iOS SDK HOW TO USE DOCUMENTATION

DOCUMENT VERSION:	11.1
iOS SDK VERSION:	3.9.7
iOS BUNDLE VERSION	3.9.7
DATE REVISED	April 2022

Installation

Manual

Drag the Framework folder into your project.

Cocoapods

CocoaPods is a dependency manager for Cocoa projects. For usage and installation instructions, visit their website.To integrate SpreoUIModule into your Xcode project using CocoaPods, specify it in your Podfile:

pod 'SpreoPod'

Table of Contents

1. Overview

This document provides a step-by-step walkthrough of the IndoorKit SDK implementation process. It also provides the necessary code samples required to get started.

2. Environment

- Xcode 11 or higher Core Motion.
- iOS 11 or higher
- iOS devices with Bluetooth 4.0

3. Manual Setup

3.1. Add System Frameworks

- Core Location
- Core Motion
- Core Bluetooth
- MapKit
- Media Player
- AVFoundation
- libc++.dylib

3.2. Add Google Framework

From the finder, drag the GoogleMaps.framework into your app Frameworks group. Alternatively, from your app target -> build phase -> Link Binary with Libraries add Google- Maps.framework framework.

3.3. Add Indoor Kit Framework

From the finder, drag the IndoorKit.framework into your app Frameworks group.Alternatively, from your app target -> build phase -> Link Binary with Libraries add IndoorKit.framework framework.
In the app target go to Build Settings and search for Other Linker Flags, then add this line (copy and paste):

$(inherited) -ObjC -l"c++" -l"icucore" -l"z" -framework "AVFoundation" -framework "Accelerate" -framework "CoreBluetooth" -framework "CoreData" -framework "CoreGraphics" -framework "CoreLocation" -framework "CoreText" -framework "GLKit" -framework "GoogleMaps" -framework "ImageIO" -framework "OpenGLES" -framework "QuartzCore" -framework "Security" -framework "SystemConfiguration" -framework "MessageUI" -framework "MapKit" -framework "Accounts" -framework "Social" -framework "MediaPlayer" -framework "CoreMotion"

3.4. Add Google Resource Bundle

From the finder, drag the IndoorKit.bundle into your app Supporting Files group. Alternatively, right click on the Supporting Files group in the project tree and select Add Files To, and select the GoogleMaps.bundle.

3.5. Add Indoor Resource Bundle

From the finder, drag the IndoorKit.bundle into your app Supporting Files group. Alternatively, right click on the Supporting Files group in the project tree and select Add Files To, and select the IndoorKit.bundle.

3.6. Verifications:

3.6.1. Framework membership

Open the file inspector (top right), check framework membership by selecting the framework and ensure that the target box is checked.

3.6.2. Deployment Target

Open the General tab of your project target to ensure that Deployment Target is 11.x or above.

3.6.3. Info Plist required keys

iOS 8.x and later, make sure you add a value for either NSLocationWhenInUseUsageDescription or NSLocationAlwaysUsageDescription key in Info.plist with a message to be displayed in the prompt. Provide (it is required) a description of “why the app want to use location services” in the Info.plist
Example:
<key>NSLocationAlwaysUsageDescription</key>
<string>The app requires the device location in order to notify when close to campus</string>
<key>NSLocationWhenInUseUsageDescription</key>
<string>The app requires the device location in order to notify when close to campus</string>

4. Enable IndoorKit Services

In the app delegate implementation file add the following:
#import <IndoorKit/IndoorKit.h>
In the method didFinishLaunchingWithOptions add the call to the method setApiKey:andValue with your app key, which will enable your service (as provided for your app) .
NOTE !
If this error occurrs, further execution is not allowed.

