iOS Public LOUIE7 SDK DOCUMENTATION
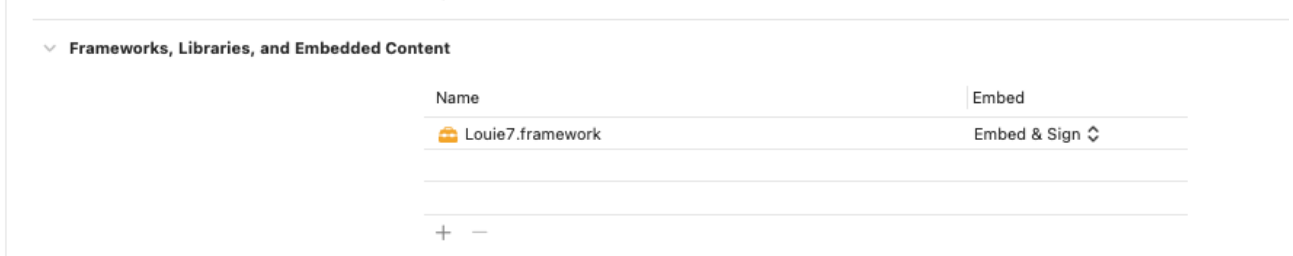
📋 Overview

The Louie7 SDK collects user data according to permissions given by the user.

Integrate Louie7 iOS SDK into your iOS app

To integrate Louie7 SDK into your app, you have to follow these steps:

1. Download the SDK file.
2. Unzip the file and copy **Louie7.framework** to the project folder.
3. Add the framework to **Frameworks, Libraries, and Embedded Content,** and make sure you select **Embed & Sign**

> Frameworks, Libraries, and Embedded Content

Name	Embed
📁 Louie7.framework	Embed & Sign ⌄

+ —

4. Import Louie7 at the top in **AppDelegate.swift**

```
import Louie7
```

5. Initialize the SDK in the AppDelegate class

```
let louieSDK = Louie7.build(apiKey: "API_KEY")
```

6. Add start() action to `didFinishLaunchingWithOptions` method body in AppDelegate

```
louieSDK.start()
```

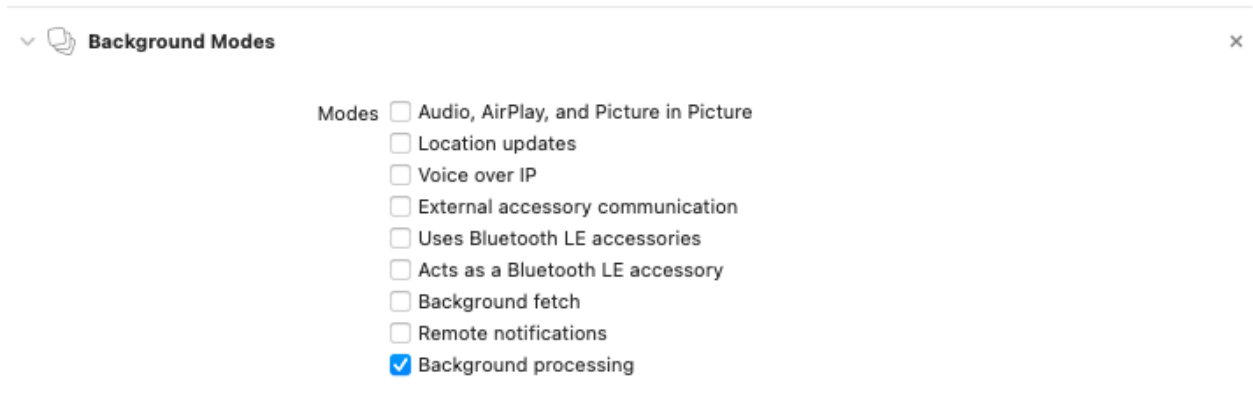7. Call this method to get the user id:

```
louieSDK.getUserIdentifier()
```

Enable iOS Background Logging

Background processing can be initialized only by the client app, the SDK can't register backgrounds task.

In order to make it possible for SDK to collect data in the background, the developer should create the background task identifier and pass it to the Louie7 SDK setup method.

Steps to setup background logging:

1. In **Xcode** > **Project** > **Capabilities**, add **Background Modes** and select **Background processing** checkbox.

2. Register a background task identifier in **Info.plist**

```
<key>BGTaskSchedulerPermittedIdentifiers</key>
<array>
  <string>background_task_identifier</string>
</array>
```

3. Configure Louie7 SDK with the task identifier in **AppDelegate.swift**

```
 func application(_ application: UIApplication,
didFinishLaunchingWithOptions launchOptions: [UIApplication.
LaunchOptionsKey: Any]?) -> Bool {
        Louie7.handleBackgroundTask(apiKey: "API_KEY", with:
"background_task_identifier")
        return true
    }
```

How to integrate with SDK's API

To integrate your client application you have authorize your device identifier. To authorize your device identifier send a POST request to SDK's API.

All API requests require the use of a received application key. To authenticate an API request, you should provide your application key in the X-API-Key header.

Header Parameter	Type	Description
X-API-Key	string	Your application key

Next , to authorize a device identifier

You have send a POST request to SDK's API using received X-API-Key key.

Authorize device identifier

```
POST /api/v1/devices
```

Body Parameter	Type	Description
identifier	string	Device identifier

Header Parameter	Type	Description
X-API-Key	string	Your application key

Authorize device identifier example

Now , that you've successfully authorized your application's device identifier , you request application's SDK's config.

Next, to receive application config

You have send a GET request to SDK's API using received access (Authorization) header.

Receive application config

```
GET /api/v1/application/config
```

Header Parameter	Type	Description

Header Parameter	Type	Description
`Authorization`	`string`	Your device identifier access

Receive application config example

You can also download the application config as a file

Download application config

```
GET /api/v1/application/config/download
```