

# Footmarks iOS SDKv4.0 Quickstart Guide

Version 2

**Patent Notice**: Various use cases and technical information provided herein are protected by a series of provisional and non-provisional patent applications.

Copyright © 2013 Footmarks, Inc. – All Rights Reserved Worldwide – Patent Pending

This document may not be reproduced without the expressed permission of Footmarks, Inc.

Any reproduction, without authorization, is an infringement of Footmarks copyright.

This document contains confidential and proprietary information.

Table of Contents

Footmarks iOS SDKv3.4 Quickstart Guide 1

Document Overview 3

Step-by-step Project Integration 4

Step-By-Step Logic Integration 8

1. Setup the AppDelegate 8

2. Authenticating to the Footmarks Service 9

3. Begin Scanning for Footmarks Beacons 9

Add Required Frameworks 10

Add Required Frameworks 11

Building with iOS 6 12

# Document Overview

This document will describe how to integrate the Footmarks iOS SDK. Developers can use this document in conjunction with the provided sample apps to quickly get up and running. Once the Footmarks SDK is integrated with your app, you can then refer to the SDK Detailed Overview document to get a complete understanding of how the SDK functions.

# Step-by-step Project Integration

1. Drag libFootmarksSDK\_VX.a into the Frameworks section of your Xcode project
   * Check “Copy items into destination group’s folder (if needed)”
   * Click the “Finish” button
2. Add Footmarks\_SDK\_VX.h to your project
3. Include the Framework’s CoreBluetooth & CoreLocation in your app.

***Note****: See Figure 1 below to see how your Project Navigator should look at this point.*

1. Within your app’s Target->Capabilities->Background Modes section, add the modes:
   * Uses Bluetooth LE accessories

***Note****: See Figure 1 below to see what your Target’s Capabilities section should look like at this point.*

1. Include the linker flag “-all\_load” (steps below)
   * Click your project in the Project Navigator
   * Click your projects Target
   * Click the Build Settings tab
   * Find the Setting named “Other Linker Flags” and add the linker flag “-all\_load”

***Note****: See Figure 2 below to see what your Target’s “Other Linker Flags” attribute should look like*

1. Add the property key *NSLocationAlwaysUsageDescription* in your projects plist file.

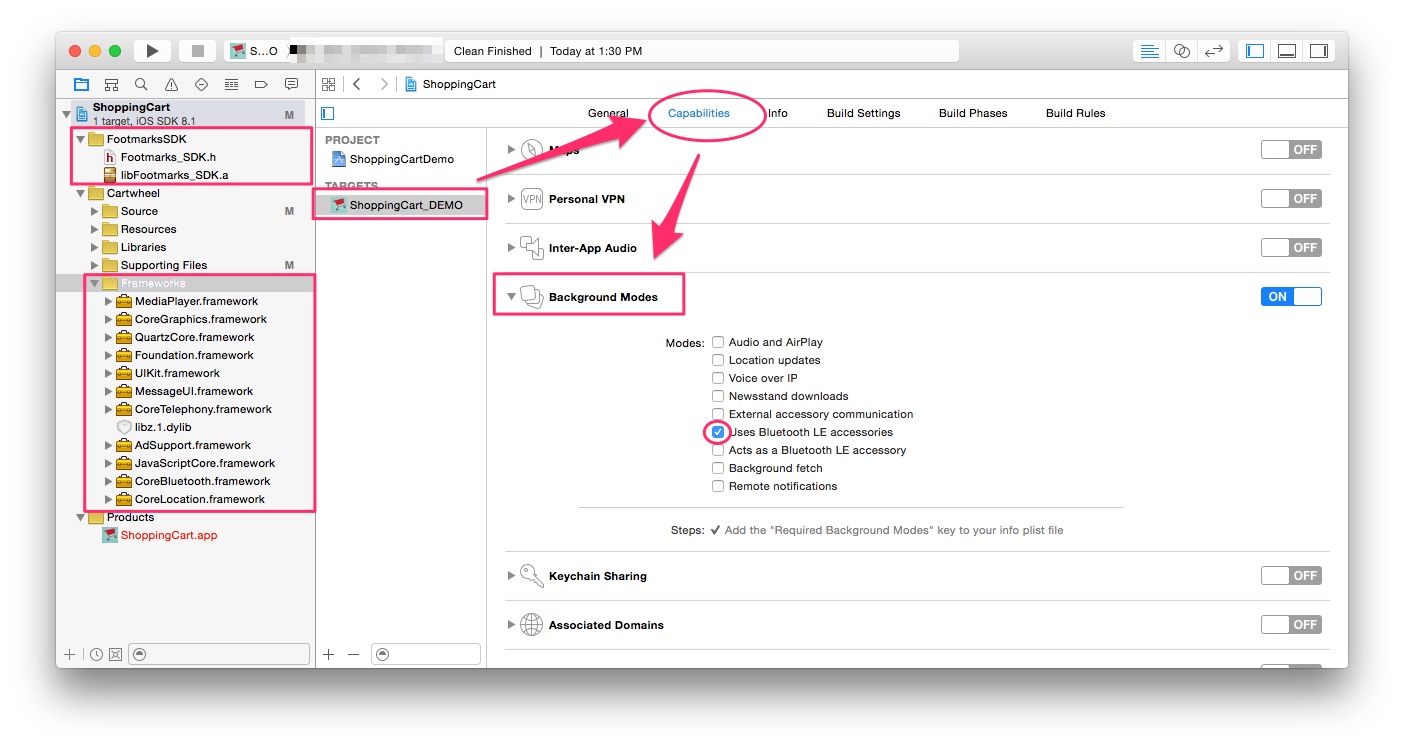
* Devices running iOS 8 require this property in order to detect beacons. The value associated with this key is defined as follows:

