



- [HOME](#)
- [TUTORIALS](#)
- [GITHUB](#)
- [FAQ](#)

## Quickstart for iOS

### Introduction

Beaconstac is an easy way to enable proximity marketing and location analytics through an iBeacon-compliant BLE network. The SDK includes api to start monitoring for beacons, send callbacks when sufficiently close to beacons and when corresponding proximity rules are triggered. iBeacons are Bluetooth Low Energy devices which keep emitting their identity at a predefined frequency, say every 200 milliseconds, also known as advertising interval. These signals are emitted at a fixed transmission power which is usually measured in dBm (dB-milliWatt). Typical values are -4 dBm, -8 dBm, -16 dBm. These signals can be detected by mobile device upto a distance of 30 - 70m.

In a typical deployment scenario where multiple beacons could be deployed as close as 5 meters from each other, a mobile device would receive signals from more than 1 beacon at a time. All the beacons would have the same UUID which is common to the organization. However, each one would have a different identity as defined by two other numbers: Major and Minor.

UUID: A 16 byte number typically common across an organisation Major: A 2 byte number (0 to 65,535) Minor: A 2 byte number (0 to 65,535)

The combination of UUID, Major and Minor should always be unique thus making a beacon easily identifiable from other.

### Beacon Region Monitoring

A Beacon region is a space which is formed by union of regions of all beacons with same UUID. Beaconstac SDK can monitor entering and exiting from Beacon region while the iOS app is in the foreground as well as in the background. Whenever the user enters or exits a beacon region, the SDK sends a delegate callback to the app along with the identifier for the region. Beacon region monitoring can be useful in

identifying when the user is entering or exiting a venue where beacons are deployed, say entering a stadium or exiting an airport area.

## Beacon Ranging

Beacon ranging is the process of detecting all the individual beacons which are detectable by mobile device inside a Beacon region. For example, if there are 5 beacons deployed in a stadium, the ranging call back from SDK would return a list of all the 5 beacons along with their respective signal strength. This could be useful in identifying relative distance from each beacons.

## Beacon CampOn/Exit

The concept of Beacon CampOn is unique to Beaconstac SDK. The SDK can figure out if the user has spent enough time within 2 - 4 meters of any beacon and call it a CampOn to this beacon. In venues where the mobile device can range more than one beacon, the SDK heuristically figures out which is the one closest to the user. When the SDK notices that the user has moved significant distance from the beacon and sufficient time has passed for the user to not return to it, the SDK calls it a beacon exit. In both the cases, a delegate callback is sent from SDK to app along with reference to the beacon and the message corresponding to the camped-on beacon only needs to be displayed to the user. In the example of museum, if one beacon is deployed next to each artifact, the SDK camps on to individual beacons as the user moves from one artifact to the other. The corresponding description/media can be shown to the user one at a time, thus reducing ambiguity about the beacon of interest.

## Triggering Rules

A Rule is a combination of multiple conditions, the primary condition being proximity to a beacon. Other conditions include time spent near the particular beacon or other user attributes configurable via Custom attributes as described further.

So, a Rule can look like:

1. If the user has camped onto Beacon with Major: 231 Minor:4328
2. AND, if the user has spent more than 3 minutes in the proximity of this beacon
3. AND, if the user is a male
4. AND, if the date is between 5th and 15th of May

When all the conditions in a rule get satisfied, the SDK sends a callback to the app with a reference to the Rule name and other actions as configured by the user on the Admin Console.

## Custom Attributes:

In the above set of conditions, #1 and #2 are intrinsic to the SDK while #3 and #4 can be custom defined by the user on the Admin Console. Likewise, the user can define any property on the console. The app code can then keep updating these properties by calling SDK api based on user behaviour. When the condition set on portal matches the property updated in the app, the rule is triggered.

Presently, 3 types of attributes are possible, viz, String, Number and Date.

## Actions:

For every rule trigger, you can configure a set of actions to be executed. Presently, there are 5 types of actions that can be configured from the portal: Text Alert, Webpage, Cards, Webhook and Custom.

1. Text Alert: This is the basic action type, which shows a popup alert with a text message when the app is open
2. Webpage: This allows you to enter a url on the Admin Console and the corresponding webpage is opened when the rule is triggered
3. Cards: Cards are a way of showing immersive full screen content configurable from the Admin Console
4. Webhook: This is a way to make calls to third party services along with given json
5. Custom: This allows one to add fixed json payload from the console. When the rule gets triggered, the payload is passed on from the SDK to the app, which can parse it and do any possible task.

