iBeacons

Boulder iOS Meetup Jan 19, 2016

john.dumais@comcast.net

What Are They?

- Bluetooth LE Devices
- Adhering to a specific protocol
    - You can think of as an advertising packet
        - That includes
            - UUID
            - 2 integers for identification
- Including an indication of relative proximity by measuring received signal strength — RSSI

What Are They?

- 1-way transmitter
  - Requiring a specific app to know what to do when encountering an iBeacon

What are They

- They come in many varieties
  - Inexpensive (~$5) tiles you can stick onto your keys
    - Which is pretty much the only way I keep track of mine
    - Not a lot of configurability
  - More expensive (~$35)
    - Lots of configurability, many that include settable transmit power

What Are They Used For?

- Advertising and mobile marketing
- Mobile point of sales
- Indoor positioning
- Measuring distance
    - Placing iBeacons at known distances from each other allow you to know how far you've traveled by counting the number of times an app detects a change in the nearest iBeacon it has "ranged"

Nicely Supported in iOS

- Using CoreBluetooth and CoreLocation
- An iOS device can be a transmitter or a receiver

A Receiving App

Your Info.plist needs…

<key>NSLocationAlwaysUsageDescription</key>
<string></string>

You can put message text in the string markup that gets displayed in addition to the system-supplied text in the alert view.

A Receiving App

- CLBeaconRegion
    - An object that allows you to receive events when you are entering or exiting an iBeacon's proximity
        - Where the iBeacon is identified by
            - A UUID
            - And optionally by a major or combination of major and minor id integer values
- Tagged with a string you can use to keep track of more than 1 region in your delegate methods
- Provides a way to fire delegate methods when "ranging" iBeacons meeting characteristics you specified
    - In conjunction with a CLLocationManager instance

A Receiving App

```
import Foundation
import CoreLocation

class BeaconUtilities
{
    static let beaconUuid = NSUUID(UUIDString: "1390B79D-30F1-48D1-8C32-AA5DBA7BC178")
    static let beaconMajorId : CLBeaconMajorValue = 62
    static let beaconMinorId : CLBeaconMinorValue = 93
    private static var beaconRegion : CLBeaconRegion?

    class func getBeaconRegion() -> CLBeaconRegion
    {
        if beaconRegion == nil
        {
            beaconRegion = CLBeaconRegion(proximityUUID: beaconUuid!, major: beaconMajorId,
minor: beaconMinorId, identifier: "")
            beaconRegion?.notifyEntryStateOnDisplay = false
            beaconRegion?.notifyOnEntry = true
            beaconRegion?.notifyOnExit = true
        }

        return beaconRegion!
    }
}
```

A Receiving App

- Initiating discovery
  - Uses a CLLocationManager instance
    - To call startMonitoringForRegion()

A Receiving App

```
import Foundation
import CoreLocation

class LocationUtilities
{
    class func getLocationManager(withDelegate delegate : CLLocationManagerDelegate) -> CLLocationManager
    {
        let locationManager = CLLocationManager()

        locationManager.delegate = delegate

        if locationManager.respondsToSelector("requestAlwaysAuthorization") == true
        {
            locationManager.requestAlwaysAuthorization()
        }

        return locationManager
    }
}
```