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
        // enable the indoor kit positioning
        IDError *error;
        [IDKit setAPIKey:@"<#PUT-YOUR-SPREO-API-KEY-HERE#>" error:&error];
        if (error) {
                // do something, that was not suppose to happen
                NSLog(@"IDKit error! %d - %@", (int)error.code, error.domain);
        }
        return YES;
}
```

5. Check for updates and Data Model Initialization

Add the IDDataUpdateDelegate to the AppDelegate supported protocols:

```
@interfaceAppDelegate () <IDDataUpdateDelegate>
@end
```

Then add the following methods to the AppDelegate:

```
#pragma mark - IDDataUpdateDelegate
- (void)dataUpdateStatus:(IDDataUpdateStatus)status
{
        switch (status) {
        case kIDDataUpdateCheckForUpdates:
        // do something, display the user the current status
        break;
        case kIDDataUpdateCopyFiles:
        // do something, display the user the current status
        break;
        case kIDDataUpdateDataDownload:
        // do something, display the user the current status
        break;
        case kIDDataUpdateInitializing:
        // do something, display the user the current status
        break;
        case kIDDataUpdateDone:
        // do something, display the user the current status
        // when done, can start user location tracking
        [IDKit startUserLocationTrack];
        break;
        default:
        break;
        }
}
-(void)dataUpdateFailedWithError:(IDError *)anError
{
        NSLog(@"%@", anError.domain);
}
```

Once the update and initialization delegate is set, add the trigger method:

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions{
// enable the indoor kit positioning
[IDKit setAPIKey:@"<#PUT-YOUR-SPREO-API-KEY-HERE#>" error:&error];
// start the background data update check, download, and initialization
[IDKit checkForDataUpdatesAndInitializeWithDelegate:self];
return YES;
}
```

6. Languages

Get supported languages

```
(NSArray*)supportedLanguages
// The Method returns an array of strings for supported languages @[@"en", @"es", @"ch", etc…]
NSArray * supportedLanguages = [IDKit supportedLanguages];
```

Get current Language

```
+ (NSString*)getCurrentLanguage
// The method returns the SDK current language
NSString* currentLang = [IDKit getCurrentLanguage];
```

Set current Language

```
+ (BOOL)setCurrentLanguage:
```

The method should be called to set the SDK current language,
Some things can vary by the selected language, like POI's data, map labels if founded, etc...,
Then the map will reload with the new language data by selected current language.

```
[IDKit setCurrentLanguage:@"en"];
```

7. Get CMS POIs

In Order to get all CMS Poi's array. The POI id should include campus id and facility id as exampled:

```
NSArray * allPois = [IDKit getPOIsWithID:@"short_hills.short_hills"];
```

8. Get Sorted POIs

In order to get the POI list to be sorted alphabetically by category :

```
NSArray * alphabetPois =[IDKit getPOIsSortedAlphabeticallyWithCategories:@[@"Services"]
atPathID:@"short_hills.short_hills"];
```

In order to get the POI list sorted alphabetically:

```
NSArray * alphabetPois =[IDKit getPOIsSortedAlphabeticallyWithPathID:@"short_hills.short_hills"];
```

In order to get the POI list sorted by distance from location (if location was nil, the system will sort poi's by user location):

```
NSArray * sortedPoisByDistance =[IDKit getPOIsSortedDistantlyWithPathID:@"short_hills.short_hills"
fromLocation:nil];
```

In order to get the POI list sorted by user location position for categories array:

```
NSArray * sortedPoisByDistance =[IDKit getPOIsSortedDistantlyWithCategoriesy:@[@"Services"]
atPathID:@"short_hills.short_hills"
fromLocation:nil];
```

9. Add Custom POI's

Adding custom POI's to the data module can be done once the data update is finished.

```
- (void)addCustomPoi
{
        // create the poi location
        IDLocation *poiLocation = [[IDLocation alloc] initWithCampusId:@"shorthils"
        facilityId:@"shorthils"
        outCoordinate:CLLocationCoordinate2DMake(0.f, 0.f)
        inCoordinate:CGPointMake(659.f, 780.f)
        andFloorId:1];
        // create the POI object (IDPoi) with it's location and extra info
        IDPoi *indoorPoi = [[IDPoi alloc] initPoiWithTitle:@"Mavericks"
        subtitle:@"surf boards"
        description:@"Specialize in long to short boards"
        identifier:@"mavericks"
        categories:@[@"shop",@"outdoor"]
        location:poiLocation
        andInfo:@{@"telephone": @"177-8080-2020"}];
        // add the poi to the data module, associate the poi objects to the object path with format:
        // @"campus_id.facility_id"
        [IDKit addPOIsInArray:@[indoorPoi] toObjectPath:@"shorthils.shorthils"];
}
```

10. Container Maps Shared Protocol IDMapViewProtocol

The shared protocol IDMapViewProtocol defined as the shared API Methods for the Map Containers. basically there are two types of API methods:

