AyuSynk IOS SDK

AyuSynkSdk encapsulates the core functionalities found in the AyuShare app and allows you to seamlessly integrate AyuSynk devices into your iOS applications.

The AyuSynkSdk is designed to be easily integrated using Swift Package Manager.

Requirements

- **Minimum Deployment** : iOS 12
- **Supported Destinations** : iPhone and iPad

Required Permissions

For Connecting Devices via Bluetooth

To enable the functionality of AyuSynk devices, the following permissions must be added to your app's Info.plist file:

1. **Privacy - Bluetooth Always Usage Description** : This permission is necessary for establishing seamless and continuous connections with Bluetooth devices.
2. **Privacy - Bluetooth Peripheral Usage Description** : This permission allows your device to interact with nearby Bluetooth devices.

Usage

You can incorporate the AyuSynkSdk library into any iOS project by adding it to your project and using its public API functions.

For generating diagnosis reports, you'll need a specific clientId provided by AyuDevices. Refer to the #online-live-streaming section for more details. If you have a clientId, add it to your app using the instructions in the Add-Client-Id section.

Instructions

Step-by-step instructions for using AyuSynkSdk:

Step 1: Open Your Xcode Project

Launch Xcode, open your existing iOS project, or create a new one.

Step 2: Adding Your SDK via SPM

1. In Xcode, go to "File" > "Swift Packages" > "Add Package Dependency..."
2. In the "Choose Package Repository" dialog, enter the 'PACKAGE_URL' provided by AyuDevices.
3. Click "Next" and select the appropriate version.
4. Choose your target, then click "Finish."

Please note that our SDK relies on the Coreplot library to display waveforms. Ensure you add the Coreplot library to your project. Sample code includes a pod for adding Coreplot via CocoaPods.

Step 3: Import and Use the SDK

In your Swift source files, import the SDK using:

```
import AyuSynkSDK
```

Now, you can start using the features and functionality provided by the SDK in your app.

Step 4: Build and Run

Build and run your app. The SDK functionality is now available for use within your app.

Sample Project

To quickly get started, follow these steps to run the sample project:

Step 1: Navigate to the Sample Folder

Navigate to the sample project folder within your app's directory.

Step 2: Install Dependencies

1. Open a Terminal and run the following command to install necessary dependencies using CocoaPods:

```
pod install
```

This command downloads and installs CorePlot for the sample project.

Step 3: Open the Workspace

Open the project using the `AyuSynk_IOS_SDK.xcworkspace` file instead of the `.xcodeproj` file. CocoaPods integrates additional libraries into the Xcode workspace.

Step 4: Run the Sample App

Open the sample app in Xcode, select your target (usually a physical device), and click the "Run" button to build and run the sample project on your device.

You're all set to explore the features and functionality of the AyuSynk iOS SDK within the sample project.

For a comprehensive demonstration of the SDK capabilities, refer to the AyuShare app, which is also based on this SDK.

Features

AyuSynkSdk offers the following features:

1. List available AyuSynk devices
2. Connect to AyuSynk devices
3. Stream audio
4. Waveform UI
5. Record audio
6. Save audio
7. Get AyuSynk battery status
8. Get AyuSynk signal strength
9. Share audio
10. Denoise audio
11. Diagnosis Report Generation

Diagnosis Report Generation

To generate diagnosis reports, you need a clientId from AyuDevices. Refer to the #Get-sdk-access section for more details. If you have a clientId, add it to your app using the instructions in the Add-Client-Id section.

1. Report Generation Options:

1. Generate Using Cloud URL
2. Upload and Generate

2. Generate Using Cloud URL

1. Upload recorded wav files to a public cloud storage.
2. Set a listener using the `setDiagnosisReportUpdateListener()` function to receive the diagnosis report once generated.
3. Pass the cloud URL of the wav file to the `generateDiagnosisReport()` function, as shown below:

```
let soundData = SoundData(fileLink: "https://your-cloud-storage-link.wav", location
let soundFile = SoundFile(soundData: soundData, soundType: SoundType.HEART)
AyuSynk.generateDiagnosisReport(soundFile: soundFile)
```

3. Upload and Generate

1. Save the recorded audio in wav file format.
2. Set a listener using the `setDiagnosisReportUpdateListener()` function to receive the diagnosis report once generated.
3. Pass the wav file to the `generateDiagnosisReport()` function, as shown below. Refer to `MainViewController.swift` for the `generateReportButtonClicked()` function:

```
let soundData = SoundData(fileUrl: fileURL, locationName: LocationType.Heart.aortic
let soundFile = SoundFile(soundData: soundData, soundType: Sound)
AyuSynk.generateDiagnosisReport(soundFile: soundFile)
```

NOTE

1. it is recommended to send proper heart/lung location if known for better results.
2. Based on our research in past years we recommend to record sound for 10 sec which is enough for doctor to diagnose properly
3. Report generation can take up to ~15-20 secs for 10 sec audio. More time will be taken if the size of recording exceeds then recommended 10 sec audio.