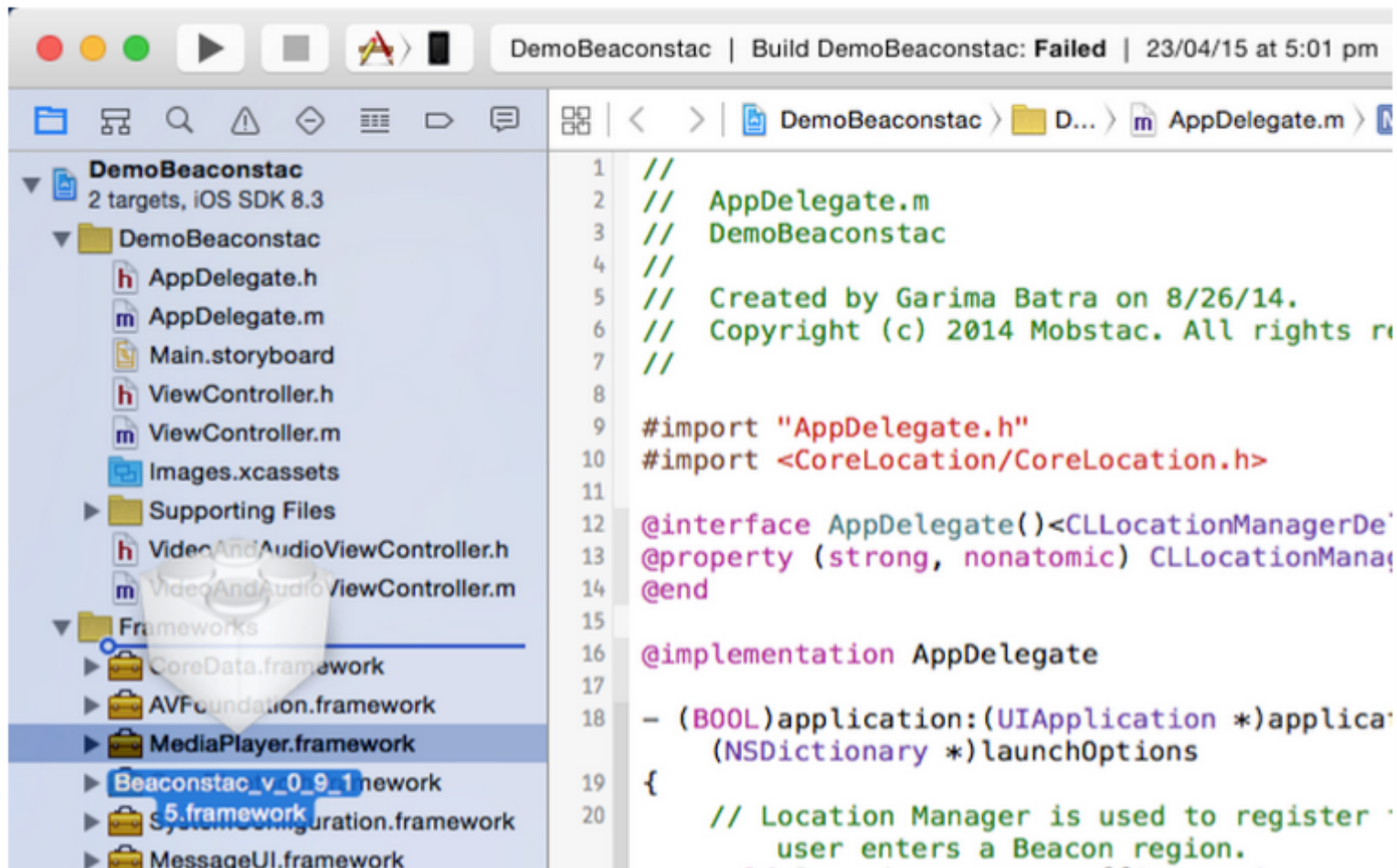
For our first app, to get started we want to make a simple app which shows a text alert when a user comes close to one of your beacons.

## Prerequisites:

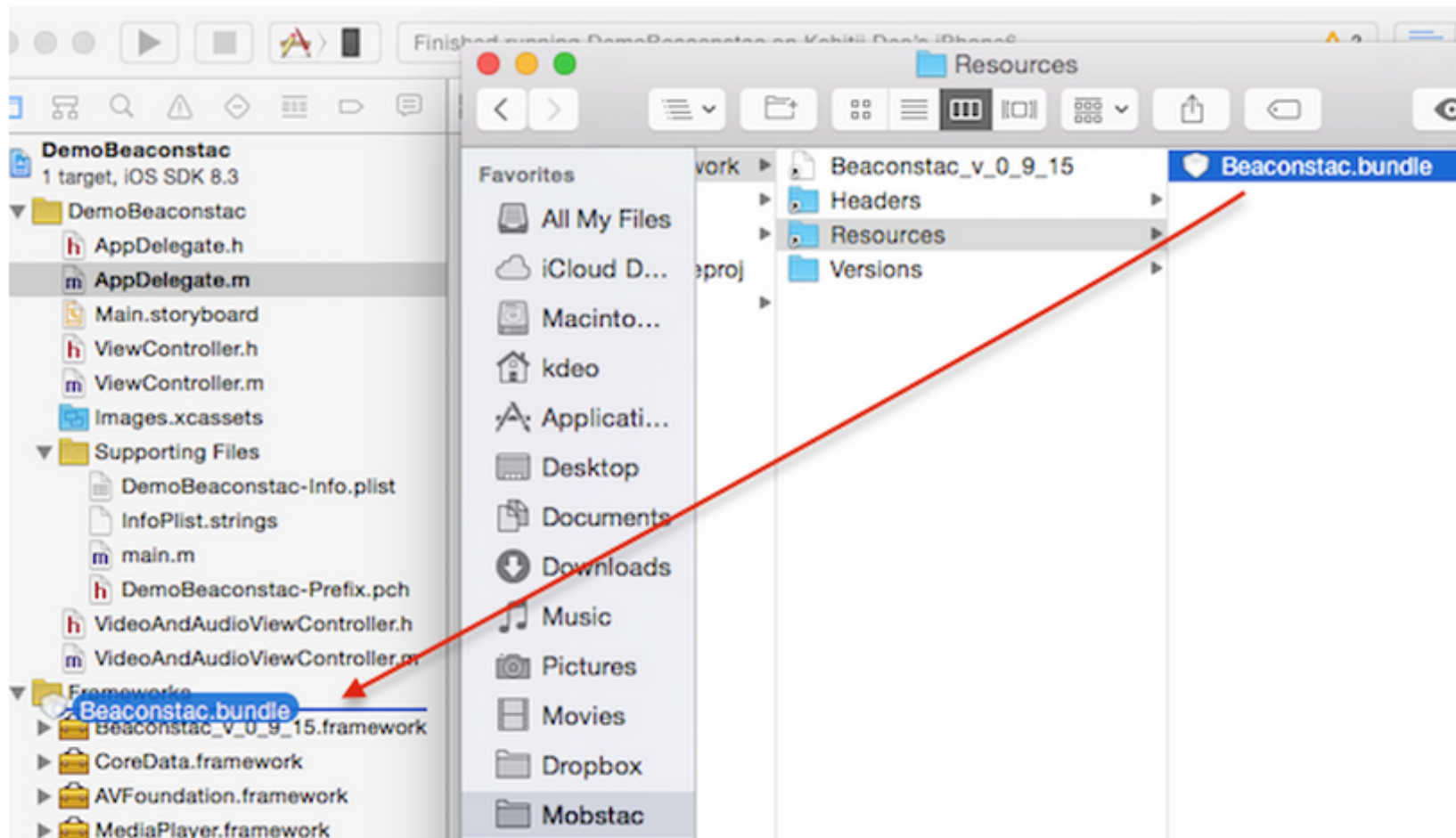
- A machine running with Xcode 6 installed on it
- An iOS device with BLE hardware and running iOS 8.0 or above
- An active Beaconstac developer account
- Beaconstac starter kit (alternatively Beaconstac beacons which are added to your Beaconstac account)

## Steps:

1. Create a new XCode project/open your existing project. Choose the language as Objective-C.
2. Download the Beaconstac iOS SDK from <https://github.com/Beaconstac/iOS-SDK>. Drag and drop the Beaconstac.framework file into your Xcode project. Make sure that "Copy Items to Destination's Group Folder" is checked.



