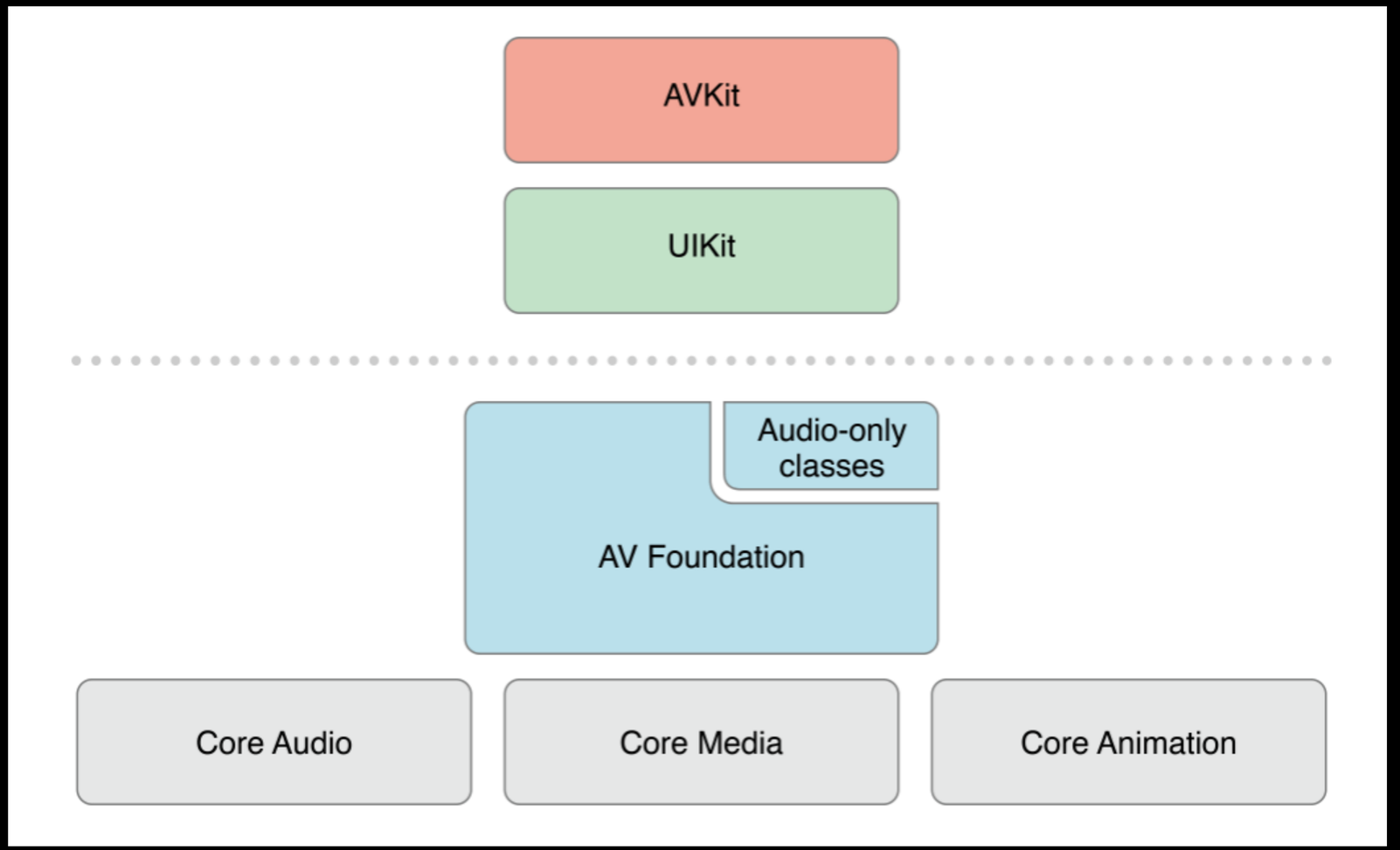
AVFoundation & AVKit

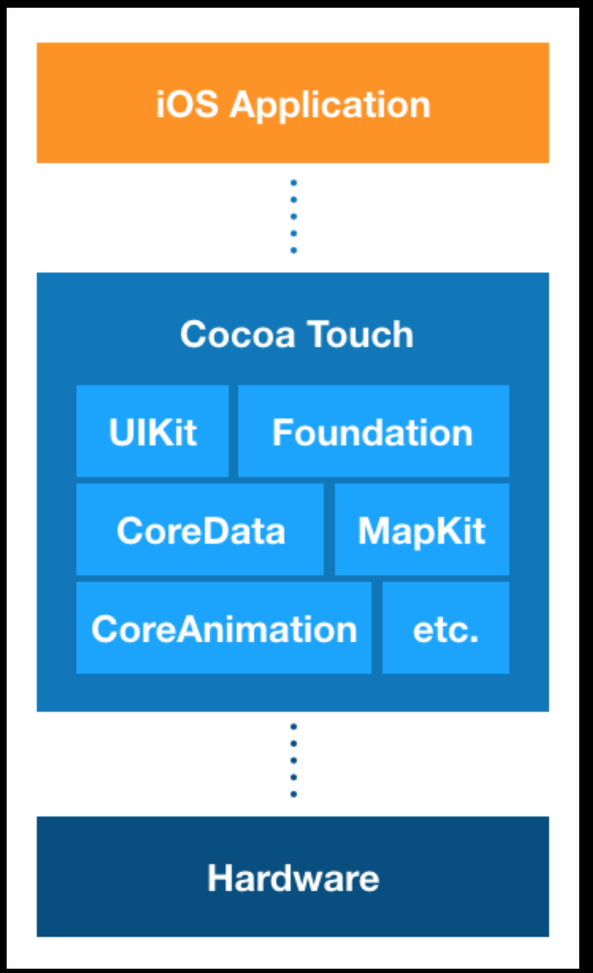
2022.10.16(Mon)

AVFoundation

AVFoundation is the full featured framework for working with time-based audiovisual media on iOS, macOS, watchOS and tvOS. Using AVFoundation, you can easily play, create, and edit QuickTime movies and MPEG-4 files, play HLS streams, and build powerful media functionality into your apps.

- AVFoundation은 시간 기반 시청각 미디어 작업을 위한 프레임워크
- AVFoundation을 사용하면 QuickTime 동영상 및 MPEG-4 파일을 쉽게 검사 / 생성 / 편집 / 재인코딩 - ex. 실시간 캡쳐
- HTTP 라이브 스트리밍 스트림을 재생하고 강력한 미디어 기능을 앱에 구축 가능





Audio-Only Classes :

