

青风带你玩蓝牙 nRF52832 系列教程.....	2
-----作者: 青风.....	2
作者: 青风.....	3
出品论坛: www.qfv8.com	3
淘宝店: http://qfv5.taobao.com	3
QQ 技术群: 346518370.....	3
硬件平台: 青云 QY-nRF52832 开发板.....	3
第 10 章 蓝牙 ibeacon 的应用.....	3
10.1 蓝牙 ibeacon 的基本介绍.....	3
10.2 蓝牙 ibeacon 代码解析.....	4
10.3 蓝牙 ibeacon 的应用.....	10
10.3.1 蓝牙 ibeacon 的微信摇一摇.....	10
10.3.2 蓝牙测距.....	14

作者: 青风**出品论坛: www.qfv8.com****淘宝店: <http://qfv5.taobao.com>****QQ 技术群: 346518370****硬件平台: 青云 QY-nRF52832 开发板**

第 10 章 蓝牙 ibeacon 的应用

10.1 蓝牙 ibeacon 的基本介绍

iBeacon 是苹果在 2013 年 WWDC 上推出一项基于蓝牙 4.x (Bluetooth LE | BLE | Bluetooth Smart) 的精准微定位技术, iBeacon 是基于蓝牙 Bluetooth Low Energy 低功耗蓝牙传输技术一种应用。iBeacon 基站不断向四周发送蓝牙信号, 当智能设备进入设定区域时, 就能够收到信号。

iBeacon 设备只使用了广播通信信道, 因此没有 BLE 的后续连接相关步骤, 所有这种情况使得 ibeacon 的功耗相当的低, 一块纽扣电池就能工作长达 2 年时间, 而且在设备不断对外发射信号情况下。

正如 beacon 英文信标、灯塔的字面意思, 这种设备以一定的时间间隔发送数据包, 并且发送的数据可以被像手机这样的设备获取。同时与信标的远近可以根据信标的信号强度大小来判断, 距离越远, 信标信号约弱。其中“iBeacon”一词专指苹果自家的 Beacon 平台。实际上, 该技术并非苹果公司专有, 市面上还有很多其他的 Beacon 服务及设备。每个公司都会定制自己相应的广播包格式。

根据 Beacon 的这种特性, 其在消息推送、移动支付、室内定位等方面都有广泛的应用。

比如在 Beacon 的系统框架下, 消息推送的应用十分方便, 比如: 用户手机上的特定应用可以发现用户正位于博物馆的某展区附近, 然后将与该展区相关的富信息“拉取”到手机当中, 例如文字介绍、音频、视频、链接、相关文物等信息。比如推送如下的图片:



又或者结合室内导航和消息推送的应用：当你走入一家大型商场的店铺，同时这也意味着你已经进入了这家店铺的 iBeacon 信号区域。然后 iBeacon 基站便可以向你的手机传输各种信息，比如优惠券或者是店内导航信息，甚至当你走到某些柜台前面时，iBeacon 还会提供个性化的商品推荐信息。也就是说用户如果在 iBeacon 基站的信息区域内，用户可以通过手中的智能手机便能够获取个性化的信息推送通知。更进一步，如果你需要购买这个商品，ibeacon 还可以通过 app 向用户推送支付二维码，实现支付功能。

这个应用里，iBeacon 仅提供位置服务，推送消息功能需要您去开发一款 APP 软件，并安装到用户手机中，在后台挂载相应服务程序。例如进入 iBeacons 广播范围的用户手机收到商场打折促销信息，信息是由手机上的应用去获取的，而不是保存在 iBeacons 里。具体实现过程是：当 APP 软件接收到 iBeacon 广播的位置信息后，经过一定计算来获取自己的当前位置，当计算出来的位置符合您设定的特定条件时，APP 向您的广告数据服务器请求对应内容并呈现给用户，到此完成一条消息的推送。也就是说要实现动态的信息推送，至少需要一个 APP 软件和一个数据服务器。当然如果仅向用户呈现固定内容，则数据服务器不是必须的。

同时 iBeacon 可以分清楚不同的距离概念，比如近（near），适中（medium）和远（far），从而使得 iBeacon + 蓝牙基于位置服务中远远好于 GPS + WiFi 组合。在平均 20 平米 1 个 beacon 的部署密度情况下。定位精度保持在分米级别。而且 beacon 硬件和施工成本相当低。在很多场景下与现有室内导航技术相比的技术优势明显。

10.2 蓝牙 ibeacon 代码解析

ibeacon 的核心就是广播，ibeacon 不需要连接，app 所有信息获取的信息都是以广播形式广播出去。

我们关注的就是 ibeacon 广播哪些信息。你需要广播哪些信息？技术要求默认的需要哪些信息？Beacon 编码格式“帧格式”，我们以 Apple 的 iBeacon 定义为基础进行介绍。Apple 的 iBeacon 定义

