iOS 中 iBeacon 开发

什么是iBeacon?

`iBeacon` 是苹果公司2013年9月发布的移动设备用OS（iOS7）上配备的新功能。其工作方式是，配备有低功耗蓝牙（BLE）通信功能的设备使用`BLE`技术向周围发送自己特有的 ID，接收到该 ID 的应用软件会根据该 ID 采取一些行动。

从个人的角度看： `iBeacon`向四面八方不停地广播信号，就像是往平静的水面上扔了一块石子，泛起层层涟漪(俗称水波)，波峰相当于 iBeacon 的`RSSI`(接受信号强度指示)，越靠近中心点的地方波峰越高(RSSI 越大)，这个波峰的大小(RSSI 的值)受到扔石子时用力大小(发射功率)和水质(周围环境因子)的影响，离中心点越远水波越趋向于平静，超过了一定值，水波会消失于无形，也就是说 iBeacon 向外广播的距离是有范围的，超过了这个范围，将接受不到 iBeacon 的信号。

从iOS开发者的角度看： iBeacon 在 `CoreLocation` 框架中抽象为`CLBeacon`类, 该类有6个属性，分别是:

- `proximityUUID`，是一个 `NSUUID`，用来标识公司。每个公司、组织使用的 iBeacon 应该拥有同样的`proximityUUID`。
- `major`，主要值，用来识别一组相关联的 beacon，例如在连锁超市的场景中，每个分店的 beacon 应该拥有同样的 `major`。
- `minor`，次要值，则用来区分某个特定的 beacon。
- `proximity`，远近范围的，一个枚举值。

```
typedef NS_ENUM(NSInteger, CLProximity) {
        CLProximityUnknown,// 无效
        CLProximityImmediate,//在几厘米内
        CLProximityNear,//在几米内
        CLProximityFar//超过 10 米以外，不过在测试中超不过10米就是far
}
```

- `accuracy`，与iBeacon的距离。
- `rssi`，信号轻度为负值，越接近0信号越强，等于0时无法获取信号强度。

Tip： `proximityUUID`，`major`，`minor` 这三个属性组成 `iBeacon` 的唯一标识符。

只要进入`iBeacon`的范围，就能唤醒 App（大约10秒钟），即使在程序被杀掉的情况下。必要时，可以使用`UIApplication`类的`-(UIBackgroundTaskIdentifier)beginBackgroundTaskWithExpirationHandler:(void (^)(void))handler;`方法，请求更多的后台执行时间。

iBeacon的用途：我们可以用`iBeacon`可以进行室内定位（车库，商场），智能打卡，提醒（离开某物体的时候，比如离开家）。

iBeacon 与 BLE 的区别

iOS 中 iBeacon 是基于地理位置的微定位技术，虽然借助手机蓝牙进行接收`Majro`、`Minor`，但是他们在开发工程中没有任何关系。

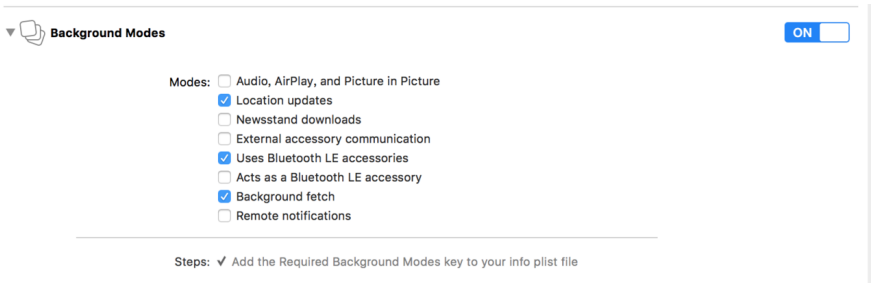
`iBeacon`使用苹果提供`CoreLocation`库，然而在 BLE 在开发过程中使用`CoreBluetooth`库。从上面提供的库来看就很清楚了，特别是在 iOS8.0 之后的时候如果想使用`iBeacon`，必须让用户点击是否允许`XXapp`使用地理位置。如果在第一次使用 iOS App 扫描`iBeacon`的时候没有提示这句话，是不可能接收到`iBeacon`的信号（除非iOS 8.0之下）。如果是 BLE 则的开发过程中之需要提示用户打开蓝牙，并不要求其他的地理位置任何信息。

iBeacon 在 iOS 中的运用

权限请求

在`info.plist`中添加`NSLocationAlwaysAndWhenInUseUsageDescription`,`NSLocationWhenInUseUsageDescription`,`NSLocationAlwaysUsageDescription`，请求地理位置权限。

开启`Background Modes`



相关代码

import `<CoreLocation/CoreLocation.h>`。

初始化`locationManager`和`beaconRegion`。

```
- (CLLocationManager *)locationManager {
    if (!_locationManager) {
        _locationManager = [[CLLocationManager alloc] init];
        _locationManager.delegate = self;
    }
    return _locationManager;
}

- (CLBeaconRegion *)beaconRegion {
    if (!_beaconRegion) {
        _beaconRegion = [[CLBeaconRegion alloc] initWithProximityUUID:[[NSUUID alloc] initWithUUI
```

```
        _beaconRegion.notifyEntryStateOnDisplay = YES;
    }
    return _beaconRegion;
}
```

