inBeacon Cordova SDK guide

Document date	Changelog
29-jun-2016	Initial version, based on Cordova component 1.0.0 (iOS sdk 1.3.9 and android sdk1.5.0)

This document describes the latest version of the inBeacon Cordova SDK. Additionally you can refer to our github to view the latest readme docs:

https://github.com/inbeacon/cordova-plugin-inbeacon

Authors: Daniel Fortuyn

Padualaan 8
UtrechtInc Room W129
3584 CH Utrecht
The Netherlands

inBeacon SDK’s

Integrating inBeacon is now easier than ever. It’s a matter of pasting a few lines of code to integrate our SDK into your project. The inBeacon mobile SDK’s are available for the following platforms:

● inBeacon iOS Xcode SDK	(available)
● inBeacon Android java SDK	(available)
● inBeacon Xamarin SDK	(available)
● inBeacon iOS Appcelerator/Titanium SDK	(available)
● inBeacon cordova SDK	(available)

For other environments and platforms, the inBeacon device/mobile API can be addressed directly.

Getting Started

Both the Android and the iOS parts of this component can be implemented in a Cordova project by following the regular instructions for Android and iOS native apps as provided by inBeacon on the SDK documentation page . These instructions assume you already have an account at inBeacon and are in possession of an Client ID and Client Secret. You can find your client-ID and client-Secret in your account overview. See https://console.inbeacon.nl/accmgr

Implementation example

Please refer to the cordova developers site to create a new cordova project.

Consquently add the inbeacon plugin like so:

```
cordova plugin add https://github.com/inbeacon/cordova-plugin-inbeacon.git --variable
INBEACON_CLIENTID="your-clientid" --variable INBEACON_SECRET="your-secret"
```

Note the possibility to include the clientId and the clientSecret as parameters

There is also a complete example project available at

https://github.com/inbeacon/inbeacon-cordova-example

SDK methods

- initialize
- refresh
- attachUser
- detachUser
- checkCapabilitiesAndRights
- setLogLevel (iOS only)
- checkCapabilitiesAndRightsWithAlert (iOS only)
- getInRegions (iOS only)
- getBeaconState (iOS only)
- checkCapabilitiesAndRights (android only)

cordova.plugins.inBeacon.initialize

Initialize the SDK with your clientID and clientSecret. These credentials are used for communication with the server.

```
var userInfo = {
    clientId : 'clientId obtained from InBeacon',
    secret   : 'secret obtained from InBeacon'
};

cordova.plugins.inBeacon.initialize(userInfo, function () {
        console.log('Succesfully initialized inBeacon API');
    }, function (error) {
        console.error('inBeacon initialization failed ' + error);
    });
```

cordova.plugins.inBeacon.attachUser

Attach local userinformation to inbeacon. For instance, the user might enter account information in the app. It is also possible not to attach a user, in that case the device is anonymous.

Note: *User account information is not stored by the SDK so you'll need to call attachUser every time the app starts (after SDK initialization)*

There are a lot of fields available for userinformation, however none of them are required. Supply the necessary information.

field	Description
name	Full user name, both first and family name. Example 'Dwight Schulz'
email	User email. Example: 'dwight@a-team.com'
customerid	This can be any identifier used for identifying your customer, like a CRM client id or a loyalty card number. If the inBeacon backend communicates with a CRM system, this ID can be used to identify the customer. For inBeacon, this ID is a black box.
advertising_id	fill this with the IDFA number. Note: you need to specify the use of the IDFA on app submission. See https://developer.apple.com/news/?id=08282014a
gender	Either 'male' or 'female' is accepted

address	Street address and number
zip	zip / postal code
city	
country	ISO 2 letter country code
birth	Date of birth (string format yyyymmdd)
phone_mobile	
phone_home	
phone_work	
social_facebook_id	Facebook ID of user
social_twitter_id	Twitter ID of user, like @woenz
social_linkedin_id	Linkedin ID of user
avatar	URL to user avatar (png or jpg)

```
var userInfo = {
    name  : 'Dwight Schultz',
    email : 'dwight@ateam.com'
};
cordova.plugins.inBeacon.attachUser(userInfo, function () {
        console.log('Succesfully attached user');
    }, function () {
        console.error('Failed to attach user');
    });
```

Note that attached user info is not stored by the SDK. On app restart, you need to attach the user again.

cordova.plugins.inBeacon.detachUser

Remove current user info connected to the device. From now only anonymous info is send to inBeacon server.
`cordova.plugins.inBeacon.detachUser();`

cordova.plugins.inBeacon.refresh

Obtain fresh trigger and region information from the inBeacon platform. Best practice is to call this when the app is

- started AND
- returned to the foreground so info is kept updated. Internally, the SDK automatically refreshes when a beacon area is entered.
- Additionally it is imported to note that data will only be transfered when something has changed in the configuration. Otherwise the device is told it is up to date.

```
cordova.plugins.inBeacon.refresh();
```

cordova.plugins.inBeacon.checkCapabilitiesAndRights

Check to see if the app has the rights to run location and notification services. Returns BOOL NO if there is a problem.

```
cordova.plugins.inBeacon.checkCapabilitiesAndRights()
```

cordova.plugins.inBeacon.beaconState

Get an array of all actual beacons within view, including their respective distance and proximity state. This method returns raw data without any filtering.