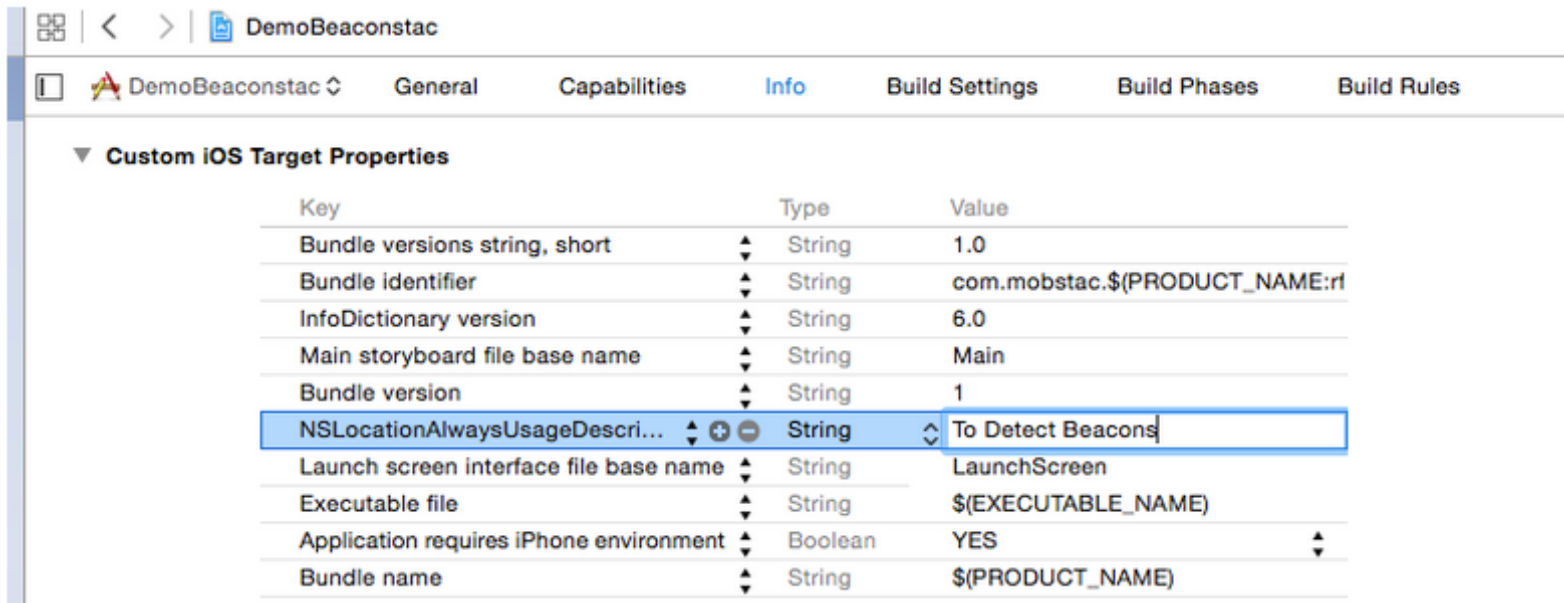
3. Navigate to Beaconstac.framework/Resources folder in Finder. Drag and drop the Beaconstac.bundle into Project navigator area. Make sure that "Copy Items to Destination's Group Folder" checked.



4. In Build Phases under Target, add the following frameworks in “Link Binary With Libraries” section:

1. CoreData.framework
2. SystemConfiguration.framework
3. CoreBluetooth.framework
4. CoreLocation.framework

5. In Info.plist, add a new field, NSLocationAlwaysUsageDescription with relevant string that you want to show to the user. This is mandatory for iOS 8 and above. Under Targets, choose the Deployment Target as 8.0 or above.



6. Import the Beaconstac framework header in this ViewController.h file and make sure the class conforms to BeaconstacDelegate protocol

```
#import <Beaconstac/Beaconstac.h>

@interface ViewController () <BeaconstacDelegate>
```

7. In the ViewDidLoad method of the class, initialize Beaconstac using SDK method:

```
Beaconstac *beaconstacInstance = [Beaconstac sharedInstanceWithOrganizationId: <orgId> developerToken: <devToken>];
beaconstacInstance.delegate = self;
```

You can find the organization id and developer token from Admin Console.

## My Beaconstac Account

Developer token e62435a78e67ec98bba3b879ba00448650032557

Organization ID 720

For basic access authentication to the API, please construct the Authorization header as follows:

```
Authorization: Token e62435a78e67ec98bba3b879ba00448650032557
```

### 8. To start ranging beacons:

```
[beaconstacInstance startRangingBeaconsWithUUIDString:<Enter valid 16 byte UUID string> beaconId
```

### 9. Implement delegate methods to receive callbacks when beacons are ranged:

```
- (void)beaconstac:(Beaconstac *)beaconstac rangedBeacons:(NSDictionary *)beaconsDictionary
{
    NSLog(@"Beacons around %@",beaconsDictionary);
}
```

If you run the program on your iOS device and beacons with the above mentioned UUID are near the device, you should see logs in the XCode console:

1. Similarly, you can implement delegate methods to receive callbacks when device camps on a beacon.

```
- (void)beaconstac:(Beaconstac *)beaconstac campedOnBeacon:(MSBeacon *)beacon amongstAvailableBeacons
{
    NSLog(@"Camped on beacon %@", beacon);
}
```

If you bring a beacon closer than 2 meter to the device, console log will show:

1. To show Rule trigger in action, we need to create a new rule on the Admin Console. Go to Rules in the left sidebar and click “Add a new Rule”. Enter the Rule name and select the Beacon you want to attach the rule to.



## Compose rule

A "rule" allows you to define a set of conditions that have to be fulfilled for an action to be performed.