`CLBeaconRegion`类，提供了3个初始化方法：

```
//监听该UUID下的所有Beacon设备
- (instancetype)initWithProximityUUID:(NSUUID *)proximityUUID identifier:(NSString *)identifier;

//监听该UUID，major下的所有Beacon设备
- (instancetype)initWithProximityUUID:(NSUUID *)proximityUUID major:(CLBeaconMajorValue)major ide

//监听唯一的Beacon设备
- (instancetype)initWithProximityUUID:(NSUUID *)proximityUUID major:(CLBeaconMajorValue)major mi
```

在开始监控之前，我们需要使用`isMonitoringAvailableForClass`判断设备是否支持，是否允许访问地理位置。

```
BOOL availableMonitor = [CLLocationManager isMonitoringAvailableForClass:[CLBeaconRegion class]];

if (availableMonitor) {
    CLAuthorizationStatus authorizationStatus = [CLLocationManager authorizationStatus];
    switch (authorizationStatus) {
        case kCLAuthorizationStatusNotDetermined:
            [self.locationManager requestAlwaysAuthorization];
        break;
        case kCLAuthorizationStatusRestricted:
        case kCLAuthorizationStatusDenied:
            NSLog(@"受限制或者拒绝");
        break;
        case kCLAuthorizationStatusAuthorizedAlways:
        case kCLAuthorizationStatusAuthorizedWhenInUse:{
            [self.locationManager startRangingBeaconsInRegion:self.beaconRegion];
            [self.locationManager startMonitoringForRegion:self.beaconRegion];
        }
        break;
    }
} else {
    NSLog(@"该设备不支持 CLBeaconRegion 区域检测");
}
```

监听方式

可用两种方式检测区域`Monitoring`或`Ranging`方式

Monitoring：可以用来在设备进入/退出某个地理区域时获得通知，使用这种方法可以在应用程序的后台运行时检测iBeacon，但是只能同时检测 20 个 region 区域，并且不能够推测设备与 iBeacon 的距离。

```
// 开始检测区域
[self.locationManager startMonitoringForRegion:beaconRegion];

// 停止检测区域
[self.locationManager stopMonitoringForRegion:beaconRegion];

// Monitoring成功对应回调函数
- (void)locationManager:(CLLocationManager *)manager didStartMonitoringForRegion:(CLRegion *)regi

// 设备进入该区域时的回调
- (void)locationManager:(CLLocationManager *)manager didEnterRegion:(CLRegion *)region;

// 设备退出该区域时的回调
- (void)locationManager:(CLLocationManager *)manager didExitRegion:(CLRegion *)region;
```

```
// Monitoring有错误产生时的回调
- (void)locationManager:(CLLocationManager *)manager monitoringDidFailForRegion:(nullable CLRegic
```

Ranging：可以用来检测某区域内的所有 iBeacons。

```
// 开始检测区域
[self.locationManager startRangingBeaconsInRegion:beaconRegion];

// 停止检测区域
[self.locationManager stopRangingBeaconsInRegion:beaconRegion];

// Ranging成功对应回调函数
- (void)locationManager:(CLLocationManager *)manager didRangeBeacons:(NSArray<CLBeacon *> *)beacc

// Ranging有错误产生时的回调
- (void)locationManager:(CLLocationManager *)manager rangingBeaconsDidFailForRegion:(CLBeaconRegi
```

进程 kill 之后，进入 iBeacon 区域的回调

```
// 当程序被杀掉之后，进入ibeacon区域，或者在程序运行时锁屏/解锁 会回调此函数
- (void)locationManager:(CLLocationManager *)manager
      didDetermineState:(CLRegionState)state forRegion:(CLRegion *)region
```

争取更多的后台时间

必要时，可以使用`UIApplication`类的`-(UIBackgroundTaskIdentifier)beginBackgroundTaskWithExpirationHandler:(void (^)(void))handler;`方法，请求更多的后台执行时间。

用 iPhone 手机模拟 iBeacon

任何支持使用蓝牙低功耗共享数据的 iOS 设备都可以用作 `iBeacon`。

import `<CoreBluetooth/CoreBluetooth.h>`和`<CoreLocation/CoreLocation.h>`

在`terminal`中使用`uuidgen`命令，生成一个 UUID `063FA845-F091-4129-937D-2A189A86D844`。

其实利用`BLE`来模拟 beacon 设备发送信号，很简单。

相关代码

初始化`peripheralManager`

```
self.peripheralManager= [[CBPeripheralManager alloc] initWithDelegate:self queue:nil options:nil]
```

发送信号

```
NSUUID *proximityUUID = [[NSUUID alloc] initWithUUIDString:self.UUIDTextField.text];
//创建beacon区域
CLBeaconRegion *beaconRegion = [[CLBeaconRegion alloc] initWithProximityUUID:proximityUUID major:
NSDictionary *beaconPeripheraData = [beaconRegion peripheralDataWithMeasuredPower:nil];
```

```
if(beaconPeripheraData) {
    [self.peripheralManager startAdvertising:beaconPeripheraData];;//开始广播
}
```

停止广播

```
[self.peripheralManager stopAdvertising];
```

注意点

- 需要访问地理位置权限。
- 设备需要开启蓝牙。
- 利用 iOS 设备模拟 beacon信号，Home 出去之后是不能发送信号的。

Demo

官方 Demo

Blog Demo

参考文章

- Apple 官方文档
- iOS 中 iBeacon 总结
- iBeacon工作原理
- 在ios 的开发中 iBeacon 和 BLE 的区别