition.colorYCbCrMatrix      = AVVideoYCbCrMatrix_ITU_R_709_2
```

Custom Video Compositor

Declaring wide color awareness

NEW

```
class MyCustomVideoCompositor : AVVideoCompositing {
   // ...
   var supportsWideColorSourceFrames: Boolean  { return true }
}
```

Explicitly Tagging Buffers

If you generate source buffers for rendering, you may need to tag them

```
CVBufferSetAttachment(pixelBuffer, kCVImageBufferColorPrimariesKey,
   kCVImageBufferColorPrimaries_P3_D65, kCVAttachmentMode_ShouldPropagate)
CVBufferSetAttachment(pixelBuffer, kCVImageBufferTransferFunctionKey,
   kCVImageBufferTransferFunction_ITU_R_709_2, kCVAttachmentMode_ShouldPropagate)
CVBufferSetAttachment(pixelBuffer, kCVImageBufferYCbCrMatrixKey,
   kCVImageBufferYCbCrMatrix_ITU_R_709_2, kCVAttachmentMode_ShouldPropagate)
writerAdaptor.append(pixelBuffer, withPresentationTime: PTS)
```

Best Practices for Playback

How can I make my videos start as fast as possible?

Speeding Up Local File Playback

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer(playerItem: playerItem)
let playerLayer = AVPlayerLayer(player: player)
```

Speeding Up Local File Playback

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer(playerItem: playerItem)
let playerLayer = AVPlayerLayer(player: player)
```

1. set up audio-only playback

Speeding Up Local File Playback

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer(playerItem: playerItem)
let playerLayer = AVPlayerLayer(player: player)
```

1. set up audio-only playback

2. set up audio+video playback

Order Matters

Ask for the final goal first

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player)
player.replaceCurrentItemWithPlayerItem(playerItem)
```

Order Matters

Ask for the final goal first

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player)
player.replaceCurrentItemWithPlayerItem(playerItem)
```

Order Matters

Ask for the final goal first

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player)
player.replaceCurrentItemWithPlayerItem(playerItem)
```

Order Matters

Ask for the final goal first

```
let asset = AVURLAsset(url: url)
let playerItem = AVPlayerItem(asset: asset)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player)
player.replaceCurrentItemWithPlayerItem(playerItem)
```

1. set up audio+video playback

Best Practice

Configure `AVPlayer` and `AVPlayerItem` first

Connect `AVPlayerLayer` to `AVPlayer`,
or `AVPlayerItemVideoOutput` to `AVPlayerItem`