### RULE NAME

☐ Single Beacon☐ Tags

### BEACON

[Add a new beacon](#) **TRIGGER ON****AFTER** 

seconds

[SET FILTERS WITH CUSTOM ATTRIBUTES](#)

Scroll down the page and select the action type as Text Alert and enter a text message:

## Set Actions

An action is what you want Beaconstac to do when the conditions of a rule are met. Think of it as 'if **conditions** then **action**'

**ACTION NAME**

Show text alert

**ACTION TYPE**

Text alert

Hi! welcome to Beaconstac.

You have **174** characters left

Add another action

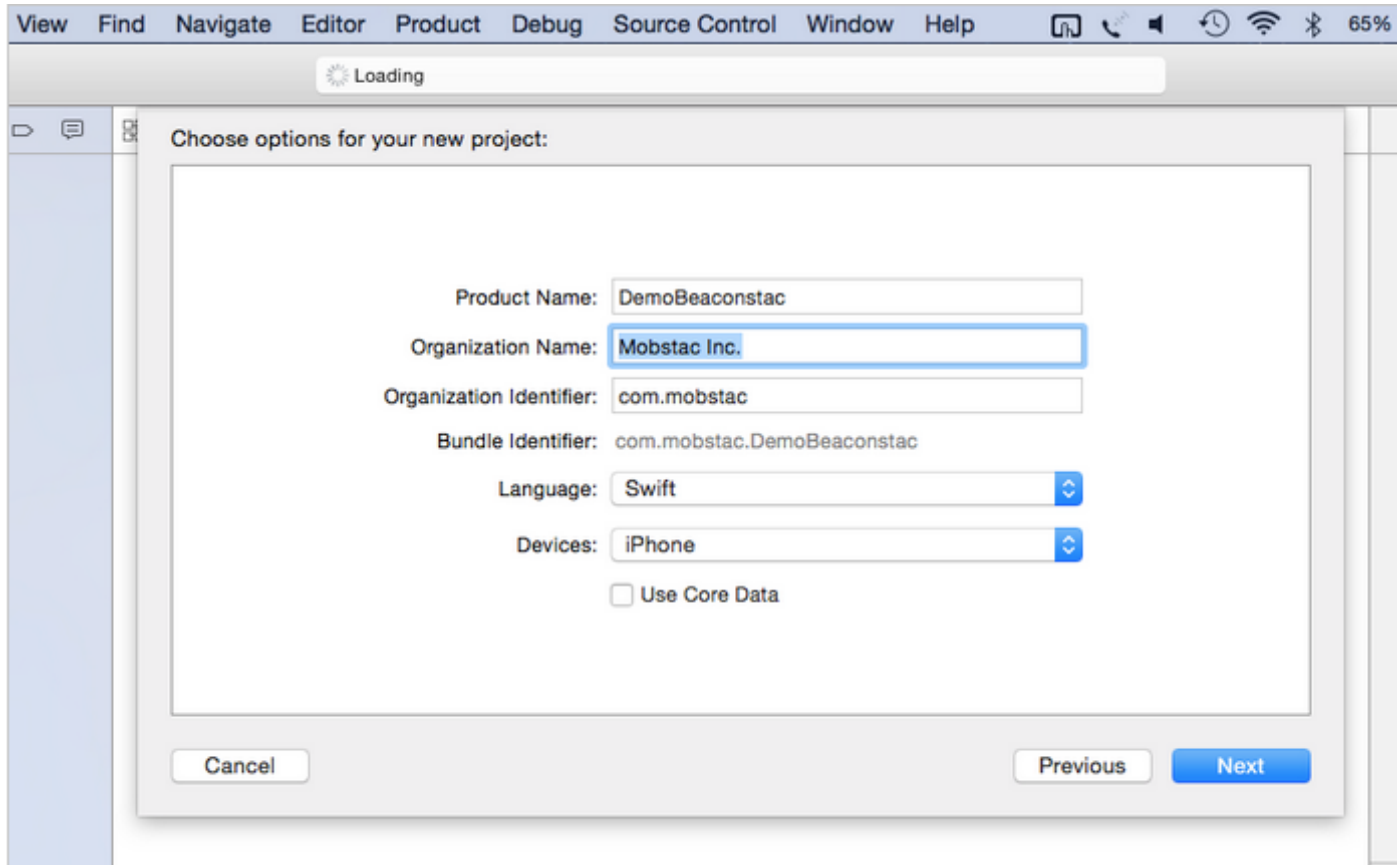
2. Now, in the XCode project, implement delegate method for Rule Trigger. The SDK will send a callback when the device camps on to this beacon and the rule gets triggered.

```
- (void)beaconstac:(Beaconstac *)beaconstac triggeredRuleWithRuleName:(NSString *)ruleName actionA
{
    NSLog(@"Triggered Rule %@",ruleName);
    MSAction *action = [actionArray firstObject];
    if (action.type == MSActionTypePopup) {
        [[[UIAlertView alloc] initWithTitle:action.name message:action.message delegate:nil
        }
    }
}
```

