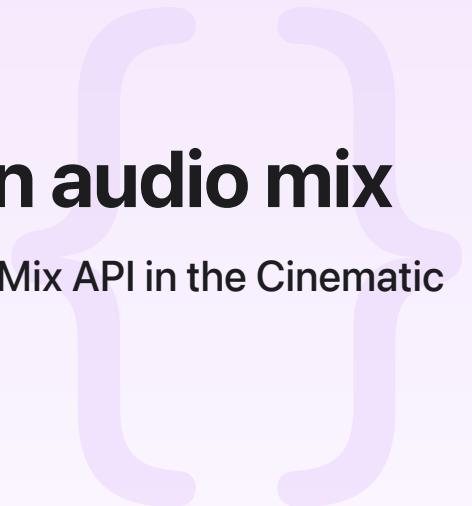Sample Code

Editing Spatial Audio with an audio mix

Add Spatial Audio editing capabilities with the Audio Mix API in the Cinematic framework.

Download

macOS 26.0+ | Xcode 26.0+

Overview

Beginning with iPhone 16, you can use Spatial Audio capture to record video with 3D audio, and edit the audio mix in the Photos app. With Audio Mix, you have creative control of the background and foreground sounds in a recording. It isolates speech as foreground and ambience as background, and you can select between multiple creative rendering styles to adjust the mix.

The `SpatialAudioCLI` sample project is a command-line tool that demonstrates three different methods for applying an audio mix: using `AVPlayer`, using `AVAssetWriter`, and using `AUAudioMix`.

> **Note**
>
> This sample code project is associated with WWDC25 session 251: Enhance your app's audio content creation capabilities.

Configure the sample code project

For best results, use `SpatialAudioCLI` with media that contains a Spatial Audio track. On all iPhone 16 models, Spatial Audio recording is available when capturing video with the Camera app. See the iPhone User Guide for how to change sound recording options.

You can record Spatial Audio in your app by setting the `multichannelAudioMode` property of the `AVCaptureDeviceInput` to a value of `firstOrderAmbisonics`.

Adjust the audio mix in AVPlayer

The simplest way to adjust the audio mix is to play Spatial Audio assets with `AVPlayer`.

First, the sample loads the specified input file into an `AVPlayerItem`:

```
let myAsset = AVURLAsset(url: URL(filePath: "myMediaURL"))
let myPlayerItem = AVPlayerItem(asset: myAsset)
```

Then the sample uses the `AVAsset` to initialize an instance of `CNAssetSpatialAudioInfo`:

```
do {
    // This command throws if the input file does not have proper Spatial Audio.
    let audioInfo = try await CNAssetSpatialAudioInfo(asset: myAsset)
} catch {
    print("A problem occured reading the spatial audio asset: \(error)")
}
```

The two primary mix parameters are `effectIntensity` and `renderingStyle`. The sample creates an `AVAudioMix` with the specified mix parameters and sets it on the `AVPlayerItem`:

```
// Sets the mix parameters.
let intensity = 0.5 // Float values between 0.0 and 1.0.
let style = CNSpatialAudioRenderingStyle.cinematic

// Creates an `AVAudioMix` with effect intensity and rendering style.
let newAudioMix = audioInfo.audioMix(effectIntensity: intensity, renderingStyle: sty
myPlayerItem.audioMix = newAudioMix

// AVPlayer plays the asset with the new audio mix parameters.
let player = AVPlayer(playerItem: myPlayerItem)
```

See Also

Editing

struct **CNDetection**

A structure that represents a detected subject, face, torso or pet at a particular time.

struct **CNDecision**

An object that represents a decision to focus on a particular detection, or group of detections, at a particular time.

class **CNDetectionTrack**

An object representing a series of detections of the same subject over time.

class **CNFixedDetectionTrack**

An object representing the fixed detection track.

class **CNCustomDetectionTrack**

An object representing a discrete detection track composed of individual detections.

enum **CNDetectionType**

The type of object detected, such as face, torso, cat, dog and so on.