```
player.play()
```

```
player.replaceCurrentItemWithPlayerItem(playerItem)
```

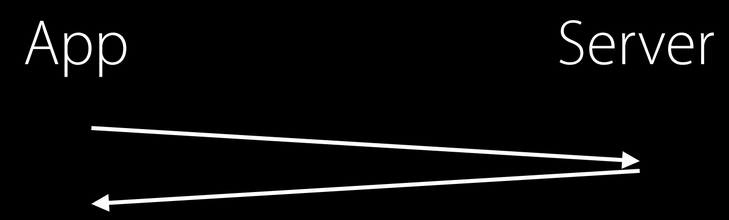
Speeding Up HTTP Live Streaming

Consider the network round-trips

Speeding Up HTTP Live Streaming

Consider the network round-trips

Retrieve master playlist

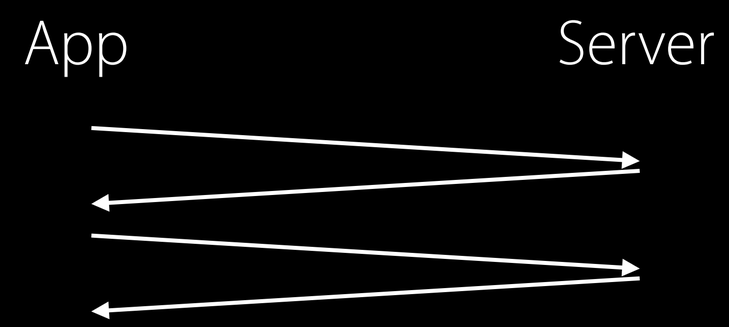


Speeding Up HTTP Live Streaming

Consider the network round-trips

Retrieve master playlist

Retrieve content keys



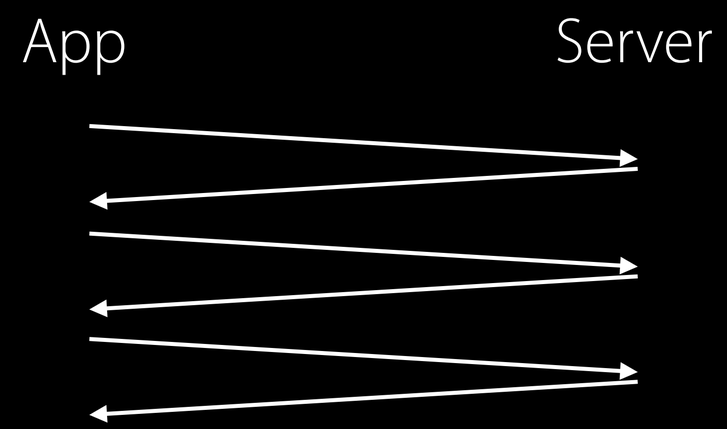
Speeding Up HTTP Live Streaming

Consider the network round-trips

Retrieve master playlist

Retrieve content keys

Retrieve selected variant playlist



Speeding Up HTTP Live Streaming

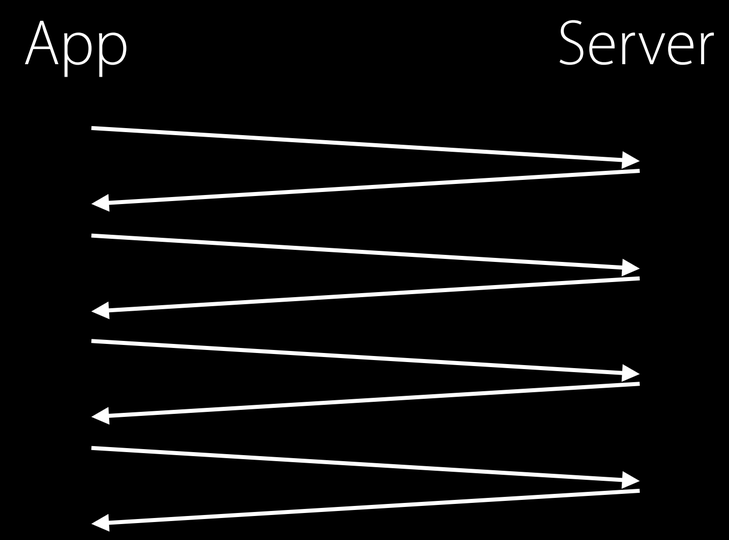
Consider the network round-trips

Retrieve master playlist

Retrieve content keys

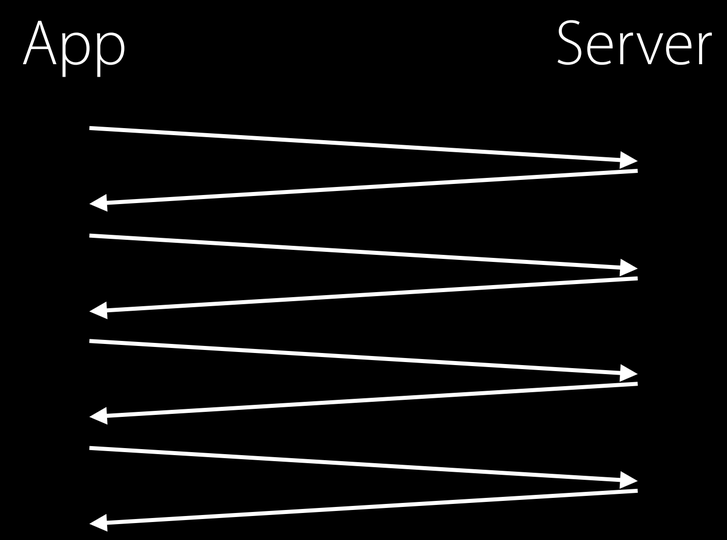
Retrieve selected variant playlist

Retrieve segments



Speeding Up HTTP Live Streaming

Consider the network round-trips



Retrieve master playlist

Retrieve content keys

Retrieve selected variant playlist

Retrieve segments

Can you do some of these before the user hits "play"?

Speeding Up HTTP Live Streaming

Preloading the Master Playlist