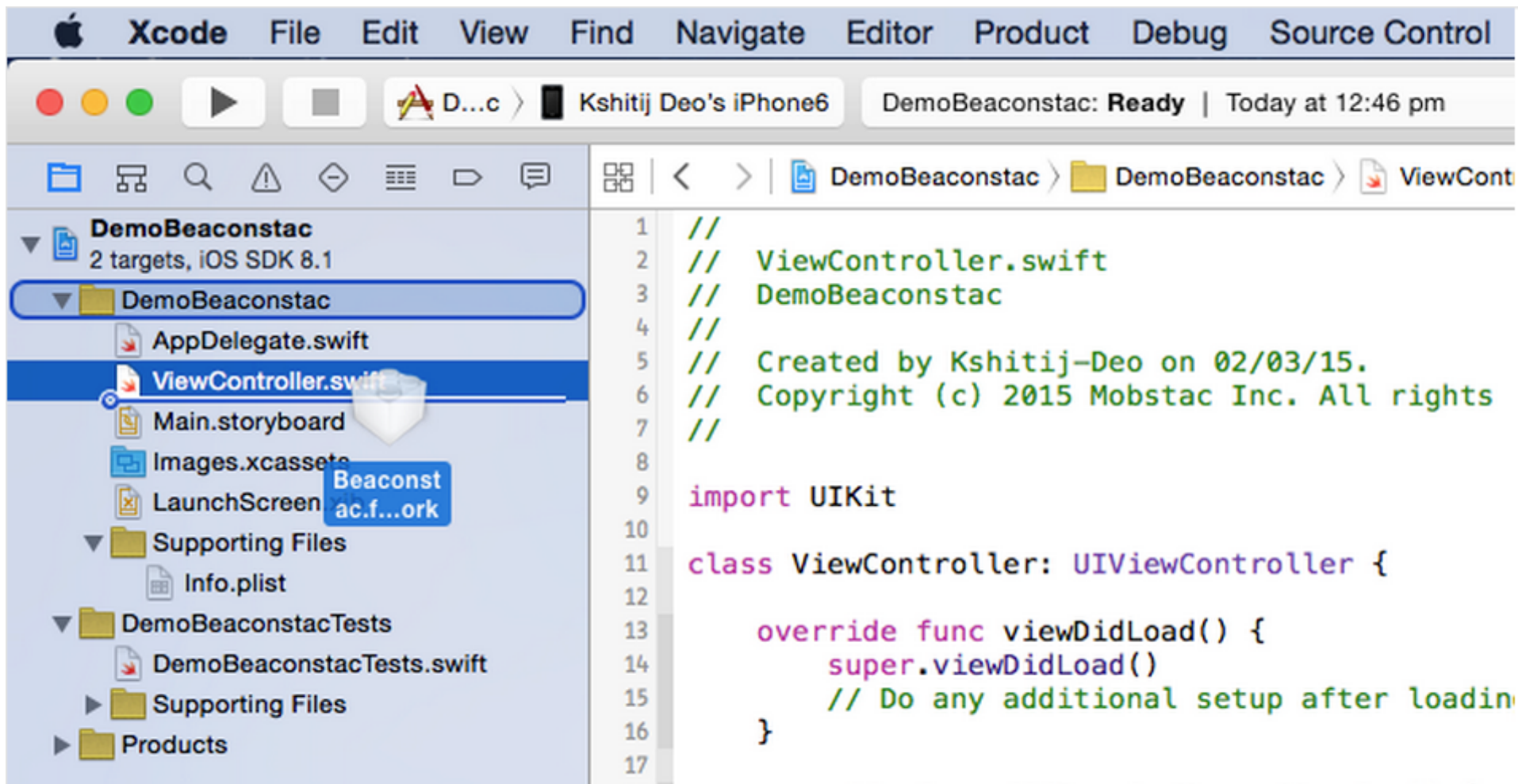
The app on phone should show a popup with the message you entered above.

#### #### Integration in Swift project

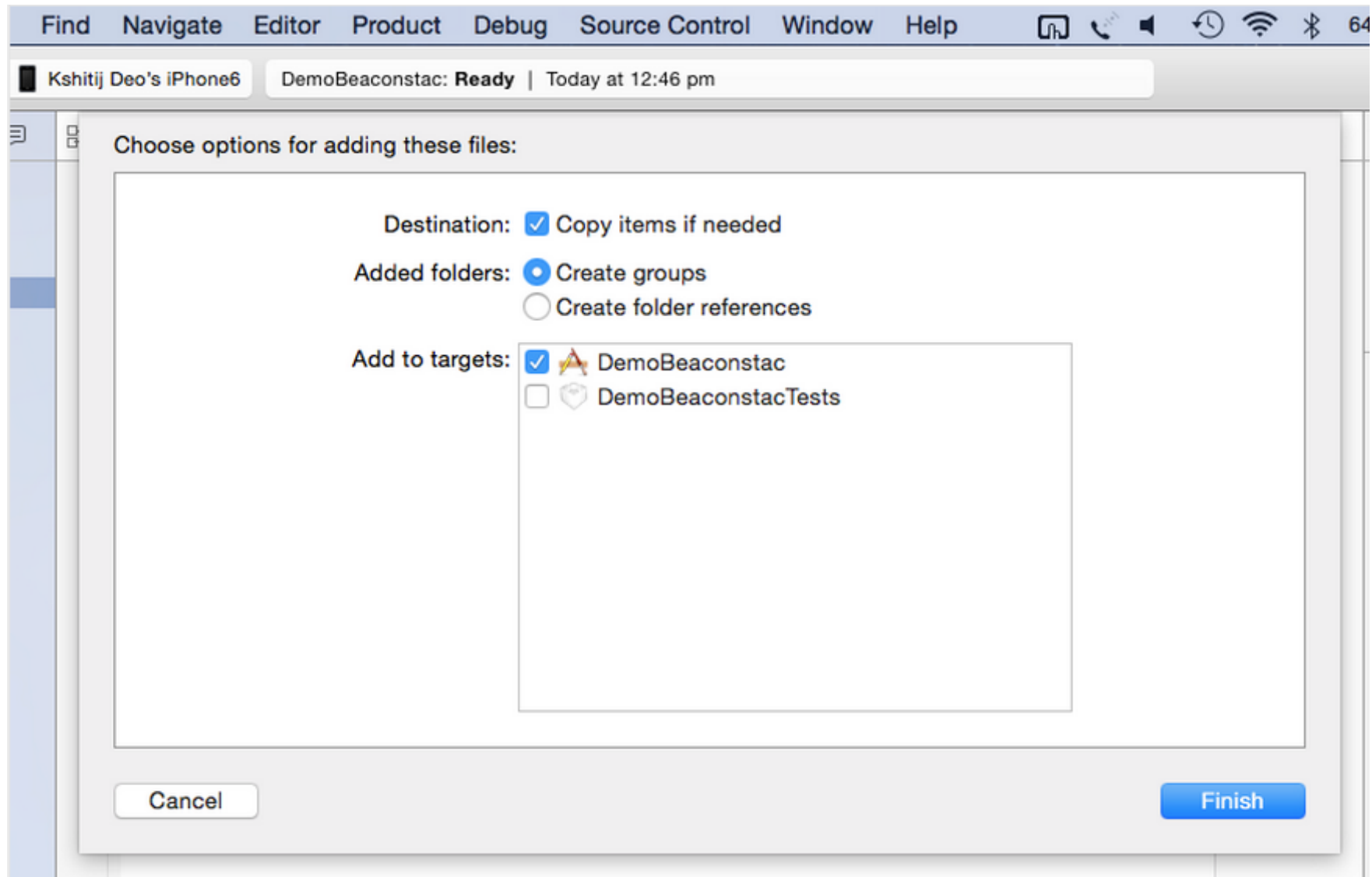
1. Create a New Swift project/open an existing one in XCode.