可以用下表表示:

AD Field Length	Type	Company ID	iBeacon Type	iBeacon Length	UUID iBeacon	Major	Minor	TX Power
-----------------	------	------------	--------------	----------------	-----------------	-------	-------	----------

上面的参数可以解释如下:

AD Field Length 表示 advertisement Data 的长度, 表示有用的广播信息长度。

Type 表示 advertisement type, 也就是广播类型

Company ID 数据字段以两字节的公司 ID 码开始。SIG 将这些 ID 码发放给公司, 并且能知道应用程序如何解析这些字段。上图的 0x004C 就是 apple 公司的 ID。0x0059 是 Nordic Semiconductor 公司的 ID。

iBeacon type 指明了其类型是“proximity beacon”, 即 0x02。

iBeacon Length 长度字段描述了剩下的字段的长度。

iBeacon UUID 表示该设备服务分配的私有任务的 128bit 的基础 UUID。

Major 和 Minor 字段包含了位置信息, Major 字段表示分组号, 通常指示了建筑物。Minor 指示的是该分组内的单元编号, 通常指示建筑物里的位置信息。两个参数可以表示哪个分组里的哪个 ibeacon。

Tx Power 字段表示 app 通过 ibeacon 发送信号强度估算出的在 1m 的时候的 RSSI 强度。

那么对于 ibeacon 的数据, 通过广播形式广播出去, 那么其主要设置就是主函数里广播初始化部分(首先请先阅读之前关于广播初始化的教程)所有定义的参数必须在广播初始化的时候进行配置, SDK 里提供了函数 `ble_advdata_set` 对广播参数进行配置, 代码如下所示:

```
01. static void advertising_init(void)
02. {
03.     uint32_t      err_code;
04.     ble_advdata_t advdata;
05.     uint8_t       flags = BLE_GAP_ADV_FLAG_BR_EDR_NOT_SUPPORTED;//广播类型
06.
07.     ble_advdata_manuf_data_t manuf_specific_data;
08.     manuf_specific_data.company_identifier = APP_COMPANY_IDENTIFIER;//公司 ID 号
09.     .....
10.     .....
11.     manuf_specific_data.data.p_data = (uint8_t *) m_beacon_info;//数据结构体
12.     manuf_specific_data.data.size   = APP_BEACON_INFO_LENGTH;//数据长度
13.
14.     // 初始化广播参数
15.     memset(&m_adv_params, 0, sizeof(m_adv_params));
16.
17.     m_adv_params.properties.type =
18.     BLE_GAP_ADV_TYPE_NONCONNECTABLE_NONSCANNABLE_UNDIRECTED;
19.     m_adv_params.p_peer_addr     = NULL;      // Undirected advertisement.
20.     m_adv_params.filter_policy   = BLE_GAP_ADV_FP_ANY;
21.     m_adv_params.interval        = NON_CONNECTABLE_ADV_INTERVAL;
22.     m_adv_params.duration        = 0;          // Never time out.
23.
24.     err_code = ble_advdata_encode(&advdata, m_adv_data.adv_data.p_data,
25.                                   &m_adv_data.adv_data.len);
```



```

26.     APP_ERROR_CHECK(err_code);
27.
28.     err_code = sd_ble_gap_adv_set_configure(&m_adv_handle, &m_adv_data, &m_adv_params);
29.     APP_ERROR_CHECK(err_code);
30. }

```

下面就对这段广播内容进行解释:

第 5 行: 设置广播类型为 BR/EDR not supported

第 8 行: 设置公司 ID.

第 9-10 行: 设置制造商的信息数据结构和广播数据长度

关于广播公司 ID 和制造商的信息数据在广播数据文件 ble_advdata.h 代码中, 通过一个结构体进行定义, 一个公司的 ID 编号, 一个上制造商的信息:

```

31.     typedef struct
32.     {
33.         uint16_t      company_identifier;          /*公司 ID 号编码*/
34.         uint8_array_t  data;                      /*添加的制造商信息. */
35. } ble_advdata_manuf_data_t;

```

其中制造商的信息数据在通过一个结构体在主函数 main.c 中进行了声明, 如下代码所示:

```

36. static uint8_t m_beacon_info[APP_BEACON_INFO_LENGTH] =                /**<
    Information advertised by the Beacon. */
37. {
38.     APP_DEVICE_TYPE,          // 制造商特定的信息。在这个实现中指定设备类型。
39.     APP_ADV_DATA_LENGTH, // 制造商特定的信息。在这个实现中指定制造商特定数据的长
    度。
40.     APP_BEACON_UUID,         // 128 位的 UUID 的值
41.     APP_MAJOR_VALUE,         // Major 值表示分组位置
42.     APP_MINOR_VALUE,         // Minor 值表示分组里的位置值
43.     APP_MEASURED_RSSI        // Beacon 测试的 TX 功率值
44.
45. };

