#WWDC19

Audio API Updates

Peter Vasil, Audio Software Engineer
What’s New in AVAudioEngine
AVAudioEngine Enhancements

Voice processing support

New realtime audio input and output nodes
  • AVAudioSinkNode
  • AVAudioSourceNode

Spatial audio rendering improvements
Voice processing mode

- For use in VoIP apps
- Not supported in manual rendering mode
- Set on either input or output node

For echo cancellation, both input and output nodes will be operating in voice processing mode
open func setVoiceProcessingEnabled(_ enabled: Bool) throws

open var isVoiceProcessingEnabled: Bool { get }
AVAudioEngine — Source and Sink Nodes

Wraps a user-defined block that allows apps to send or receive audio from AVAudioEngine

- AVAudioSourceNode
- AVAudioSinkNode

- When rendering to a device, blocks operate under realtime constraints and should not perform any blocking calls
AVAudioSourceNode

Wraps a user-defined render block that supplies audio data on audio IO thread

Supported in both realtime and manual rendering mode

Supports linear PCM conversions

One output bus, no input
// Create Engine
let engine = AVAudioEngine()

// Create and Attach AVAudioSourceNode
let sourceNode = AVAudioSourceNode() { (silence, timeStamp, frameCount, audioBufferList) -> OSStatus in
    let ablPointer = UnsafeMutableAudioBufferListPointer(audioBufferList)
    for buffer in ablPointer {
        ...
    }
    return noErr
AVAudioSinkNode

Wraps a user-defined block to receive input on audio IO thread

Must be downstream of the input node and operates in realtime

One input bus, no output
AVAudioSinkNode

// Create Engine
let engine = AVAudioEngine()

// Create and Attach AVAudioSinkNode
let sinkNode = AVAudioSinkNode() { (timeStamp, frames, audioBufferList) -> OSSStatus in
    ...
}
engine.attach(sourceNode)
AVAudioEngine — Spatial Rendering

Automatic spatial rendering algorithm

Improvements to support spatialization of multichannel audio content
Auto Spatial Rendering Algorithm

Automatically picks the most appropriate spatialization algorithm for current route

Enable best experience fine tuned for each supported product

Adds near-field and in-head rendering for headphones
public enum AVAudio3DMixingRenderingAlgorithm : Int {
    ...
    case auto
}

open var outputType: AVAudioEnvironmentOutputType

public enum AVAudioEnvironmentOutputType : Int {
    case auto
Ability to Spatialize Multichannel Streams

Supports point-source and ambience bed rendering

Channel-based formats and higher-order Ambisonics
var sourceMode: AVAudio3DMixingSourceMode

public enum AVAudio3DMixingSourceMode : Int {
    case spatializeIfMono
    case bypass
    case pointSource
    case ambienceBed
}

var pointSourceInHeadMode: AVAudio3DMixingPointSourceInHeadMode

public enum AVAudio3DMixingPointSourceInHeadMode : Int {
}
// Example: Ambience Bed with Auto Rendering Algorithm

```swift
let engine = AVAudioEngine()

// Create and Configure Environment Node
let environment = AVAudioEnvironmentNode()
// use automatic detection of output type (does not work in Manual Rendering modes)
environment.outputType = .auto
engine.attach(environment)

// Create an Ambience Bed Using Auto Rendering Algorithm
let player = AVAudioPlayerNode()
player.renderingAlgorithm = .auto
```
What’s New in AVAudioSession
AVAudioSessionPromptStyle

A hint to apps that play voice prompts to modify the style of prompt played

• Prompt style changes in response to other audio activity on the system, such as Siri or phone calls
• Recommended for navigation apps for better user experience
• .none — refrain from playing any prompts
Other AVAudioSession Enhancements

Default policy is to mute haptics and system sounds when audio recording is active.

Apps can now allow haptics and system sounds while recording using the `allowHapticsAndSystemSoundsDuringRecording` property.
More Information

developer.apple.com/wwdc19/510