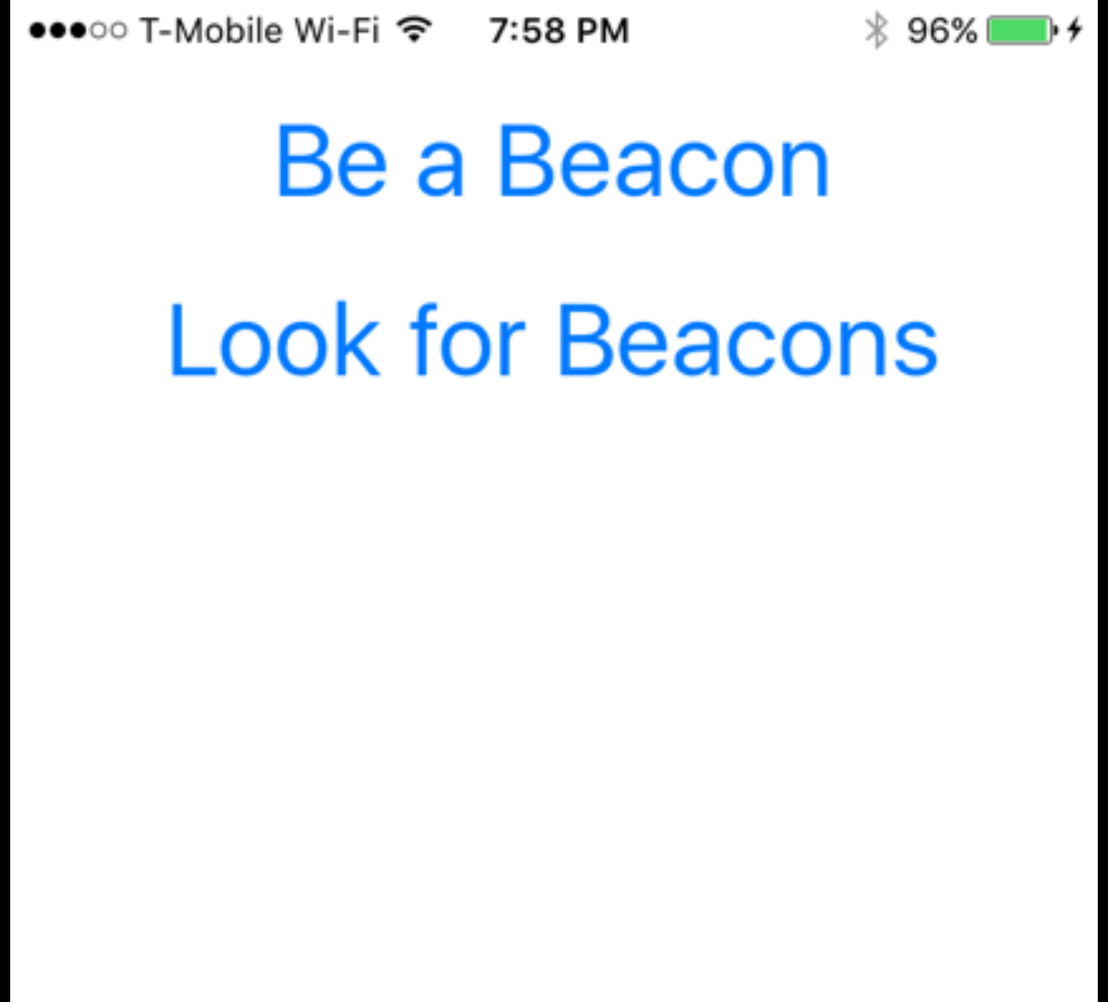
A Receiving App

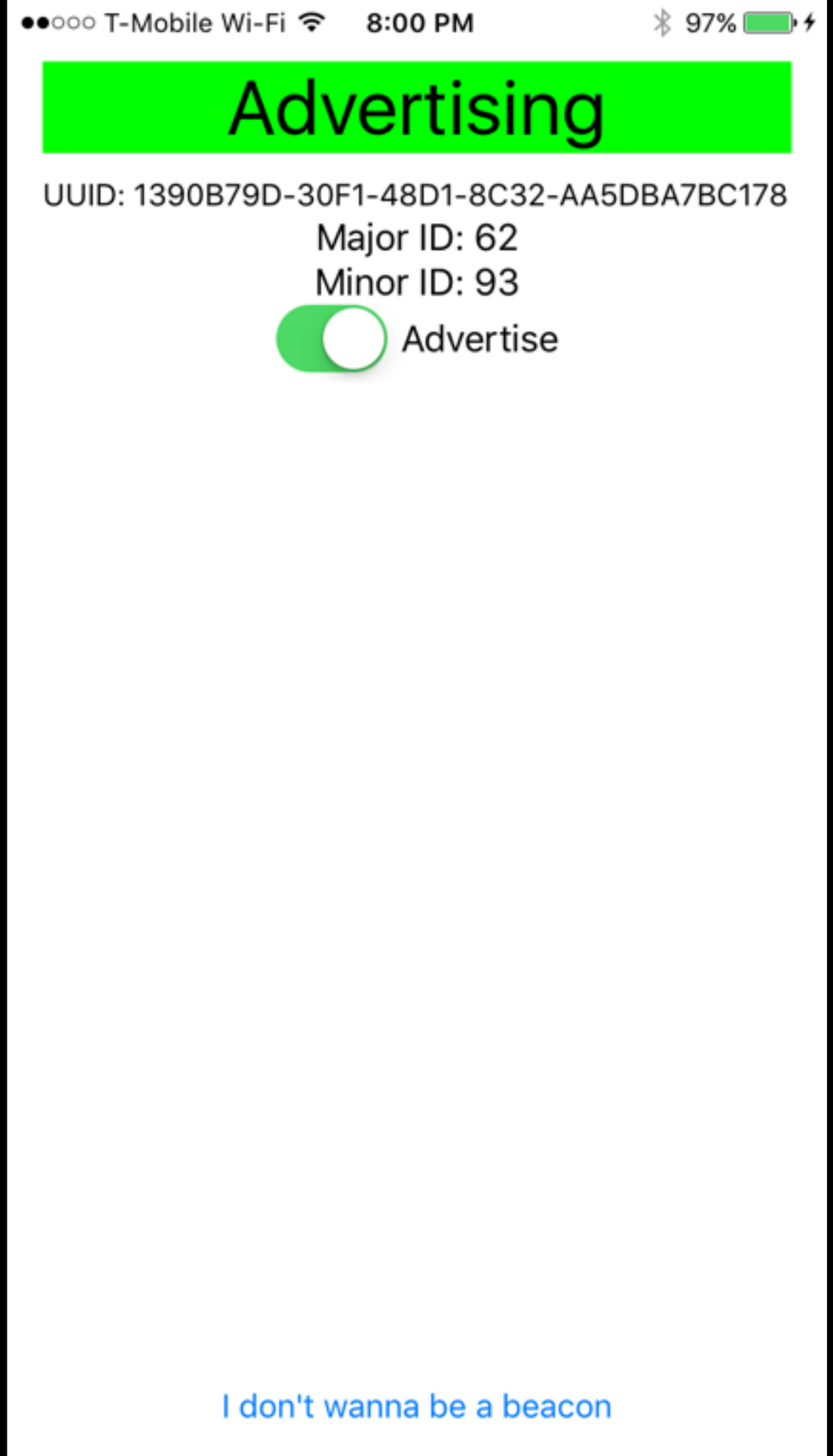
```
@IBAction func lookForBeaconButtonTapped(sender: AnyObject)
{
    lookForBeaconsButton.enabled = false
    locationManager!.startMonitoringForRegion(BeaconUtilities.getBeaconRegion())
}
```

```
extension BeaconFinderViewController : CLLocationManagerDelegate
{
    func locationManager(manager: CLLocationManager, didStartMonitoringForRegion region: CLRegion)
    {
        print("Started monitoring for beacon region: \(region)")

        locationManager!.startRangingBeaconsInRegion(region as! CLBeaconRegion)
    }
    …
}
```

A Receiving App

```
extension BeaconFinderViewController : CLLocationManagerDelegate
{
    …

    func locationManager(manager: CLLocationManager, didRangeBeacons beacons: [CLBeacon], inRegion region: CLBeaconRegion)
    {
        if beacons.count > 0
        {
            let beacon = beacons.last

            switch beacon!.proximity
            {
            case CLProximity.Immediate:
                beaconStatusLabel.text = String(format: "Immediate, %ld", beacon!.rssi)
                beaconStatusLabel.backgroundColor = UIColor.greenColor()