단순 재생: AVAudioPlayer

오디오를 녹음: AVAudioRecorder

AVAsset

AV Foundation 프레임워크가 미디어를 나타내기 위해 사용하는 기본 클래스

미디어 데이터(오디오 및 비디오 트랙) 컬렉션

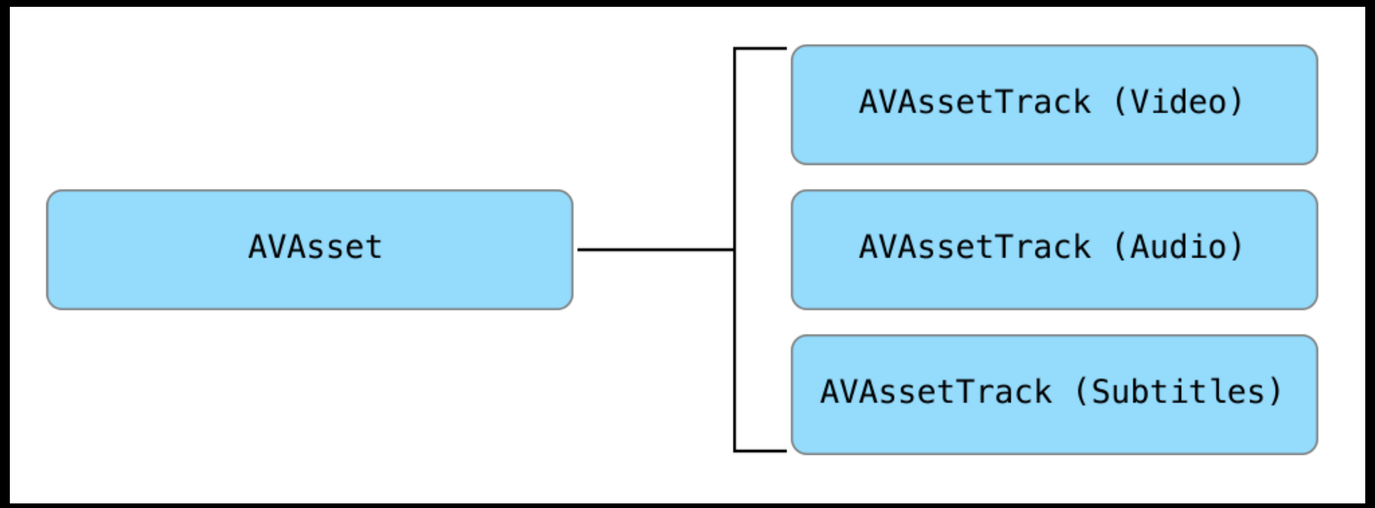
AVAsset: url로 미디어를 객체화하는 역할

전체 미디어에 대한 제목, 기간, 사이즈 등 정보 제공

미디어 파일 포맷을 지원하고 HTTP Live Streaming(HLS)도 지원

AVAsset에 있는 미디어 데이터의 각 부분을 `track` 트랙이라고 정의

AVAsset




```
init(url URL: URL, options: [String : Any]? = nil)
```

AVAsset 인스턴스를 통해 미디어 데이터의 재생 가능 여부, 총 재생 시간, 생성 날짜, 메타 데이터 프로퍼티 사용이 가능

메타데이터는 보통 데이터가 크고 원활한 UI 작업을 위해, 비동기적으로 로드

AVAsset이 생성될 때 프로퍼티들은 자동으로 같이 만들어지지 않고 요청이 오면 불러오기 시작

(AVAsynchronousKeyValueLoading protocol 준수하여 별도로 로드)

```swift
import AVKit
import AVFoundation
import Foundation

// URL of a bundle asset called 'example.mp4'
let url = Bundle.main.url(forResource: "example", withExtension: "mp4")!
let asset = AVAsset(url: url)
let playableKey = "playable"

// Load the "playable" property
asset.loadValuesAsynchronously(forKeys: [playableKey]) {
    var error: NSError? = nil
    let status = asset.statusOfValue(forKey: playableKey, error: &error)
    switch status {
    case .loaded:
        // Sucessfully loaded. Continue processing.
    case .failed:
        // Handle error
    case .cancelled:
        // Terminate processing
    default:
        // Handle all other cases
    }
}
```

```swift
let url = Bundle.main.url(forResource: "audio", withExtension: "m4a")!
let asset = AVAsset(url: url)
let formatsKey = "availableMetadataFormats"
asset.loadValuesAsynchronously(forKeys: [formatsKey]) {
    var error: NSError? = nil
    let status = asset.statusOfValue(forKey: formatsKey, error: &error)
    if status == .loaded {
        for format in asset.availableMetadataFormats {
            let metadata = asset.metadata(forFormat: format)

            // 타이틀 관련 메타데이터 작업
            let titleID = AVMetadataIdentifier.commonIdentifierTitle
            let titleItems = AVMetadataItem.metadataItems(from: metadata, filteredByIdentifier: titleID)
            if let item = titleItems.first {
                // process title item
            }

            // 앨범 아트워크 관련 작업
            let artworkItems =
                AVMetadataItem.metadataItems(
                    from: metadata,
                    filteredByIdentifier: AVMetadataIdentifier.commonIdentifierArtwork
                )
            if let artworkItem = artworkItems.first {
                // dataValue 프로퍼티를 사용하여 값을 NSData로 강제 변환
                if let imageData = artworkItem.dataValue {
                    let image = UIImage(data: imageData)
                    // process image
                } else {
                    // No image data found
                }
            }

        }
    }
}
```

