Using AVFoundation to merge and overlay multiple videos

Ashwin Pokharel Follow 5 min read · Sep 20, 2021

Listen Save

Photo by Sora Sagano on Unsplash

I wanted to accomplish one simple task (feeling one video stream on top of another) using Apple's AVFoundation. I thought this would be straightforward, this would be an easy task. What followed was a long journey into the world of video composition with lots of confusing pitfalls. I hope this guide will help make that journey just a bit easier for you.

Before we begin, I am assuming that you are familiar with AVFoundation as well as Core Animation within Swift. AVFoundation gives a more granular control over audio/video tracks while still retaining a high level of abstraction. This illustration below shows the increasing level of granularity of different frameworks for AV manipulation.

Image Photoshop credit Wikimedia Commons at https://commons.wikimedia.org/wiki/File:OSX_Layers.svg

With these things out of the way let's begin the descent into madness.

AVFoundation basics

There are many different parts of AVFoundation. I will only go over the ones relevant to this guide but you can find more here.

AVAsset

- It is the primary way of representing media.
- It is format agnostic.
- Each asset consists of multiple tracks (audio/ video).

AVComposition

- It is just a collection of tracks that are each presented over a given time range.
- The tracks are AVCompositionTracks that can represent both audio and video.
- AVMutableComposition allows you to add or remove tracks. It also allows you to edit the time range for each track.

AVMutableVideoComposition

- This describes instructions on how to use the given video tracks to generate a video frame over a given time range.
- You can provide instructions for the composition using a AVVideoCompositionInstruction.
- If you use the built in AVVideoComposition you can specify spatial transformations for individual video tracks using AVVideoCompositionLayerInstruction.
- You can also define your own custom compositor by implementing the AVVideoCompositing protocol. (This is what we will be doing)

customVideoCompositionClass

- This allows you to create your own custom instructions for the AVVideoCompositing.

Adobe Premiere Spotlight

- These instructions allow you to define tasks such as Timing among others.

AVVideoCompositing

- This allows you to define your own custom compositor to get more granular control for how the frames are rendered.

AVAssetExportSession

- This object will be responsible for the actual process of transcoding the content of your asset into the final product.
- It allows you to choose the file type you want to export your video out to as well as the frame rate and quality.

With the basics out of the way, let's get to the good stuff. Keep in mind the following code is very barebones with little to no error handling. For your own project you will want to handle possible exceptions more gracefully.

. . .

Code Setup

1. Setting up the composition

Initially we will want to load the two assets , and create an AVMutableComposition. Then for each of the assets, we will add the audio and video tracks separately to the mutable composition.

```
//Instantiate and unwrapped(with exc and) a video local or remote
var firstAsset = AVAsset(url:firstURL)!          //load the first video as an asset
var secondAsset = AVAsset(url:secondURL)!        //load second video as an asset
var composition = AVMutableComposition()         //create a mutable composition

guard
let firstTrack = composition.addMutableTrack(withMediaType: .video,
                                             preferredTrackID: kCMPersistentTrackID_Invalid)
else
//handle error
return

guard
let secondTrack = composition.addMutableTrack(withMediaType: .video,
                                              preferredTrackID: kCMPersistentTrackID_Invalid)
else
//handle error
return

let timeRange = CMTimeRange(start: zero, duration: firstAsset.duration)
    //the range that determines the position of the frames presented
    //in the composition based on time.
    //duration: how long the frames will be shown for
//insert the given track and assetTrack when it would begin playing in the composition
try
insertTimeRange(timeRange, of: firstAsset!.tracks(withMediaType: .video),
                atInsertionTime: CMTime(seconds:0, preferredTimescale:1))
catch
//handle error

let timeRange = CMTimeRange(start: zero, duration: secondAsset.duration)
let firstTimeRange = CMTimeRange(start: firstAsset.duration,
                                 duration: secondAsset.duration)
                                 //at: when to start

try
insertTimeRange(firstTimeRange, of: secondAsset!.tracks(withMediaType: .video),
                atInsertionTime: CMTime(seconds:0, preferredTimescale:1))
catch
//handle error

var firstAudioTrack = composition.addMutableTrack(withMediaType: .audio,
                                                  preferredTrackID: kCMPersistentTrackID_Invalid)
guard
let firstAssetAudioTrack = firstAsset!.tracks(withMediaType: .audio).first
else
//handle error

let timeRange = CMTimeRange(start: zero, duration: firstAsset.duration)
try
insertTimeRange(timeRange, of: firstAssetAudioTrack!, atInsertionTime: zero)
catch
//handle error

var secondAudioTrack = composition.addMutableTrack(withMediaType: .audio,
                                                   preferredTrackID: kCMPersistentTrackID_Invalid)
guard
let secondAssetAudioTrack = secondAsset!.tracks(withMediaType: .audio).first
else
//handle error

let timeRange = CMTimeRange(start: zero, duration: secondAsset.duration)
let secondTimeRange = CMTimeRange(start: firstAsset.duration,
                                  duration: secondAsset.duration)
                                  //at: when to start
try
insertTimeRange(secondTimeRange, of: secondAssetAudioTrack!, atInsertionTime: zero)
catch
//handle error
```

codegists.com/snippet/ios/ios-9-100ms/ view raw

2. Setting up a MutableVideoComposition

Get Ashwin Pokharel's stories in your inbox
Join Medium for free to get updated articles from this writer.
Email address Subscribe