```

主函数中, 对以上几个参数进行复赋值, 如下所示:

```

46. #define APP_BEACON_INFO_LENGTH      0x17 //广播信息长度
47. #define APP_ADV_DATA_LENGTH         0x15 //广播数据长度
48. #define APP_DEVICE_TYPE             0x02  //设备类型
49. #define APP_MEASURED_RSSI           0xC3   //1m 距离下 rssi 的大小
50. //#define APP_COMPANY_IDENTIFIER    0x0059 //公司 ID 号
51. #define APP_MAJOR_VALUE              0x00, 0x0A // major 的值
52. #define APP_MINOR_VALUE             0x00, 0x07 // minor 的值
53. #define APP_BEACON_UUID             0x01, 0x12, 0x23, 0x34, \
44.                                     0x45, 0x56, 0x67, 0x78, \
55.                                     0x89, 0x9a, 0xab, 0xbc, \
56.                                     0xcd, 0xde, 0xef, 0xf0 //UUID 的值

```

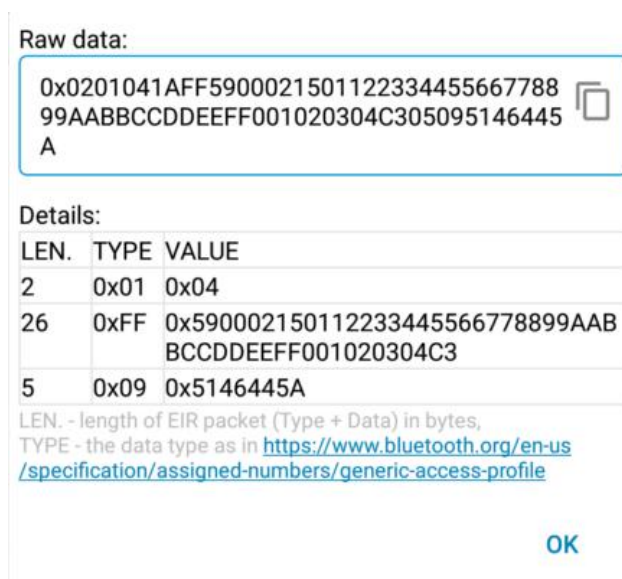
第 15-22 行, 设置广播类型和广播参数, 关于广播参数的设置, 这里就不再累述, 大家参考之前广播初始化的教程

第 24 行: 广播数据编码, 通过 ble_advdata_encode 进行广播数据的编码, 把之前设置的广播数据

放入 ibeacon 广播中。

第 28 行：广播参数配置，通过协议栈底层函数 `sd_ble_gap_adv_set_configure` 配置广播射频模式

程序设置完毕后，下载到 ibeacon 或者开发板中，通过手机 APP nrf connect 可以观察广播包如下图所示，图中可以查看广播包以上的几个字段：



第一行：长度 2，表示后 2 个字节，第一个字节 TYPE 为 0x01，表示广播类型为：设备被发现能力类型。第二个字节 value 的值 0x04 表示 BR/EDR not supported。

第二行：长度 26，表示广播数据长度 26 字节。字节 TYPE 为 0xFF,表示用户定义设备数据，后面的 value 值就是我们前面宏定义已经赋值的数据参数值。

下表是 Type 类型的定义：

AD Data Type	Data Type Value	Description
Flags	0x01	Device discovery capabilities
Manufacturer Specific Data	0xFF	User defined

主函数中，调用初始化广播函数 `advertising_init()`，当函数 `advertising_start()` 启动广播后，就可以实现广播信息的发出，也就是信标的广播包的广播，代码如下所示：

```

57. int main(void)
58. {
59.     uint32_t err_code;
60.     // 初始化
61.     .....
62.     .....
63.
64.     ble_stack_init();//协议栈初始化
65.     advertising_init();//广播初始化
66.
67.     // 输出 BLE Beacon 开始
68.     NRF_LOG_INFO("BLE Beacon started\r\n");
69.     advertising_start();//开始广播
70.