AVKit

AVKit은 AVFoundation을 기반으로 미디어 플레이어 Interface를 쉽게 제공하는 SDK

AVFoundation은 UIKIt 밑단에 있기 때문에 표준화된 UI를 제공하지 않음

UI를 구현하려면, AVFoundation과 low-level 수준의 깊은 지식이 필요

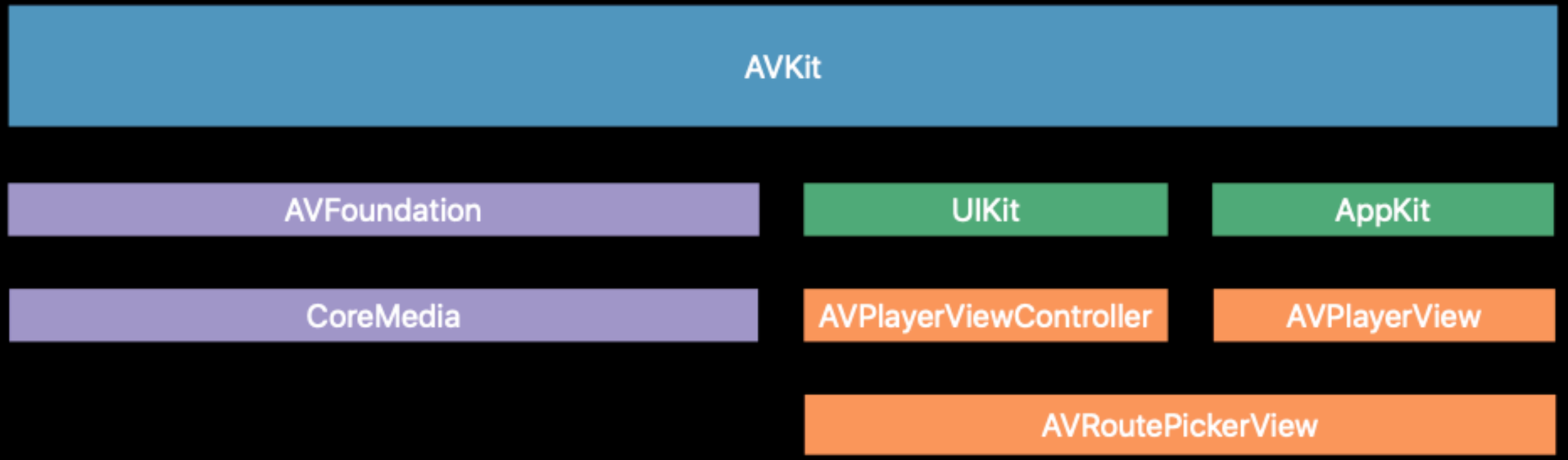
플레이어 UI를 커스텀하지 않고 네이티브한 사용의 경우, AVKit

커스텀을 하고 싶은 경우, AVPlayer, AVPlayerItem

AVPlayer는 AVFoundation에 직접적인 접근을 제공, 미디어 관련된 다양한 기능들을 구현 가능

모두 재생 선택화면을 추가하는 AVRoutePickerView를 제공

AVKit


AVKit
AVFoundation
UIKit
AppKit
CoreMedia
AVPlayerViewController
AVPlayerView
AVRoutePickerView

```swift
// 1) Create an AVPlayer
let urlOfSource = URL(string: "https://my.example/video.m3u8")
let player = AVPlayer(url: urlOfSource!)

// 2) Create an AVPlayerViewController
let playerViewController = AVPlayerViewController()
playerViewController.player = player

// 3) Show it
present(playerViewController, animated: true)
```

```swift
import AVKit
import UIKit

class PlayerViewController: UIViewController {
    @IBOutlet weak var playerViewController: AVPlayerViewController!

    var player: AVPlayer!
    var playerItem: AVPlayerItem!

    override func viewDidLoad() {
        super.viewDidLoad()

        // 1) Define asset URL
        let url: URL = // URL to local or streamed media

        // 2) Create asset instance
        let asset = AVAsset(url: url)

        // 3) Create player item
        playerItem = AVPlayerItem(asset: asset)

        // 4) Create player instance
        player = AVPlayer(playerItem: playerItem)

        // 5) Associate player with view controller
        playerViewController.player = player
    }
}
```