            case CLProximity.Near:
                beaconStatusLabel.text = String(format: "Near, %ld", beacon!.rssi)
                beaconStatusLabel.backgroundColor = UIColor.yellowColor()

            case CLProximity.Far:
                beaconStatusLabel.text = String(format: "Far, %ld", beacon!.rssi)
                beaconStatusLabel.backgroundColor = UIColor.redColor()

            default:
                beaconStatusLabel.text = "Cannot determine proximity"
                beaconStatusLabel.backgroundColor = UIColor.groupTableViewBackgroundColor()
            }
        }
        else
        {
            beaconStatusLabel.text = "Not ranging"
            beaconStatusLabel.backgroundColor = UIColor.groupTableViewBackgroundColor()
        }
    }
}
```


●●●○○ T-Mobile Wi-Fi
7:58 PM
96%
Be a Beacon
Look for Beacons


●●○○○ T-Mobile Wi-Fi
8:00 PM
97%
Advertising
UUID: 1390B79D-30F1-48D1-8C32-AA5DBA7BC178
Major ID: 62
Minor ID: 93
Advertise
I don't wanna be a beacon


●●○○○ T-Mobile Wi-Fi
8:01 PM
97%
Start looking
Immediate, -36
I'm done Looking

Wrap Up

- Get the sample app and the Keynote presentation from…

- https://github.com/GraciesPadre/iOS-IBeacon-Meetup.git