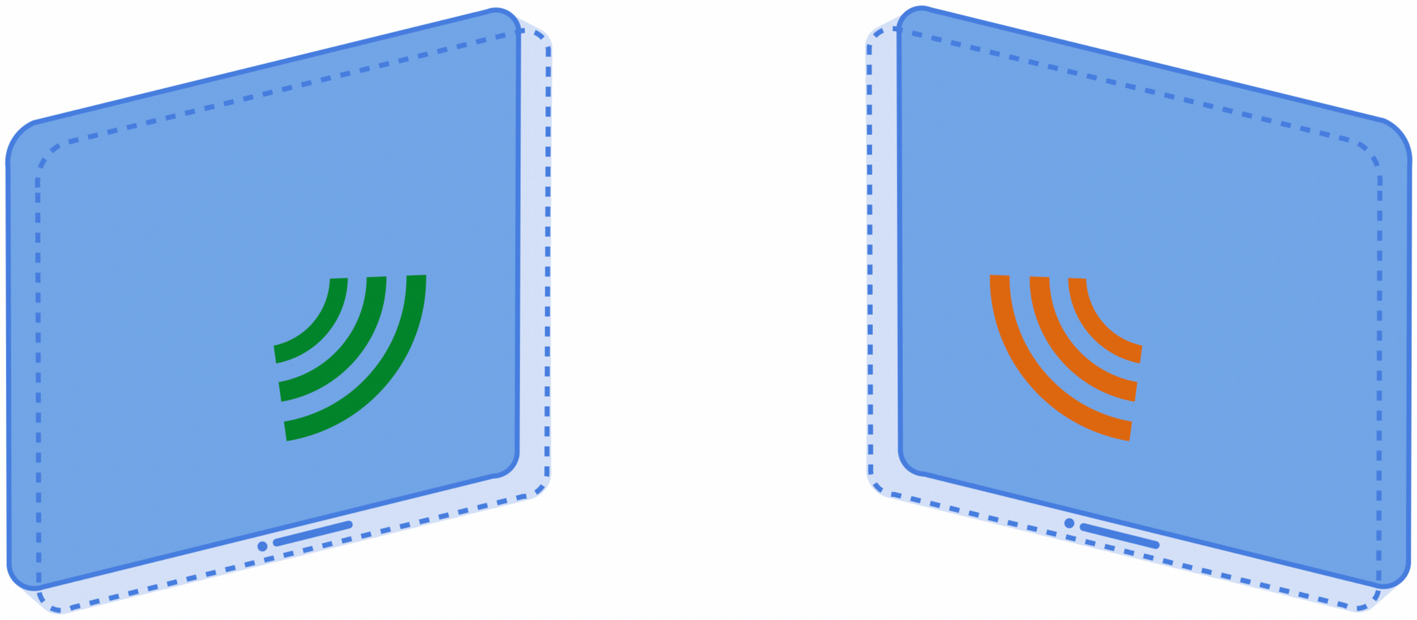
Article

Anchoring sound to a window or volume

Provide unique app experiences by attaching sounds to windows and volumes in 3D space.

Overview

Many audio playback APIs have a property to configure their 3D spatial rendering using the `SpatialAudioExperience` type `HeadTrackedSpatialAudio`. This article shows how to take advantage of `HeadTrackedSpatialAudio` to place each sound at the center of its intended `UIScene` in your multiwindow or multivolume application.



Get the scene's identifier

Placing a sound on a specific `UIScene` requires knowledge of the target scene's `persistent Identifier`. In a SwiftUI application, that means adding both a `UIApplicationDelegate` and

UISceneDelegate to your SwiftUI App:

```
import SwiftUI

@main
struct MyApplication: App {
    @UIApplicationDelegateAdaptor var delegate: MyAppDelegate

    var body: some Scene {
        WindowGroup {
            ContentView()
        }
    }
}

class MyAppDelegate: NSObject, UIApplicationDelegate, ObservableObject {
    func application(_ application: UIApplication,
                     configurationForConnecting connectingSceneSession: UISceneSessi
                     options: UIScene.ConnectionOptions) -> UISceneConfiguration {
        let sceneConfig = UISceneConfiguration(name: nil, sessionRole: connectingSce
        sceneConfig.delegateClass = MySceneDelegate.self
        return sceneConfig
    }
}

class MySceneDelegate: NSObject, UISceneDelegate, ObservableObject {
    var sceneIdentifier: String?

    func scene(_ scene: UIScene, willConnectTo session: UISceneSession, options conn
        sceneIdentifier = session.persistentIdentifier
    }
}
```

The following code makes the identifier for each UIScene accessible from any SwiftUI View using your UISceneDelegate as an EnvironmentObject:

```
import SwiftUI

struct ContentView: View {
    @EnvironmentObject var sceneDelegate: MySceneDelegate

    var body: some View {
        Text("\(String(describing: sceneDelegate.sceneIdentifier))")
    }
}
```

Anchor the sound to the scene

With a UIScene identifier in-hand, configure each sound using a HeadTrackedSpatialAudio structure.

```
import SwiftUI
import AVFAudio

struct ContentView: View {
    @EnvironmentObject var sceneDelegate: MySceneDelegate

    @State var player: AVAudioPlayer? = {
        guard let url = Bundle.main.url(forResource: "my_sound", withExtension: "wav
            return nil
        }
        return try? AVAudioPlayer(contentsOf: url)
    }()

    var body: some View {
        Text("Hello, Sound!")
    }
    .onAppear {
        guard let player = self.player, let sceneID = self.sceneDelegate.sceneIdenti
            return
        }

        player.intendedSpatialExperience = .headTracked(.scene(identifier: sceneID))
        player.play()
    }
}
```

Besides just AVAudioPlayer, you can also use SpatialAudioExperience types with the other playback APIs listed below.

Spatialize system and alert sounds

Configure the spatial audio experience of your system and alert sounds using:

- `AudioServicesPlaySystemSound(_:spatialExperience:)`
- `AudioServicesPlayAlertSound(_:spatialExperience:)`

Spatialize audio-only playback APIs

Configure the spatial audio experience of audio-only playback APIs using the `intendedSpatialExperience` property on:

- AVAudioPlayer
- AVAudioOutputNode
- AUAudioUnit
- CHHapticEngine

Spatialize audio playback APIs that also have video

Setting a scene identifier on playback APIs that have video content isn't always necessary as their sound automatically anchors to its visual counterpart. However, if there is no video or if you prefer something besides the automatic behavior, configure the spatial audio experience of these playback APIs using the `intendedSpatialAudioExperience` property on:

- AVPlayer
- AVSampleBufferRenderSynchronizer

See Also

Playback and Recording

Audio Queue Services

Connect to audio hardware and manage the recording or playback process.

Audio Services

Play short sounds or trigger a vibration effect on iOS devices with the appropriate hardware.

Music Player

Create and play a sequence of tracks, and manage aspects of playback in response to standard events.