The returned beaconState array has 1 entry for each beacon in view. Each array item has the following data:

Field	Description
`uuid`	beacon UUID value
`major`	beacon major value
`minor`	beacon minor value
`proxes`	Timestamps of all proximities last seen, format: <proximity>: <time last seen> Proximities are F, N and I.
`rawdist`	raw beacon distance in metres
`rawprox`	raw proximity (F, N, I or U) U = undefined, beacon currently not visible

Because beacon information is updated once per second, it is not useful to obtain the beaconstate more often.

Receiving inBeaconSDK events

Event	Description
inbeacon.enterregion	*user entered a region*
inbeacon.exitregion	*user left a region*
inbeacon.enterlocation	*user entered a location*
inbeacon.exitlocation	*user left a location*
inbeacon.regionsupdate	*region definitions were updated*
inbeacon.enterproximity	*user entered a beacon proximity*
inbeacon.exitproximity	*user left a beacon proximity*
inbeacon.proximity	*low level proximity update, once every second when beacons are around*
inbeacon.appevent	*defined in the backend for application events*
inbeacon.appaction	*defined in the backend to handle your own local notifications*

Example:

```
document.addEventListener('inbeacon.enterregion', handleEnterRegion, false);

function handleEnterRegion(event){
    console.log('Event name:' + event.name);
    console.log('Event data:' + JSON.stringify(event.data));
}
```

Android details

Add required permissions to the manifest

In the Properties/AndroidManifest.xml, add the following permissions:

```
	<uses-permission android:name="android.permission.BLUETOOTH" />
	<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
	<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
	<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
```

Also add the following inside the <application></application> block:

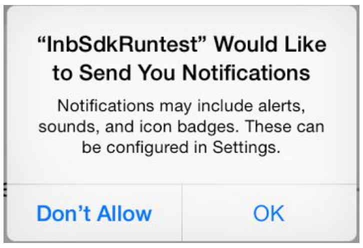
```
	<receiver android:name="org.altbeacon.beacon.startup.StartupBroadcastReceiver">
		<intent-filter>
			<action android:name="android.intent.action.BOOT_COMPLETED" />
			<action android:name="android.intent.action.ACTION_POWER_CONNECTED" />
			<action android:name="android.intent.action.ACTION_POWER_DISCONNECTED" />
		</intent-filter>
	</receiver>
	<service android:enabled="true" android:exported="false" android:isolatedProcess="false"
	android:label="beacon" android:name="org.altbeacon.beacon.service.BeaconService" />
	<service android:name="org.altbeacon.beacon.BeaconIntentProcessor" android:enabled="true"
	android:exported="false" />
```

iOS details

Permissions

When the SDK is integrated, there are a few permissions needed from the user. If the app already needed those permissions before integration of the SDK,, there will be no change (they will not be requested twice). These permissions are requested once after installation when the app is run (and the inBeacon SDK is initialized).

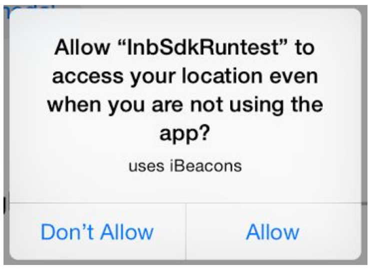
1. Application would like to send you notifications



"InbSdkRuntest" Would Like to Send You Notifications

Notifications may include alerts, sounds, and icon badges. These can be configured in Settings.

2. Application would like to access your location even when you are not using the app.

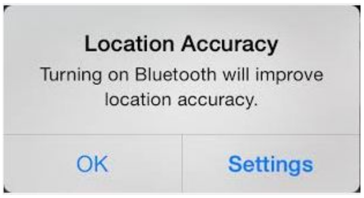


Allow "InbSdkRuntest" to access your location even when you are not using the app?

uses iBeacons

   It is also possible to run the SDK in "Geofenced Location Authorisation" mode (SDK 1.3.5). In this case the SDK asks for "when in use" access only, and asks for "even when you are not using the app" mode at the moment the user enters one of the predefined Geofences.