AVMutableMovie
AVComposition
AVPlayerItem
AVURLAsset
HLS
AVCompositing
AVAudioSession
AVAudioMix
movie compositions
AVPlayer
external display support
background audio
AVMovie
AVQueuePlayer
AVMediaSelectionOption
offline HLS
AVAsset
AirPlay 2
temporal compositions
HDR
subtitles
content key management
AVAssetTrack
4K
AVPlayerLayer
AVMetadataItem
file-based playback
AVMutableAssetTrack
network playback

iOS 13.0 에서 새로나온 기능

Full screen callbacks

- Extends AVPlayerViewControllerDelegate
- Notifies delegate when beginning or ending full screen presentation(iOS 12.0)
- Focus on

```
@available(iOS 12.0, *)

func playerViewController(_ playerViewController: AVPlayerViewController,
        willBeginFullScreenPresentationWithAnimationCoordinator coordinator:
        UIViewControllerTransitionCoordinator)

func playerViewController(_ playerViewController: AVPlayerViewController,
        willEndFullScreenPresentationWithAnimationCoordinator coordinator:
        UIViewControllerTransitionCoordinator)
```

```swift
class AVPlayerCoordinator: NSObject, AVPlayerViewControllerDelegate {
        func playerViewController(
            _ playerViewController: AVPlayerViewController,
            willBeginFullScreenPresentationWithAnimationCoordinator coordinator: UIViewControllerTransitionCoordinator
        ) { }

        func playerViewController(
            _ playerViewController: AVPlayerViewController,
            willEndFullScreenPresentationWithAnimationCoordinator coordinator: UIViewControllerTransitionCoordinator
        ) {
            coordinator.animate(alongsideTransition: { (context) in
                // Add coordinated animations
            }) { (context) in
                // 여기서 전환이 성공했는지, 취소됐는지 알 수 있습니다.
                if context.isCancelled {
                    print("전환 실패")
                } else {
                    print("전환 성공")
                    // Take strong reference to playerViewController if needed
                }
            }
        }
    }
```

만약 스크롤 뷰에 플레이어 AVPlayerViewController가 임베드 되어있다고 했을 때,

전체화면 상태에서 기기가 회전하면서 스크롤 뷰 오프셋이 변경되는 경우가 있는데,

테이블 뷰나 컬렉션 뷰에서 이럴 경유 뷰 자체가 할당해제될 수 있어서 문제 생김

그래서, context가 정상적으로 전체화면으로 변경되었는지 확인이 필요하다면 강한 참조를 사용

해당 delegate 메서드를 통하여 애니메이션 이전 이점에서 playerViewController의 위치 reset 필수

iOS 13.0 에서 새로나온 기능

AVPlayerViewController in iPad Apps on the Mac

- Exact same API as on iOS Picture-in-picture support
  iOS Picture-in-Picture 지원과 정확히 동일한 API
- Includes AVPictureInPictureController (also for AppKit) And AVPlayerView, for AppKit-based apps
  AppKit 기반 앱의 경우 AVPictureInPictureController(AppKit용) 및 AVPlayerView 포함
- Touch Bar, keyboard, and Now Playing support Audio and AirPlay routing
  터치 바, 키보드 및 현재 재생중인 미디어 정보 지원 및 AirPlay 라우팅을 지원
- Available in macOS 10.15

External Metadata

- Title, artwork, 혹은 그 외의 메타데이터를 AVKit이 자동으로 처리
- 원하는 Metadata를 보완수정 가능

```
extension AVPlayerItem {
  open var externalMetadata: [AVMetadataItem]
}
```

iOS 13.0 에서 새로나온 기능

Improved Support for Custom Controls

- Interactive dismissals
- Landscape support for portrait-only apps
- Keyboard and Touch Bar support
- Now Playing management
- Automatic video zoom