```
var asset = AVURLAsset(url: url)
asset.loadValuesAsynchronously(forKeys: ["duration"], completionHandler: nil)
```

Speeding Up HTTP Live Streaming

Compress Playlists

Compress Master Playlists and Variant Playlists with gzip

- Your server may be able to do this for you

Speeding Up FairPlay Streaming Startup

Initiate key exchange earlier

```
var asset = AVURLAsset(url: url)
asset.resourceLoader.preloadsEligibleContentKeys = true
```

Master playlist must contain SESSION-KEY declarations

Speeding Up HTTP Live Streaming

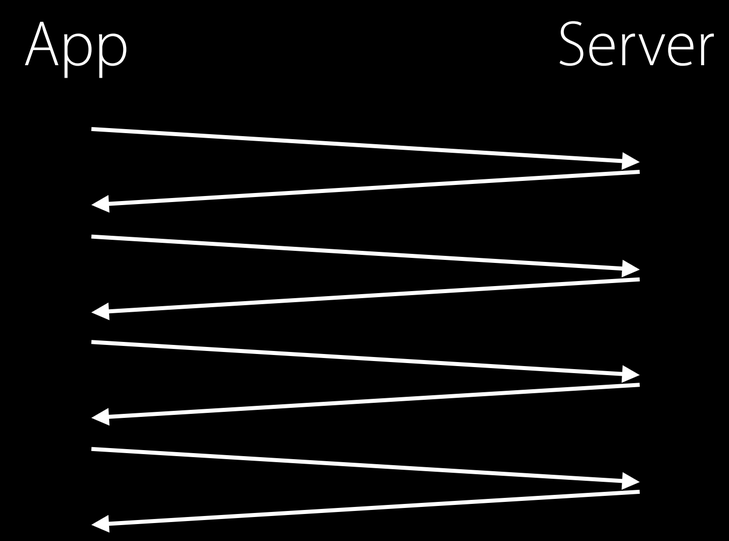
What else can we do?