*“Describes the reason why the app accesses the user’s location information. When the system prompts the user to allow access, this string is displayed as part of the alert panel”*

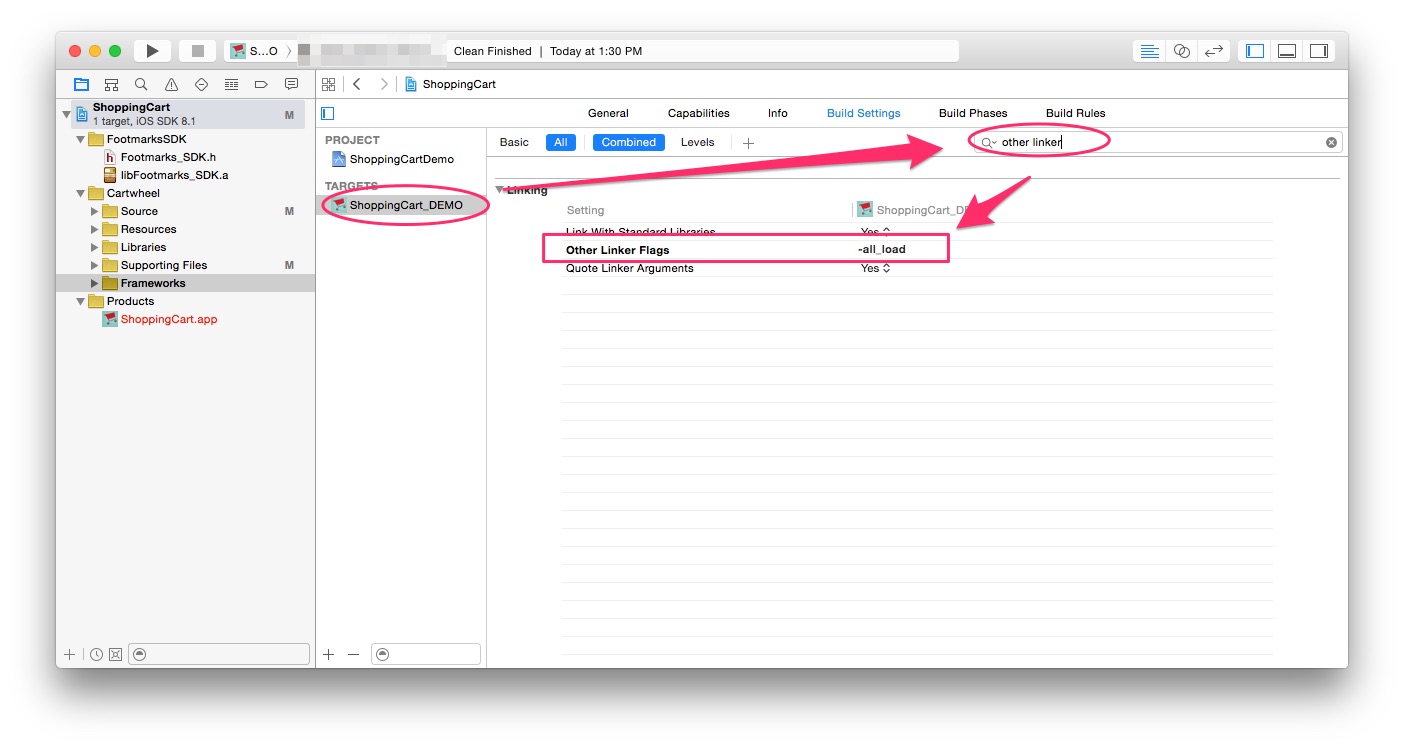
***Note****: See Figure 3 below to see an example of a plist file containing the NSLocationAlwaysUsageDescription* property