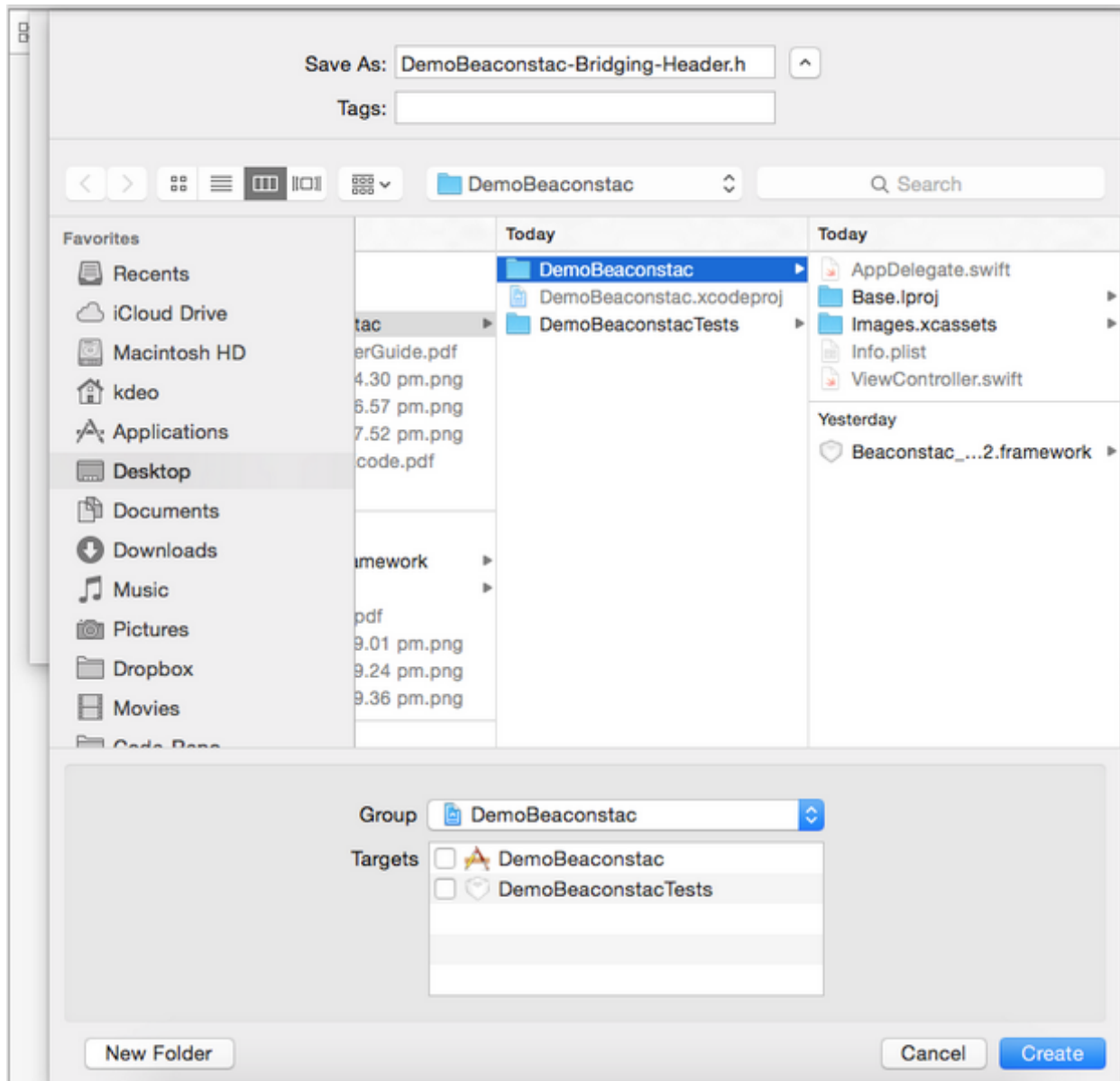
2. Drag and drop Beaconstac.framework in Project navigator



3. Select "Copy if needed"



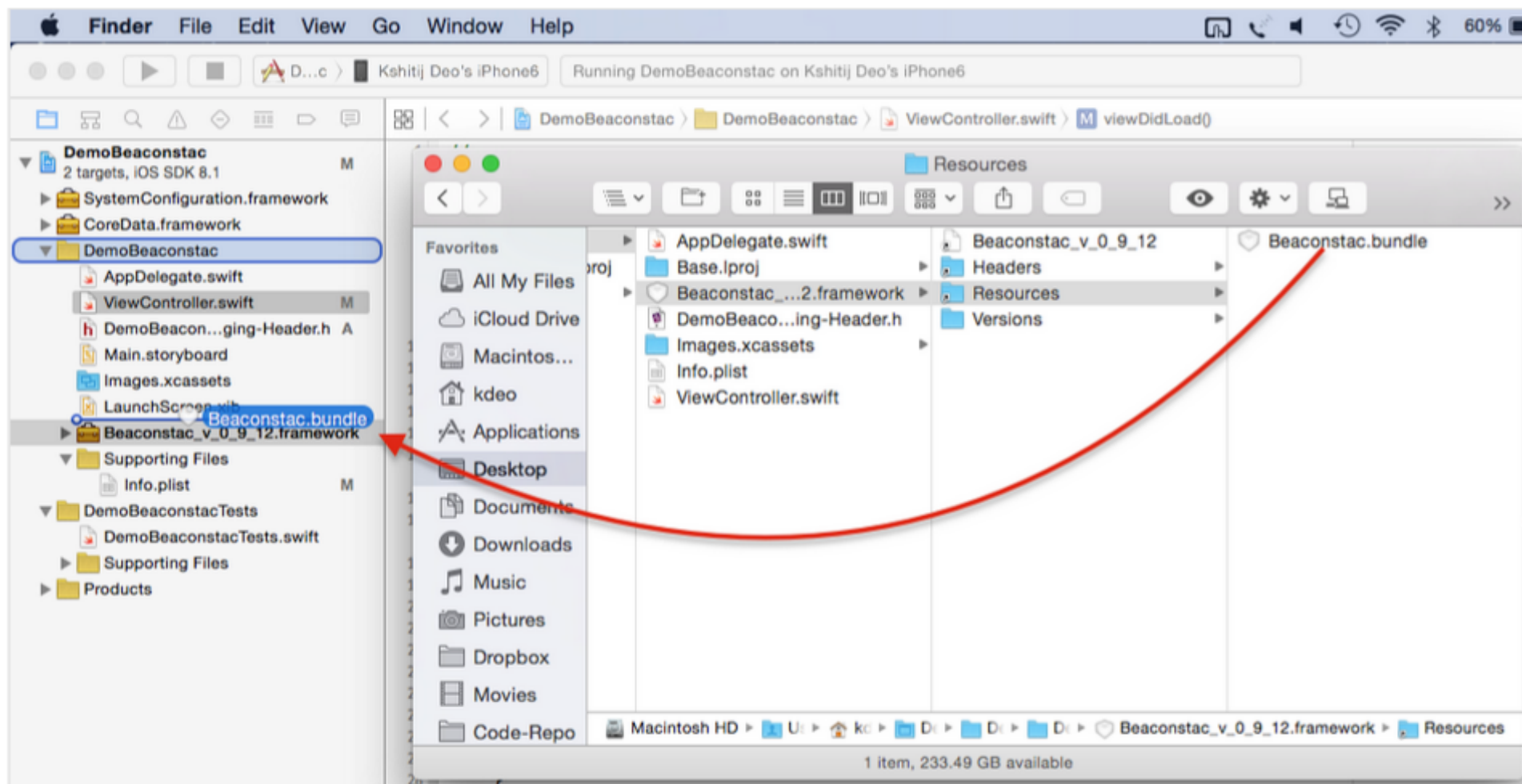
4. Create a new Header file and name it ProjectName-Bridging-Header.h