11.1 Control methods

```
// present location on map
[self.mapVC presentLocation:aLocation];

// present poi on map
[self.mapVC presentPoiOnMapWithPoi:aPoi];

// show / hide bubble For Poi
[self.mapVC showBubbleForPoi:aPoi];
[self.mapVC hideBubbleForPoi:aPoi];

// center map on facility by providinf a facilityId and a campusId
[self.mapVC centerFacilityMapWithFacilityId:aFacilityId atCampusId:aCampusId];

// center map on campus by providing a campusId
[self.mapVC centerCampusMapWithCampusId:aCampusId];

// show floor components at faciliy
[self.mapVC showFloorWithID:1 atFacilityWithId:aFacilityId];

// enter follow me mode
[self.mapVC showMyPosition];

// zoom in
[self.mapVC zoomIn];

// zoom out
[self.mapVC zoomOut];

// set zoom level in case provided zoom level < self.mapVC.mapMinZoomLevel
// and zoom level > self.mapVC.mapMaxZoomLevel
[self.mapVC setMapZoomLevel:aZoomLevel];

// reload all map components
[self.mapVC mapReload];
```

11.2 Customization methods

```
// custom the map type
[self.mapVC setMapType:kIDMapTypeStandard];

// custom the map layers
[self.mapVC setMapShowLayer:kIDMapLayerPaths mode:YES];

// custom the map pois
[self.mapVC showAllPois];
[self.mapVC hideAllPois];
// custom the map labels
[self.mapVC showAllLabels];
[self.mapVC hideAllLabels];

// custom the map pois by categories
[self.mapVC setVisiblePOIsWithCategories:@[@“entrance"]];

// set the map rotation mode to compass mode
[self.mapVC setMapRotationMode: kIDMapRotationCompass];
```

```
// set update timer duration to 12 sec
[self.mapVC setUserAutoFollowTimeInterval:12.f];
// set poi's region radius to 30 meters
[self.mapVC setPoiRegionRadius:30];
```

11. Container Map – IDDualMapViewController

To implement this type of map view controller In the implementation file (.m) declare an interface extension. Add the property that holds the map view controller. Then add the declaration of IDDualMapViewControllerDelegate delegation support.

```
@interface MainViewController () <IDDualMapViewControllerDelegate>
@property (nonatomic, strong) IDDualMapViewController *mapVC;
@end
```

In the viewDidLoad method call to getDualMapViewController and assign its return value to the map property declared earlier. Set the map events delegation directly to the map view controller. Provide you Google Maps SDK API Key (for more info how to get google maps api key for iOS SDK:

- See http://www.themeskingdom.com/knowledge-base/how-to-generate-google-api-key
- See video link https://www.youtube.com/watch?v=69ZwR4o7oGQ
- Go to http://console.developers.google.com/)

Custom the map view controller settings. Add the map view controller and its child view to the base view controller hierarchy.

```
- (void)viewDidLoad {
[super viewDidLoad];
// get the dual map view controller
self.mapVC = [IDKit getDualMapViewController];
// provide your google api key
[self.mapVC provideGoogleMapsAPIKey:GOOGLE_API_KEY];
// custom the map settings
self.mapVC.settings.indoorPicker = YES;
self.mapVC.settings.myLocationButton = YES;
self.mapVC.delegate = self;
self.mapVC.padding = UIEdgeInsetsMake(self.topLayoutGuide.length, 0, 44, 0);
// add the map view controller view as sub view
[self.view addSubview: self.mapVC.view];
[self.view sendSubviewToBack:self.mapVC.view];
// add the map view controller as a child view controller
[self addChildViewController:self.mapVC];
[self.mapVC didMoveToParentViewController:self]; }
```

12. Customizing Container Map – IDDualMapViewControllerDelegate

The map components can also be customized by responding to one or more of the map delegation methods IDDualMapViewControllerDelegate.