Retrieve master playlist

Retrieve content keys

Retrieve selected variant playlist

Retrieve segments



Speeding Up HTTP Live Streaming

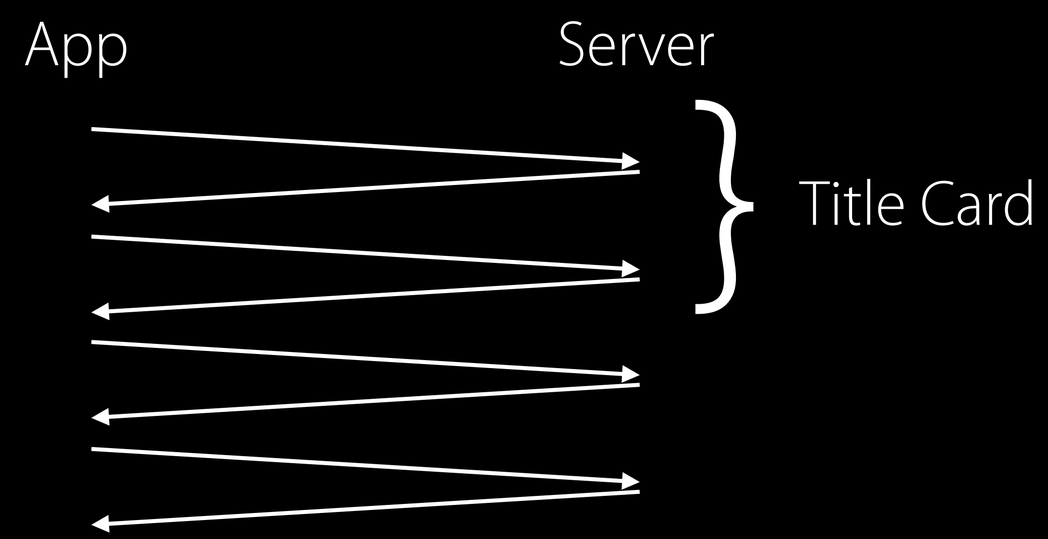
What else can we do?

Retrieve master playlist

Retrieve content keys

Retrieve selected variant playlist

Retrieve segments



Speeding Up HTTP Live Streaming

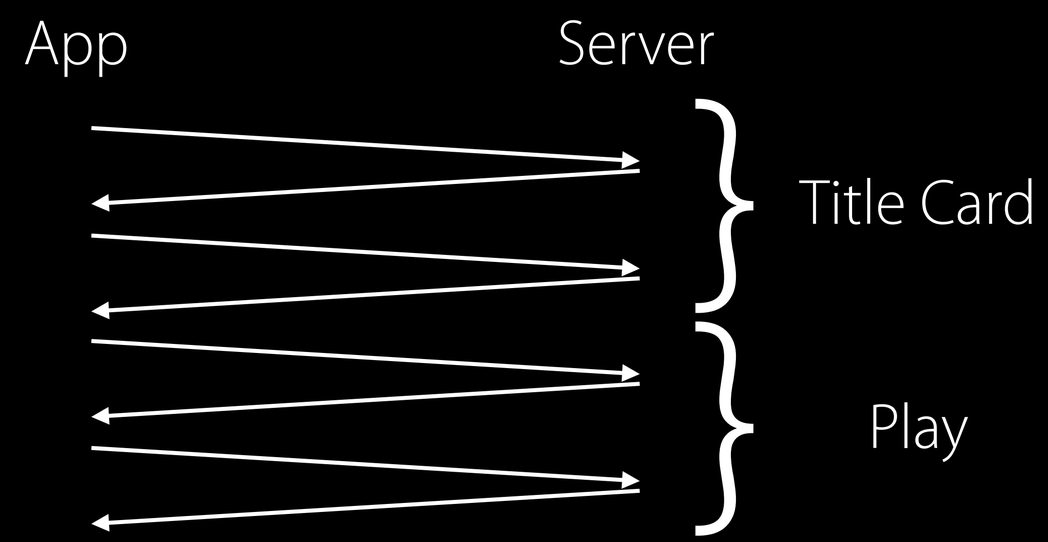
What else can we do?

Retrieve master playlist

Retrieve content keys

Retrieve selected variant playlist

Retrieve segments



Speeding Up HTTP Live Streaming

NEW

Preload segments before playback