Custom Playback Controls

- showsPlaybackControls = false
  익히 알고 있는 재생 컨트롤바 표시 여부
- present AVPlayerViewController modally
  플레이어 뷰를 모달 형식으로 띄우기
- Add controls to contentOverlayView
  비디오 컨텐츠와 재생 컨트롤 사이에 보여지는 뷰

Tutorial Example

Showing Full Screen - Splash

```
// Add as child:
parent.addChild(playerViewController)
parent.view.addSubview(playerViewController.view)
// Or other UIView insertion API

// Enable auto layout and set up constraints
playerViewController.didMove(toParent: parent)

// Remove from parent:
playerViewController.willMove(toParent: nil)
playerViewController.view.removeFromSuperview()
playerViewController.removeFromParent()

// 재생 컨트롤을 표시하지 않음:
playerViewController.showsPlaybackControls = false

// 전체 화면을 채우도록 함:
playerViewController.videoGravity = .resizeAspectFill

// 필요하다면 백그라운드 컬러 변경:
playerViewController.view.backgroundColor = .clear
// Or any other UIColor
```

AVFoundation에서 고려해야 할 사항

- AVPlayer.allowsExternalPlayback 비활성화
  스플래시 화면 으로 재생 중인 항목으로 외부 재생을 방지
  누군가의 Apple TV에서 재생되는 음악을 방해하고 싶지 않기 때문

- AVAudioSession을 2차 media playback으로 설정
  Use .ambient category
  다른 어플리케이션 사용 시 서로가 방해가 일어나지 않도록 방지

- 다른 응용 프로그램에 대한 AudioSession Control
  AVAudioSession.silenceSecondaryAudioHintNotification를 통해
  다른 응용 프로그램에서 오디오를 재생하고 있는지 알려줌

  현재 재생하고자 하는 미디어가 우선이 되기를 희망하기에

  다른 실행 중인 오디오가 있는 경우
  AVAudioSession.secondaryAudioShouldBeSilencedHint를 통해
  음소거

Tutorial Example

Showing Full Screen - Full Screen

```
import AVKit
// 1a) Create an AVPlayer
let player = AVPlayer(
    url: URL(
        string: "https://my.example/video.m3u8"
    )!
)

// 1b) Add external metadata if needed
player.currentItem?.externalMetadata = []
// Array of [AVMetadataItem]

// 2) Create an AVPlayerViewController
let playerViewController = AVPlayerViewController()
playerViewController.player = player

// 3) Show it
present(playerViewController, animated: true)
```

- Present modally : 상태 표시 줄 처리가 용이
- Use the default modalPresentationStyle
  - 기본값을 사용하는 이유는 화면 뷰가 UIKit에 의해서 제거될 수 있기 때문
  - 화면이 회전될 때 보이지 않는 다면 레이아웃을 할 이유가 없음
- Do not set AVPlayerViewController.videoGravity
- 전체 화면 상태 확인을 위해 AVPlayerViewControllerDelegate를 사용
  - Do not rely on viewWillAppear(_:) and friends
- AVPlayerItem를 미리 설정해두기
  - Set AVPlayer.rate for setting item
- AVPlayer.status 와 AVPlayerItem.status 관찰해서 제어하기
  - Don’t begin playback until status is .readyToPlay
  - Check error property when status is .failed
    - Rebuild AVFoundation objects if .mediaServicesWereReset
- Enable AVPlayer.usesExternalPlaybackWhileExternalScreenIsActive
- Configure AVAudioSession for .playback

-
  -
  -
  -
-
-
  -
  -
  -
-
  -
  -
-

Delegate 기반 트래킹

프레전트 스타일 설정

커스텀 레이어

뷰 contain 관련 설정

Reference

Apple Docs Archive - AVFoundation Programming Guide

Apple Docs Archive - Media Playback Programing Guide

WWDC 19 - Delivering Intuitive Media Playback with AVKit

주희하다 Blog

Alpaca Blog