* **Footmarks SDK Project integration is now complete**

**Figure 1**



**Figure 2**



**Figure 3**

# ::::::::::private:var:folders:j6:s60_882x0gg7k5hpjy24kj400000gp:T:com.skitch.skitch:DMDA12C8D03-1AF1-43A4-BAC3-E3B5F751FF06:Screen_Shot_2015-02-23_at_1_39_46_PM.jpg

# Step-By-Step Logic Integration

This section details the quick process of integrating the necessary SDK logic to get your app up and running.

**\*Note**: An experienced iOS developer should take a look at step 2. After this, the rest may be unnecessary and you could simply mimic the functionality that is shown in the sample app packaged with the iOS SDK.

## 1. Setup the AppDelegate

The Footmarks SDK requires your AppDelegate to comply with the following protocols:

* FMBeaconManagerDelegate
* FMExperienceManagerDelegate.

**\*Note**: **Configuring these delegates in the AppDelegate is important because it allows the app to always receive experiences (i.e. callbacks) when the app is not running.**

Configure the AppDelegate as shown below:

a. Add FMBeaconManagerDelegate and FMExperienceManagerDelegate to AppDelegate

@interface AppDelegate ()<FMBeaconManagerDelegate, FMExperienceManagerDelegate>

b. Add the required protocol methods

- (void)bluetoothDidSwitchState:(CBCentralManagerState)state {}

- (void)locationServicesFailedWithError: (NSError \*)error {}

- (void)didCompleteExperiences: (NSArray\*) experiences {}

c. Call the following method within the AppDelegate’s application:didFinishLaunchingWithOptions: method:

[(FMAccount \*)[FMAccount sharedInstance] application:application didFinishLaunchingWithOptions:launchOptions];

## 2. Authenticating to the Footmarks Service

Authentication is done using the FootmarksAccount’s loginToFootmarksServer: method. The necessary steps are shown below.

a. Add FootmarksAccountDelegate to UIViewController that will invoke the loginToFootmarksServer() method

@interface ViewController : UIViewController <FootmarksAccountDelegate>

b. Add FootmarksAccountDelegate methods to your UIViewController

- (void) loginSuccessful

- (void) loginUnsuccessful: (NSString\*)error

c. Set FootmarksAccountDelegate and Invoke Login method

[[FootmarksAccount sharedInstance] setAccountDelegate:self];

[[FootmarksAccount sharedInstance] loginToFootmarksServer:@"appKey" andAppSecret:@"appSecret" andUserId:@"yourUsersUniqueId"];

## 3. Begin Scanning for Footmarks Beacons

Begin scanning in the loginSuccessful() callback method:

- (void) loginSuccessful

{

[[FMBeaconManager sharedInstance] startScanningForFMBeaconsWithError:&error];

}Configuring Smart Profile

This section explains how to enable the Footmarks SDK to begin creating Smart Profiles (SP) on users. SPs are created by retrieving various data on the device and sending this data to the Footmarks Server for processing. SPs allow the Footmarks Platform to create better targeted experiences for each user. Also, more powerful analytics will be calculated and displayed in the Footmarks Platform.

## Add Required Frameworks

Add the following Frameworks if you do not already have them in your project:

* AdSupport.framework
* MessageUI.framework
* CoreTelephony.framework
* CoreLocation.framework
* libz.1.dylib

## ::::::::::private:var:folders:j6:s60_882x0gg7k5hpjy24kj400000gp:T:com.skitch.skitch:DMD62883FF8-4616-4A9C-B16B-FFAA32CA14FA:Screen_Shot_2015-02-23_at_2_00_09_PM.jpgAdd Required Frameworks

Call FMAccount’s useSmartProfilesInApplication() method in AppDelegate’s didFinishLaunchingWithOptions() method (shown below). This will enable Smart Profiles upon launch of the app.

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions

{

[[FMAccount sharedInstance] useSmartProfilesInApplication:application withLaunchOptions:launchOptions];

return YES;

}

# Building with iOS 6

The Footmarks SDK is designed to run on devices with iOS 7 or above. Apple introduced the majority of its beacon functionality in iOS 7 (i.e. “iBeacon methods”).

However, there is a way to prevent older devices from crashing that run an app with the Footmarks SDK. Change the CoreLocation framework from ‘Required’ to ‘Optional’ in the Application Target’s Build Phases. Also, check the user’s iOS version prior to calling any of the SDK methods.