Here is an example of some delegate methods:

```
#pragma mark – IDDualMapViewControllerDelegate

// Custom Map Use Annotation Icon Image
- (UIImage *)mapIconForUserAnnotaion
{
        return nil;
}

// Custom Map Poi Annotation Icon Image
- (UIImage*)mapIconForPoi:(IDPoi *)aPoi
{
        return nil;
```

```
}

// Custom Map Navigation Route Color
- (UIColor *)mapColorForRoute
{
        return [UIColor greenColor];
}
```

13. Control Map Locked Region

Call this method to lock the map to a region – as a result the map will lock either the indoor mode or the outdoor mode or unlock the views. For example, lock to indoor map

```
- (void)setMapLockToIndoor
{
        // lock to indoor mode
        IDLocation * location = [[IDLocation alloc] init];
        location.campusId = @" kCampusId";;
        location.facilityId = @"kFacilityId";;
        // first
        [self.mapVC presentLocation:location];
        // second
        NSString *path = :[NSString stringWithFormat:@"%@.%@", location.campusId, location.facilityId];
        [self.mapVC setMapLockToRegionPath: path];
        // lock to outdoor mode
        location.facilityId = nil;
        path = :[NSString stringWithFormat:@"%@.%@", location.campusId, location.facilityId];
        [self.mapVC setMapLockToRegionPath: path];
        // unlock
        [self.mapVC setMapLockToRegionPath: nil];
}
```

14. Set Draw Trip Overview

The method will add trip overview routes, with enumerated POIs (in circles), to the indoor map. The arrived (visited) Pois array will draw a circle to the Poi location in a different color.

```
// to set map draw trip overview with poisTrip list array
[self.mapVC setMapDrawTripOverviewWithPois:myPoisTripList arrivedPois:nil drawSwitchFloorsCircles:NO
drawEntrancesCircles:NO];
```

15. Remove Trip Overview

```
//call this method to Remove Trip Overview from map.
[IDKit setMapRemoveTripOverview];
```

16. Order Trip's POIs List

In order to get POI list of the itinerary sorted by shortest journey:

```
// get ordered pois by user location and add switch floors pois and the parking location
NSArray* sortedArr = [IDKit orderPoisLocationsArray:array2sort
addSwitchFloorsLocations:YES addParkingLocation:YES];
// remove exist overview
[IDKit setMapRemoveTripOverview];
// to set map draw trip overview with sortedPoisArray
[self.mapVC setMapDrawTripOverviewWithPois: sortedPoisArray arrivedPois:nil];
```

Custom The Map Trip Overview (declared in IDMapViewControllerDelegate):

```
- (UIColor*) mapColorForTripOverviewRoute
{
```

```
        return [UIColor blueColor];
}
// to customize trip over view circle
// implement mapColorForTripOverviewCircle
- (UIColor*)mapColorForTripOverviewCircle
{
        return [UIColor orangeColor];
}
// to customize trip over view arrived circle
// implement mapColorForTripOverviewArrivedCircle
- (UIColor*) mapColorForTripOverviewArrivedCircle
{
        return [UIColor greenColor];
}
```

17. Present User Location With Bubble

Call the method to in order to customize the user location bubble when the user is outdoors.

```
- (void)presentUserLocationWithBubble:(UIView*)aView;
```

The method can be implemented for parking notation as well.

Example:

```
// create bubble custom view
UIView* parkingSpot = [[UIView alloc] initWithFrame:CGRectMake(0, 0, 220, 180)];

// add imageView
UIImageView* imageView = [[UIImageView alloc] initWithFrame:parkingSpot.frame];
imageView.image = [UIImage imageNamed:@"parking__spot.png"];
[parkingSpot addSubview: imageView];

// add add confirm button
UIButton* confirmButton = [[UIButton alloc ] initWithFrame:CGRectMake(50, 125, 120, 38)];
[confirmButton addTarget:self action:@selector(didConfirmParkingSpot)
forControlEvents:UIControlEventTouchUpInside];
[parkingSpot addSubview:confirmButton];

//Present bubble
[self.mapVC presentUserLocationWithBubble:parkingSpot];
```

To REMOVE the Bubble View call presentUserLocationWithBubble: method with nil parameter

```
[self.mapVC presentUserLocationWithBubble:nil];
```

18. Start Location Tracking

Call the method startUserLocationTrack or startUserLocationTrackWithDelegate to start the position engine. And call this method when update and initialization is done (see kIDDataUpdateDone state) or after it is done.

```
[IDKit startUserLocationTrack];
```

or

```
[IDKit startUserLocationTrackWithDelegate:self];
```

In order to receive user location updates add the IDLocationListener protocol declaration to the supported protocols:

```
@interface MainViewController () <IDLocationListener>
@end
```