AVMutableVideoComposition allows us to manipulate how frames from different tracks are composited together

```
var videoComposition = AVMutableVideoComposition(propertiesOf: composition, debugDescription: "Video")
var frameDuration = CMTime(seconds: 1/30, preferredTimescale: 600)
var renderSize = CGSize(width: 1920, height: 1080)
var instructions = [AVVideoCompositionInstruction]()
var layerInstructions = [AVVideoCompositionLayerInstruction]()
var mainInstruction = AVMutableVideoCompositionInstruction()
var currentTime = CMTime(seconds: 0, preferredTimescale: 600)
var lastTime = CMTime(seconds: 0, preferredTimescale: 600)
var tmpRange = CMTimeRange(start: currentTime, duration: zero)
var firstLayerInstruction = AVMutableVideoCompositionLayerInstruction(assetTrack: firstTrack!)
var secondLayerInstruction = AVMutableVideoCompositionLayerInstruction(assetTrack: secondTrack!)
// if we are some assets overlaps instructions
```

codegists.com/snippet/ios/ios-9-100ms/ view raw

3. Creating a Custom AVVideoCompositionInstruction

All you need to do to create your own custom instruction is to implement the AVVideoCompositionInstruction protocol. You can choose what instructions you want to pass on to the compositor. In this example I will let my compositor know whether or not a given frame should be visible in the frame.

```
class CustomVideoCompositionInstruction: NSObject, AVVideoCompositionInstruction {
    //This is variables mandat are required to implement the item
    var timeRange: CMTimeRange
    //describes the duration that the given instruction are used
    var enablePostProcessing: Bool = true
    //if it is set to true, the compositor will not be bicubic or well
    var containsTweening: Bool = false
    //if set to false, the compositor will assume that it will be for the
    var requiredSourceTrackIDs: [NSValue]?
    var passthroughTrackID: CMPersistentTrackID = kCMPersistentTrackID_Invalid
    //if set to a valid track id, the compositor will use the track id to decide
    //the compositor must be
    //time range at any given one composition
    //
    //
    //
```

Responses (3)

Write a response

What are your thoughts?

jerry
Nov 9, 2022

why not use two tracks? and play them together?

1 1 reply Reply

Onur Işık
Feb 23, 2022

Awesome article I hope to see similar post on AVfoundation video processing like this , and please do not forget to share entire project link please.

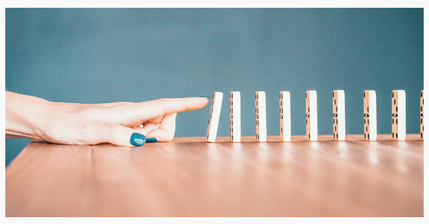
Reply

Shay Gyawali
Oct 9, 2021

so helpful omg

Reply

Recommended from Medium

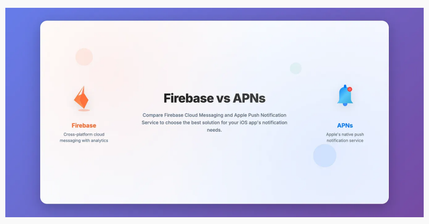


Mushthak Ebrahim

Task Cancellation in Swift Concurrency

Cooperative cancellation, clean-up strategies and task lifecycle management

Jun 18 1

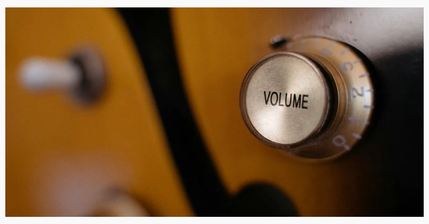


In Stackademic by Himali Marasinghe

Firebase vs APNs: Choosing the Best Push Notification Service for...

Learn the key differences, pricing, limitations, and use cases of Firebase Cloud Messaging...

4d ago 4



Théo Battello

Detect Volume Button Presses in iOS

Quick guide on how to detect volume button presses in every iOS version.

May 20 4

Ashit Ranpura

SwiftUI Layouts Are Smarter Than You Think

Let's dive into the invisible rules behind VStack, HStack, and ZStack

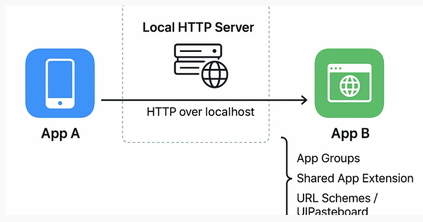
5d ago 16 1

Ravi

Why Every Swift Developer Should Master Curried Functions

Welcome to the world of curried functions—Swift's best-kept secret that 90% of...

Aug 26 24 1



Arpit Kulsreshtha | Architect | Author

Interprocess Communication (IPC) in iOS using Localhost — Part II

Interprocess Communication (IPC) in iOS refers to methods through which different...

Apr 22 1

See more recommendations