5. In the Bridging header file, insert import statement for Beaconstac framework

```
#import <Beaconstac/Beaconstac.h>
```






6. Navigate to Beaconstac.framework/Resources folder in Finder and drag and drop Beaconstac.bundle into Project navigator area



7. In Build Phases under Target, add the following frameworks in “Link Binary With Libraries” section:

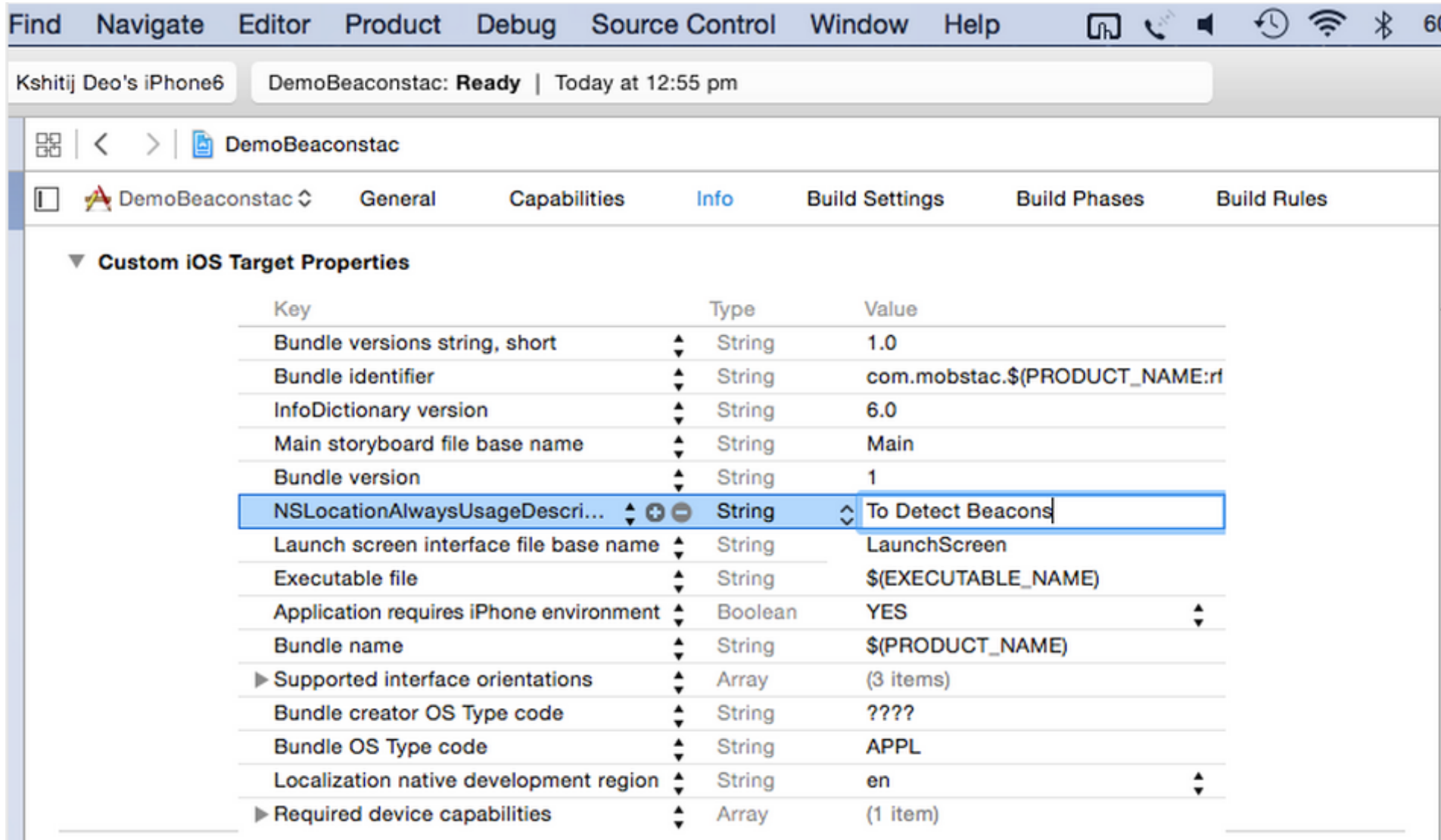
1. CoreData.framework
2. SystemConfiguration.framework
3. CoreBluetooth.framework
4. CoreLocation.framework