Add the call to register delegation in case it is not yet set:

```
[IDKit registerToLocationListenerWithDelegate:self];
```

As result to the delegation set and the call to the method startUserLocationTrackorstartUser-

LocationTrackWithDelegatethe delegation IDLocationListener are called (i.e when location updates are available).

```
- (void)updateUserLocationWithLocation:(IDUserLocation *)aLocation;
- (void)locationDetectionStatusChanged:(IDLocationDetectionStatus)aStatus;
- (void)regionEventChangedForCampusId:(NSString*)aCampusId
withEvent:(IDRegionEventType)anEventType;
- (void)regionEventChangedForFacilityWithID:(NSString*)aFacilityId
campusId:(NSString*)aCampusId
withEvent:(IDRegionEventType)anEventType;
```

19. Stop Location Tracking

Call this method to stop user location tracking, i.e. when the app is sent to the background.

```
[IDKit stopUserLocationTrack];
```

20. Reset Location Tracking

Call the method to reset the user location tracking i.e. when the app comes out of sleep mode (background).

```
[IDKit resetUserLocationTrack];
```

21. Instruction Controller

In the view controller that holds the map, implement the IDInstructionsControllerDelegate. Add a property that will hold the IDInstructionsController.

```
@interfaceMainViewController () <IDNavigationDelegate, IDInstructionsControllerDelegate>
@property (nonatomic, strong) IDInstructionsController *InstructionController;
@end
```

In the viewDidLoad method call getInstructionController and assign the return value to the instructionController property that was declared earlier.
Set the instruction controller delegate to self.
Add the instruction controller and its child view to the base view controller hierarchy.

```
- (void)viewDidLoad {
        // get the instruction view controller
        self.InstructionController = [IDKit getInstructionController];

        // set the delegate to the map
        [self.InstructionController setDelegate:self];

        // add the map view controller as a child to the base view controller
        [self addChildViewController:self.InstructionController];

        // add the map view to the view hierarchy
        [self.viewaddSubview:self.InstructionController.view];
}
```

21.1. Instruction Controller Delegate

Add and implement the methods of the IDInstructionsControllerDelegate

```
- (void)stopNavigationTapped
{
// call the method that stop navigation
}

- (void)showInstructionsList
{
// call the method to present display Itinerary Route
// explanation below
}
```

21.2. Display Itinerary Route

In order to display the itinerary route, call the instruction method:

```
- (NSArray *)getInstructionsList
```

The method returns an array of dictionary objects, which represent an instruction in the route.

```
- (void)showInstructionsList
{
	NSArray *instructionList = [self.instructionController getInstructionsList];
	InstructionListViewController *instructionListVC = [self.storyboard instantiateViewControllerWithIdentifier:@"
	instructionListVC"];
	instructionListVC.instructionList = instructionList;
	[self.navigationController pushViewController:instructionListVC animated:YES];
}
```

Table view delegate method looks like this:

```
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)index-
Path
{
	static NSString *CellIdentifier = @"InstructionCell";
	UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier forIndex-
	Path:indexPath];
	/*
	@{ "id" : IDNavInstructions enum type,
	"text" : instruction text,
	"image" : instruction UIImage}
	*/
	NSDictionary *instDic = self.instructions[indexPath.row];
	if (nil != instDic) {
	cell.textLabel.text = instDic[@"text"];
	cell.imageView.image= instDic[@"image"];
	}
	return cell;


}
```

22. Navigation

In order to receive user navigation updates add the IDNavigationDelegate protocol declaration to the supported protocols:

```
@interface MainViewController () <IDNavigationDelegate>
@end
```

22.1. Start Navigation to Location

Need to store from and to location in your class, than call:

```
[IDKit callServerRouteAPIFrom:fromLocation toLocation:toLocation];
```