```
// on title card
var playerItem = AVPlayerItem(asset: asset)
playerItem.preferredForwardBufferDuration = CMTime(value: 5, timescale: 1)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player) // keep the layer hidden
player.replaceCurrentItemWithPlayerItem(playerItem)
```

```
// as soon as playback begins, reset it to default
playerItem.preferredForwardBufferDuration = kCMTimeZero
```

Speeding Up HTTP Live Streaming

NEW

Preload segments before playback

```
// on title card
var playerItem = AVPlayerItem(asset: asset)
playerItem.preferredForwardBufferDuration = CMTime(value: 5, timescale: 1)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player) // keep the layer hidden
player.replaceCurrentItemWithPlayerItem(playerItem)
```

```
// as soon as playback begins, reset it to default
playerItem.preferredForwardBufferDuration = kCMTimeZero
```

Speeding Up HTTP Live Streaming

NEW

Preload segments before playback

```
// on title card
var playerItem = AVPlayerItem(asset: asset)
playerItem.preferredForwardBufferDuration = CMTime(value: 5, timescale: 1)
let player = AVPlayer()
let playerLayer = AVPlayerLayer(player: player) // keep the layer hidden
player.replaceCurrentItemWithPlayerItem(playerItem)
```

```
// as soon as playback begins, reset it to default
playerItem.preferredForwardBufferDuration = kCMTimeZero
```

Improving Initial Quality

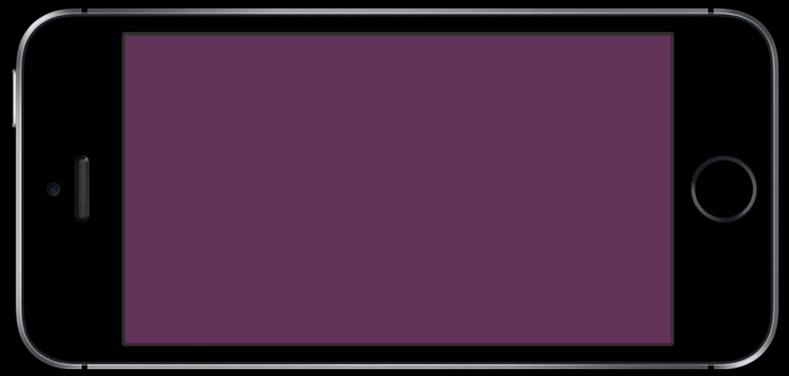
Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Improving Initial Quality

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Improving Initial Quality

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec



Improving Initial Quality

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Improving Initial Quality

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Improving Initial Quality

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

2G

Improving Initial Quality

AVPlayerLayer

Size your AVPlayerLayer appropriately and connect it to AVPlayer early

- Before bringing in playerItem

Set `AVPlayerLayer.contentsScale` on retina iOS devices

Improving Initial Quality

Control initial variant selection

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Improving Initial Quality

Control initial variant selection

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

?

Improving Initial Quality

Control initial variant selection

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Improving Initial Quality

Buffering time to start playback (10-second segments)

Dimensions	Video bitrate	Buffering time @ 2 Mbit/sec
400 x 224	110 kbit/sec	0.55 sec
400 x 224	400 kbit/sec	2 sec
640 x 360	600 kbit/sec	3 sec
960 x 540	1800 kbit/sec	9 sec
1280 x 720	4500 kbit/sec	22.5 sec
1920 x 1080	11000 kbit/sec	55 sec

Improving Initial Quality

Use previous playback's statistics

```
if let lastAccessLogEvent = previousPlayerItem.accessLog()?.events.last {
    lastObservedBitrate = lastAccessLogEvent.observedBitrate
}
```

Controlling Initial Variant Selection

Option A: Use preferredPeakBitRate to restrict first choice

Dimensions	Video bitrate
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1280 x 720	4500 kbit/sec
1920 x 1080	11000 kbit/sec

Controlling Initial Variant Selection

Option A: Use preferredPeakBitRate to restrict first choice

Dimensions	Video bitrate
1920 x 1080	11000 kbit/sec
1280 x 720	4500 kbit/sec
960 x 540	1800 kbit/sec
640 x 360	600 kbit/sec
400 x 224	400 kbit/sec
400 x 224	110 kbit/sec

Controlling Initial Variant Selection

Option A: Use preferredPeakBitRate to restrict first choice

Dimensions	Video bitrate
1920 x 1080	11000 kbit/sec
1280 x 720	4500 kbit/sec
960 x 540	1800 kbit/sec
640 x 360	600 kbit/sec
400 x 224	400 kbit/sec
400 x 224	110 kbit/sec

```
// before playback
playerItem.preferredPeakBitRate = 2000

// shortly after playback starts
playerItem.preferredPeakBitRate = 0
```

Controlling Initial Variant Selection

Option A: Use preferredPeakBitRate to restrict first choice

Dimensions	Video bitrate
1920 x 1080	11000 kbit/sec
1280 x 720	4500 kbit/sec
960 x 540	1800 kbit/sec
640 x 360	600 kbit/sec
400 x 224	400 kbit/sec
400 x 224	110 kbit/sec

```
// before playback
playerItem.preferredPeakBitRate = 2000

// shortly after playback starts
playerItem.preferredPeakBitRate = 0
```

Controlling Initial Variant Selection

Option A: Use preferredPeakBitRate to restrict first choice

Dimensions	Video bitrate
1920 x 1080	11000 kbit/sec
1280 x 720	4500 kbit/sec
960 x 540	1800 kbit/sec
640 x 360	600 kbit/sec
400 x 224	400 kbit/sec
400 x 224	110 kbit/sec

