Sample Code

Capturing system audio with Core Audio taps

Use a Core Audio tap to capture outgoing audio from a process or group of processes.

Download

macOS 26.0+ | Xcode 26.0+

Overview

This sample code project shows you how to use a tap as an input in a HAL aggregate device, just like a microphone. An audio tap object can specify which outputs it captures from a process or group of processes, as well as different mixdown options (mono, stereo, and so on). Taps can be public (visible to all users), or private (only visible inside the process that created the tap). Taps can also mute the process output so that the process will no longer play to the speaker or selected audio device, and all process output will go to the tap.

Create a tap and an aggregate device

You create a tap by passing a `CATapDescription` to `AudioHardwareCreateProcessTap(_:_:)`. This returns an `AudioObjectID` for the new tap object. You can destroy a tap using `AudioHardwareDestroyProcessTap(_:)`:

```
// Create a tap description.
let description = CATapDescription()

// Fill out the description properties with the tap configuration from the UI.
description.name = tapConfiguration.name
```

```
description.processes = Array(tapConfiguration.processes)
description.isPrivate = tapConfiguration.isPrivate
description.muteBehavior = CATapMuteBehavior(rawValue: tapConfiguration.mute.rawValu
description.isMixdown = tapConfiguration.mixdown == .mono || tapConfiguration.mixdow
description.isMono = tapConfiguration.mixdown == .mono
description.isExclusive = tapConfiguration.exclusive
description.deviceUID = tapConfiguration.device
description.stream = tapConfiguration.streamIndex

// Ask the HAL to create a new tap and put the resulting `AudioObjectID` in `tapID`.
var tapID = AudioObjectID(kAudioObjectUnknown)
AudioHardwareCreateProcessTap(description, &tapID)
```

You similarly create an aggregate device by passing a `CFDictionary` to `AudioHardwareCreateAggregateDevice(_:_:)`, and destroy it using `AudioHardwareDestroyAggregateDevice(_:)`.

```
let description = [kAudioAggregateDeviceNameKey: "Sample Aggregate Audio Device", kA
var id: AudioObjectID = 0
AudioHardwareCreateAggregateDevice(description as CFDictionary, &id)
```

Add a tap to an aggregate device

To use a tap as an input source, add it to an aggregate device that you configure for playback. First get the tap's unique identifier by passing the `kAudioTapPropertyUID` selector and the tap's audio object ID to `AudioObjectGetPropertyData(_:_:_:_:_:_:)`:

```
// Get the UID of the audio tap.
var propertyAddress = getPropertyAddress(selector: kAudioTapPropertyUID)
var propertySize = UInt32(MemoryLayout<CFString>.stride)
var tapUID: CFString = "" as CFString
_ = withUnsafeMutablePointer(to: &tapUID) { tapUID in
    AudioObjectGetPropertyData(tapID, &propertyAddress, 0, nil, &propertySize, tapUI
}
```

Then use the `kAudioAggregateDevicePropertyTapList` selector to get and set the list of taps in an aggregate device. To add a tap, pass the tap's audio object ID and a `CFArray` of `CFString` objects containing the tap's unique identifier to `AudioObjectSetPropertyData(_:_:_:_:_:_:)`:

```
var propertyAddress = getPropertyAddress(selector: kAudioAggregateDevicePropertyTapL
var propertySize: UInt32 = 0
AudioObjectGetPropertyDataSize(self.id, &propertyAddress, 0, nil, &propertySize)
var list: CFArray? = nil
_ = withUnsafeMutablePointer(to: &list) { list in
    AudioObjectGetPropertyData(tapID, &propertyAddress, 0, nil, &propertySize, list)
}

if var listAsArray = list as? [CFString] {
    // Add the new object ID if it's not already in the list.
    if !listAsArray.contains(tapUID as CFString) {
        listAsArray.append(tapUID as CFString)
        propertySize += UInt32(MemoryLayout<CFString>.stride)
    }

    // Set the list back on the aggregate device.
    list = listAsArray as CFArray
    _ = withUnsafeMutablePointer(to: &list) { list in
        AudioObjectSetPropertyData(tapID, &propertyAddress, 0, nil, propertySize, li
    }
}
```

Configure the sample code project

Before you run the sample code project in Xcode, ensure that you're using macOS 14.2 or later.

> **Important**
>
> To capture audio with a tap, you need to include the `NSAudioCaptureUsageDescription` key in your Info.plist file, along with a message that tells the user why the app is requesting access to capture audio.

The first time you start recording from an aggregate device that contains a tap, the system prompts you to grant the app system audio recording permission.

See Also

Drivers

{} Creating an Audio Server Driver Plug-in

Build a virtual audio device by creating a custom driver plug-in.

{} Building an Audio Server Plug-in and Driver Extension

Create a plug-in and driver extension to support an audio device in macOS.