Once the **call server route** method is called, the delegation methods will be called.
Implement the IDNavigationDelegate delegation methods:

```
- (void)isServerAPICallSuccess:(BOOL)aStatus typeError:(NSInteger)type {
	[IDKit startNavigateToLocation:aPoi.location withOptions:kNavigationOptionRegular andDelegate:self];
}

- (void)updateWithInstruction:(NSDictionary *)anInstruction andStatus:(IDNavigationStatus)aStatus
{
	// call the method that update the instruction controller with the current instruction and status
	[self.instructionController updateWithInstruction:anInstruction andStatus:aStatus];
}
```

```
- (void)playInstructionSound
{
        // call the method that instruct the instruction controller to play immediately the current instruction
        [self.instructionController playInstructionSound];
}
- (void)navigationUpdateWithStatus:(IDNavigationStatus)aStatus
{
        // call the method that update the instruction controller with 0the current status,
        // that is important to allow the instruction controller to act according to the navigation status
        // for situations like reroute, arrive to destination and etc.
        switch (aStatus) {
        case kNavigationStart:
        [self.instructionController presentInstructionFromOriginY:kInstructionAnimationOrigin
        toPositionY:kInstructionAnimationPosition];
        break;
        case kNavigationStopped:
        case kNavigationEnded:
        [self.instructionController dismissInstruction];
        break;
        default:
        break; }
}
```

22.2. Parking Location

+ (IDLocation*)setParkingLocation:
The Method sets a parking location.

```
IDLocation* currentLocation = [IDKit getUserLocation];
currentLocation.facilityId = nil;
[IDKit setParkingLocation:currentLocation];
```

22.3. Start Navigation to Parking Location

There are two ways to navigate to parking location
1. Get the saved parking location by using +getParkingLocation method :

```
IDLocation* parkingLocation = [IDKit getParkingLocation];
```

2. Start navigation to the parking location :

```
[IDKit startNavigateToLocation:parkingLocation
withOptions:kNavigationOptionRegular
andDelegate:self];
Or
[IDKit navigateToParkingLocationWithOptions:kNavigationOptionRegular andDelegate:self];
```

22.4. Stop Navigation

In the delegation method:
- (void)stopNavigationTapped of IDInstructionsControllerDelegateprotocol call the stop navigation method.

```
- (void)stopNavigationTapped
{
        [IDKit stopNavigation];
}
```

23. Get CMS Geofences

```
+ (NSArray*)getAllGeofencesList
```
The method returns all the CMS geofence objects (IDGeofence) in array.
```
NSArray * allCMSgeofences = [IDKit getAllGeofencesList];
```
```
+ (void)registerForGeofenceTypes:withDelegate:
```
The method registers the IDGeofenceDelegate delegate for custom geofences types
```
[IDKit registerForGeofenceTypes:@[@"banner"] withDelegate:self];
```
```
+ (void)unregisterForGeofenceTypes:withDelegate:
```
The method unregisters the IDGeofenceDelegate delegate for custom geofences types.
```
[IDKit unregisterForGeofenceTypes:@[@"banner"] withDelegate:self];
```

24. Local Notification and Campus Region Monitoring

The user can be prompted and reminded to open and use the application by enabling the campuses and facility region monitoring. In these cases, when the application is not running or is running in the background, a local notification can be pushed.
Turn on region monitoring:
Call the following method with positive parameter after that the data module was updated and initialized.
```
+ (void)monitorCampusesRegion:(BOOL)mode
- (void)dataUpdateStatus:(IDDataUpdateStatus)status
{
        switch (status) {
        case kIDDataUpdateDone:
        [IDKit monitorCampusesRegion: YES];
        break;
        }
}
```

25. Local Notification Text Customization

It is possible to customize the text notification pushed to the user by setting the IDKit setLocalNotification method
```
[IDKit setLocalNotificationText:@"welcome to the mall"];
```

26. Enable Background Operation

In In order to enable location and position while the app is in the background, use the following steps:
In the Xcode, select your application target and tap the capabilities tab:
• In the Maps section move the switch to ON.
• Open the Background Modes and move the switch to ON,
then mark the following squares:
• Audio and Air-Play.
• Location Updates.
• Use Bluetooth LE.
Also at the AppDelegate.m you can notify the SDK when the app enters background / foreground

```
- (void)applicationDidEnterBackground:(UIApplication *)application
{
        [IDKit moveToBackgroundAndContinueScanning:YES / NO];
        // Or
        [IDKit StopUserLocationTrack];
}

- (void)applicationWillEnterForeground:(UIApplication *)application
```

```
{
        [IDKit moveToForeground];
        // Or
        [IDKit resetUserLocationTrack];
}
```

27. More details

Read more about usage in “iOS SDK API” document