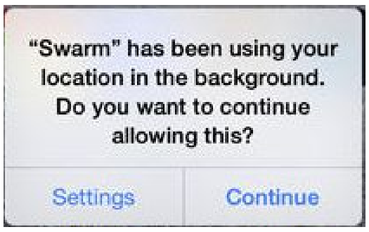
3. When bluetooth is turned off:



Location Accuracy

Turning on Bluetooth will improve location accuracy.

4. After running a few days, the user gets a notification that the app is looking for beacons or geofences (even when no beacons or geofences are actually found). We found out that this message is always given in combination with message 2), and is not related to the actual location or iBeacon features used in the app.



"Swarm" has been using your location in the background. Do you want to continue allowing this?

   When the SDK is running in "Geofenced Location Authorisation" mode, this message is not send until after the user has crossed one of the Geofences and has accepted "even when you are not using" location scanning.

Background mode

There are three fundamentally different ways to run the inBeacon SDK:

Mode	Description
full background mode (special cases only)	For iOS, full background mode requires extra location background settings so the app is able to run continuously in the background during iBeacon ranging. **Advantages:** When users approach an iBeacon in Near or Immediate proximity, all inBeacon triggers and other functionality is fully supported. During the complete stay of the user in the iBeacon location, the app will monitor the range to all defined iBeacons. **Disadvantages:** ● the app description in the app-store needs to include the following: *"Note: Continued use of GPS and <app name> running in the background can dramatically decrease battery life. <app name> will automatically shut down if you run it in the background and leave <description of location where ibeacons are used>."* ● Possibility of initial app rejection by iTunes connect (apple appstore). However in the past we were able to get all apps approved, even with full background mode ON. ● The app uses a bit more battery power when inside beacon regions. Because location monitoring is set for least-accurate, GPS is not used by the SDK. We found that the decrease of battery life is negligible.
Restricted background mode (recommended)	If you don't specify background modes, the app runs in restricted background mode. Restricted background mode notices ALL beacon regions enter/exit, even when the app is terminated or suspended (just like full background mode), but stops checking beacon proximity in the background 3 minutes after entering the region (location) This means that if a user is approaching an iBeacon within a region more than 3 minutes after entering the region, triggers will no longer work when the app is in the background. **Advantages:** ● App submission to the app store is without issues about background location scanning ● Battery power use is limited even more. ● **Disadvantages:** This limits the app to a maximum of 3 minutes background processing when the app enters or leaves a beacon region.
Geofenced Location Authorisation mode	In this mode the app starts by only asking for "when in use" location permissions. Only when the user enters a predefined geofenced region, the app will ask for full (background) location permissions. Geofenced Location Authorisation is defined on the inBeacon backend. **Advantages:** ● all advantages of restricted background mode ● only ask full permissions to the subset of your users that are within the geografic region where beacons are used. **Disadvantages** ● all disadvantages of restricted background mode

Using full, restricted background mode or Geofenced Location Autorisation mode depends on your specific situation.

App store submission

Full background mode submission notes
Only for full background mode special app store submission rules apply:

iTunes application notes
Because we use iBeacon ranging in the background you need to include a note in your app description:

> Please include the following battery use disclaimer in your Application Description:
> *"Continued use of GPS running in the background can dramatically decrease battery life."*

We were able to get applications approved with the following text:

in English:

> *Note: Continued use of GPS and <app name> running in the background can dramatically decrease battery life. <app name> will automatically shut down if you run it in the background and leave <description of location where ibeacons are used>*

in Dutch:

> *Opmerking: Langdurig gebruik van GPS en <app naam> kan de levensduur van de accu drastisch verminderen. <app naam> zal automatisch afsluiten zodra u <omschrijving van de locatie waar de ibeacons worden gebruikt> verlaat.*

Update Info.plist
iOS requires a text that explain why the app should be allowed to use the location services. Add the following keys to the Info.plist file and supply a string value with an explanation:

```
<key>NSLocationAlwaysUsageDescription</key>
<string>To detect iBeacons</string>
<key>NSLocationWhenInUseUsageDescription</key>
<string>To detect iBeacons</string>
```

If you want the inBeacon platform to run in continuous background mode see above), go to the bottom of the Info.plist file editor (section 'Background modes'), enable 'Enable Background Modes' and check 'Location updates'.

```
<key>UIBackgroundModes</key>
<array>
       <string>location</string>
</array>
```

Include audio resources

When using customized sounds with notifications, make sure to copy the audio files from the iOS SDK into your app. To do that, find the **`./resources/*.caf`** files in the iOS SDK and copy those files to the **`./Resources`** folder in your iOS project. Next, right-click on the files in Xamarin and make sure that 'Build Action' is set to 'BundleResource'. The audio files can also be found in the sample project included in the component.

Changelog

SDK version 1.0.0

initial