Add client id

```
AyuSynk.verify(clientId: "YOUR_CLIENT_ID", verificationListener: self)
```

API

You can use AyuSynkSdk as an SDK by embedding into your IOS application and calling its public API functions described below.

The whole API is defined in the AyuSynk class, thus you will need to import only this single class once you have added framework in your project:

```
import AyuSynkSDK
```

Verify User

No.	ReturnType	Function	Description
01	Void	verify(clientId: String, verificationListener: VerificationListener?)	Used to verify client

API for interacting with Device

No.	ReturnType	Function	Description
01	Bool	isBluetoothEnabled()	Used to check if bluetooth is enabled
02	Void	setDeviceScanListener(listener: DeviceScanListener)	Set this listener to get callbacks when using startScan()/stopScan() methods
03	Void	startScan()	Starts scanning for nearby device
04	Void	stopScan()	Stops scanning for nearby device.
05	Void	connect(deviceUUID: String)	Connect to device by passing the uniqueIdentifier of the device. deviceUUID is obtained in Device object of DeviceScanListener method
06	DeviceConnectionState	isDeviceConnected()	Returns DEVICE_CONNECTED if device is connected else DEVICE_DISCONNECTED
07	Void	setAyuDeviceListener(listener: AyuDeviceListener)	Set this listener to get callbacks for device connection, strength and battery value changes

No.	ReturnType	Function	Description
08	DeviceStrength	getDeviceStrength()	Returns current signal strength. Recommended to always refer values returned via AyuDeviceListener
09	Int	getCurrentBatteryLevel()	Returns current battery level. Recommended to always refer values returned via AyuDeviceListener
10	Void	disconnect()	Disconnect all existing connections

Waveform API

No.	ReturnType	Function	Description
01	Void	setupVisualizerView(view:CPTGraphHostingView, color: CGColor)	Set AyuVisualizer for plotting audio waveform. import CorePlot to get CPTGraphHostingView

Record and Playback API

No.	ReturnType	Function	Description
01	Void	setFilter(filter: FilterType)	Used to apply different denoising on audio stream
02	Void	setRecordingTimeLimit(recordingTimeLimit: Int)	Used to set the recording time limit
03	Void	setRecorderListener(listener: RecorderListener)	Used to get various recording callback when using record or stream features
04	Void	startRecording()	Starts recording the current audio stream
05	Void	pauseRecording()	Used to pause recording
06	Void	clearRecordedData()	Used to discard any recorded audio before it is completed.
07	Data?	getAudioData(recordID: String)	Get recorded data
08	Void	playAudio(recordID: String)	Used to play recorded audio
09	Void	stopAudioPlayback()	Stop playback
10	Void	changePlaybackSpeed(speed: PlayBackSpeeds)	Change playback speed to VERY_SLOW, SLOW and NORMAL

No.	ReturnType	Function	Description
11	Void	resetSpeed()	Resets speed to NORMAL
12	Void	muteAudio(mute : Bool)	Mute audio

Report Generation API

No.	ReturnType	Function	Description
01	URL?	generateFile(fileName: String, recordID: String)	Generate a wav file in local storage and returns file url
02	URL?	generateFile(fileName: String, sampleData: Data)	Generate a wav file in local storage and returns file url
03	Void	setDiagnosisReportUpdateListener(listener: DiagnosisReportUpdateListener)	Set this listener to get callbacks for report generation
04	Void	generateDiagnosisReport(soundFile: SoundFile)	To generate diagnosis reports from recorded sound file. You need to have valid clientId to generate reports.

Miscellaneous API

No.	ReturnType	Function	Description
01	Void	close()	Releases all the allocated memory by AyuSynkSdk. Should call before closing app

For any further queries. Write an email to ayuanalytics@ayudevices.com specifying subject as " <YOUR_COMPANY_NAME> - Query for IOS SDK "