▼ Link Binary With Libraries (5 items) ×

Name	Status
 CoreLocation.framework	Required ⇅
 CoreBluetooth.framework	Required ⇅
 SystemConfiguration.framework	Required ⇅
 CoreData.framework	Required ⇅
 Beaconstac.framework	Required ⇅
+ — Drag to reorder frameworks	

8. In Info.plist, add a new field, `NSLocationAlwaysUsageDescription` and add an appropriate string value which you want to show to the user



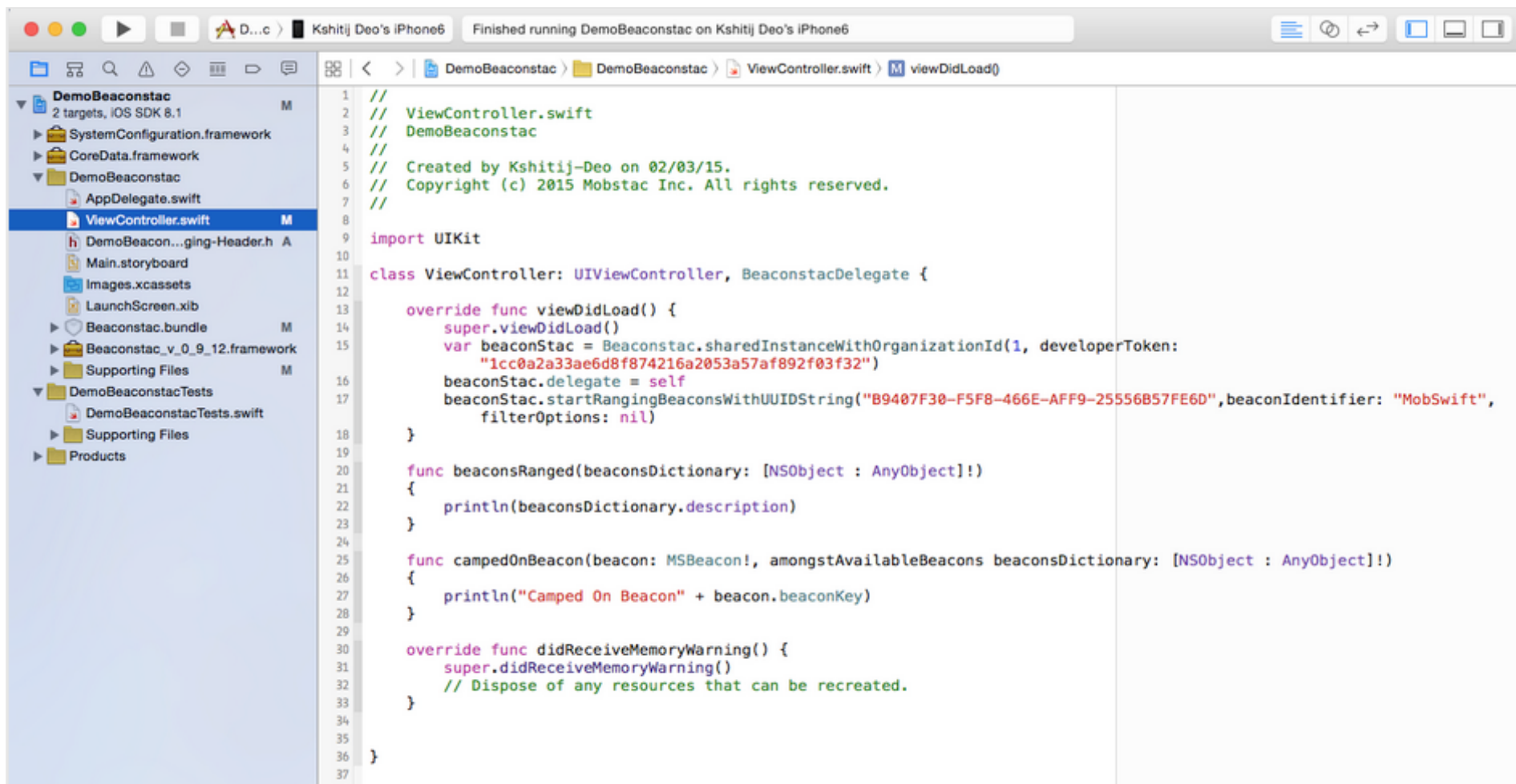


9. In your ViewController.Swift file, make sure it conforms to BeaconstacDelegate protocol

10. Initialize Beaconstac using the SDK method:

```
beaconstac.sharedInstanceWithOrganizationId(<organizationId: Int>, developerToken: <String!>)
```

11. Write the startRanging functions and implement corresponding delegate methods. Run the project on iOS device and you should see ranging callbacks in debug console.



```

1 import UIKit
2
3 class ViewController: UIViewController, BeaconstacDelegate
4 {
5     override func viewDidLoad()
6     {
7         super.viewDidLoad()
8         var beaconstac = Beaconstac.sharedInstanceWithOrganizationId(1xx, developerToken: "1ccxxxxxxxxxxxxxxxxxxxxxxxxxxxx")

```

```
9      MSLogger.sharedInstance().logLevel = MSLogLevel.Verbose
10
11      beaconStac.delegate = self
12      beaconStac.startRangingBeaconsWithUUIDString("F94DBB23-2266-7822-3782-57BEAC0952AC", beaconIdentifier: "MobstacRegion")
13  }
14
15  func beaconstac(beaconstac: Beaconstac!, rangedBeacons beaconsDictionary: [NSObject : AnyObject]!)
16  {
17      print("Beacons ranged" + beaconsDictionary.description)
18  }
19
20  func beaconstac(beaconstac: Beaconstac!, campedOnBeacon beacon: MSBeacon!, amongstAvailableBeacons beaconsDictionary: [NSObject : AnyObject]!)
21  {
22      print("Camped On Beacon" + beacon.beaconKey)
23  }
24
25  override func didReceiveMemoryWarning()
26  {
27      super.didReceiveMemoryWarning()
28  }
29 }
```

Beaconstac\_Viewcontroller hosted with [by GitHub](#)

[view raw](#)

Please follow Step 10 to Step 12 in Objective-C section on how to create your own rule and implementing callbacks for Beacon CampOn and Rule Trigger.



Powered by Beaconstac