```
// before playback
playerItem.preferredPeakBitRate = 2000

// shortly after playback starts
playerItem.preferredPeakBitRate = 0
```

Controlling Initial Variant Selection

Option B: Dynamically rewrite master playlist

Dimensions	Video bitrate
1280 x 720	4500 kbit/sec
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1920 x 1080	11000 kbit/sec

Controlling Initial Variant Selection

Option B: Dynamically rewrite master playlist

Dimensions	Video bitrate
1280 x 720	4500 kbit/sec
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1920 x 1080	11000 kbit/sec

```
var asset = AVURLAsset(url:
    NSURL(string: "myscheme://file.m3u8")!)

asset.resourceLoader.setDelegate(...)
```

Controlling Initial Variant Selection

Option B: Dynamically rewrite master playlist

Dimensions	Video bitrate
1280 x 720	4500 kbit/sec
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1920 x 1080	11000 kbit/sec

```
var asset = AVURLAsset(url:
    NSURL(string: "myscheme://file.m3u8")!)

asset.resourceLoader.setDelegate(...)
```

Controlling Initial Variant Selection

Option B: Dynamically rewrite master playlist

Dimensions	Video bitrate
1280 x 720	4500 kbit/sec
400 x 224	110 kbit/sec
400 x 224	400 kbit/sec
640 x 360	600 kbit/sec
960 x 540	1800 kbit/sec
1920 x 1080	11000 kbit/sec

```
var asset = AVURLAsset(url:
    NSURL(string: "myscheme://file.m3u8")!)

asset.resourceLoader.setDelegate(...)
```

Profile Your Code Too

Profile Your Code Too

Look for delays in your code, before AVFoundation is called

Profile Your Code Too

Look for delays in your code, before AVFoundation is called

Don't wait for likelyToKeepUp notification before setting rate

Profile Your Code Too

Look for delays in your code, before AVFoundation is called

Don't wait for likelyToKeepUp notification before setting rate

Make sure you release AVPlayers and AVPlayerItems from old playback sessions

Profile Your Code Too

Look for delays in your code, before AVFoundation is called

Don't wait for likelyToKeepUp notification before setting rate

Make sure you release AVPlayers and AVPlayerItems from old playback sessions

Use Allocations Instrument to check AVPlayer and AVPlayerItem lifespans

Profile Your Code Too

Look for delays in your code, before AVFoundation is called

Don't wait for likelyToKeepUp notification before setting rate

Make sure you release AVPlayers and AVPlayerItems from old playback sessions

Use Allocations Instrument to check AVPlayer and AVPlayerItem lifespans

Suspend other network activity in your app during network playback

Summary

Summary

automaticallyWaitsToMinimizeStalling

Summary

automaticallyWaitsToMinimizeStalling

AVPlayerLooper

Summary

automaticallyWaitsToMinimizeStalling

AVPlayerLooper

Enabling and disabling tracks during playback is smoother

Summary

automaticallyWaitsToMinimizeStalling

AVPlayerLooper

Enabling and disabling tracks during playback is smoother

Prepare for wide color video

Summary

automaticallyWaitsToMinimizeStalling

AVPlayerLooper

Enabling and disabling tracks during playback is smoother

Prepare for wide color video

Optimize playback startup through cunning and measurement

More Information

https://developer.apple.com/wwdc16/503

Related Sessions

Advances in iOS Photography	Pacific Heights	Tuesday 11:00AM
What's New in HTTP Live Streaming	Mission	Wednesday 3:00PM
Working with Wide Color	Mission	Thursday 1:40PM
AVKit on tvOS	Presidio	Friday 11:00AM
HTTP Live Streaming Authoring and Validation	Video	Watch on Demand

Labs

AVFoundation / AVKit Lab	Graphics, Games, and Media Lab C	Wednesday 9:00AM
AVFoundation / AVKit Lab	Graphics, Games, and Media Lab C	Wednesday 1:00PM
HTTP Live Streaming Lab	Graphics, Games, and Media Lab C	Wednesday 4:00PM
Photo Capture Lab	Graphics, Games, and Media Lab C	Thursday 9:00AM
AVFoundation / HTTP Live Steaming Lab	Graphics, Games, and Media Lab D	Thursday 9:00AM
AVKit Lab	Graphics, Games, and Media Lab C	Friday 1:00PM

WWDC16