```

```
71. //进入主循环
72. for (;;)
73. {
74.     if(NRF_LOG_PROCESS() == false)
75.     {
76.         power_manage();
77.     }
78. }
79. }
```

如果需要 **ibeacon** 里需要继续添加其他信息, 比如添加广播名称, 电量之类的信息, 如何处理了? 在前面的教程里讲过广播包的大小为 **31** 字节, 这里已经分配给了 **ibeacon** 的固有编码。如果需要在广播里广播更多的信息, 则需要利用广播回包来实现。当主机接收到一个广播包时, 它将发送一个叫做“扫描请求”(Scan Request)的请求来获得更多的广播数据。下面以广播发出广播名称为例。

广播名称的设置在 **GAP** 初始化那节已经讲过, 首先需要在 **gap** 初始化中定义设备名称, 使用 `sd_ble_gap_device_name_set` 函数, 定义设备名称“QFDZ”, 如下所示:

```
01. #define DEVICE_NAME                "QFDZ"
02.
03. static void gap_params_init(void)
04. {
05.     uint32_t          err_code;
06.     ble_gap_conn_sec_mode_t sec_mode;
07.
08.     BLE_GAP_CONN_SEC_MODE_SET_OPEN(&sec_mode);
09.
10.     err_code = sd_ble_gap_device_name_set(&sec_mode,
11.                                           (const uint8_t *)DEVICE_NAME,
12.                                           strlen(DEVICE_NAME));
13.     APP_ERROR_CHECK(err_code);
14. }
15.
```

在广播初始化函数里, 添加 **scanrsp** 扫描回包。**scanrsp** 扫描回包和数据包 **advdata** 数据类型一样, 都是使用宏定义 **ble_advdata_t** 结构体。在扫描回包中, 声明名称类型为 **BLE_ADVDATA_FULL_NAME** 广播全名。同时通过 **ble_advdata_set** 同时配置广播包和扫描回包参数。具体代码如下:

```
01. // Build and set advertising data. 把广播包发送出去
02.     memset(&advdata, 0, sizeof(advdata));
03.
04.     advdata.name_type          = BLE_ADVDATA_NO_NAME;
05.     advdata.flags              = flags;
06.     advdata.p_manuf_specific_data = &manuf_specific_data;
07.
```



```

08.     err_code = ble_advdata_set(&advdata, NULL);
09.     APP_ERROR_CHECK(err_code);
10.
11.     memset(&scanrsp, 0, sizeof(scanrsp));
12.     scanrsp.name_type          = BLE_ADVDATA_FULL_NAME;
13.     err_code = ble_advdata_set(&advdata, &scanrsp);
14.
15. // 初始化广播参数
16.     memset(&m_adv_params, 0, sizeof(m_adv_params));
17.
18.     m_adv_params.type          = BLE_GAP_ADV_TYPE_ADV_SCAN_IND;
19.     m_adv_params.p_peer_addr = NULL; // Undirected advertisement.
20.     m_adv_params.fp           = BLE_GAP_ADV_FP_ANY;
21.     m_adv_params.interval     = NON_CONNECTABLE_ADV_INTERVAL;
22.     m_adv_params.timeout      = APP_CFG_NON_CONN_ADV_TIMEOUT;

```

注意同时设置广播参数类型，需要把：


BLE_GAP_ADV_TYPE_NONCONNECTABLE_NONSCANNABLE_UNDIRECTED

没有连接的非定向广播类型改为

BLE_GAP_ADV_TYPE_NONCONNECTABLE_SCANNABLE_UNDIRECTED 有扫描回包的非定向广播类型。这样才能实现广播回包。

通过手机 APP nrf connect 可以观察 beacon 信息如下图所示，下图显示广播的信息包括了：mac 地址，广播 type 类型，广播 flags 格式，公司 ID 号，应类型 type，ibeacon 信息长度，UUID 号，Major 和 Minor 的值，RSSI 在 1m 左右的大小，广播扫描回包显示的广播名称：

No filter



QFDZ (nRF Beacon)
FA:5A:CD:6F:21:D2
NOT BONDED -81 dBm ↔ 77 ms

CONNECT

Type: BLE only
Flags: BrEdrNotSupported
Beacon data:
Company: Nordic Semiconductor ASA <0x0059>
Type: Beacon <0x02>
Length of data: 21 bytes
UUID: 01122334-4556-6778-899a-abbccddeff0
Major: 258
Minor: 772
RSSI at 1m: -61 dBm
Complete Local Name: QFDZ

CLONE RAW MORE

Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData																CRC	RSSI (dBm)	FCS
		Type	TxAdd	RxAdd	PDU-Length		02 01 04 1A FF 59 00 02 15 01 12 23 34 45 56	67 78 89 9A AB BC CD DE EF F0 01 02 03 04 C3																	
0x8E89BED6	ADV_DISCOVER_IND	6	1	0	36	0xFA5ACD6F21D2																	0xCA96E6	-69	OK

Access Address	Adv PDU Type	Adv PDU Header				ScanA	AdvA	CRC	RSSI (dBm)	FCS
		Type	TxAdd	RxAdd	PDU-Length					
0x8E89BED6	ADV_SCAN_REQ	3	1	1	12	0x72050299E6AA	0xFA5ACD6F21D2	0x0A6CE9	-42	OK

Access Address	Adv PDU Type	Adv PDU Header				AdvA	ScanRspData	CRC	RSSI (dBm)	FCS
		Type	TxAdd	RxAdd	PDU-Length					
0x8E89BED6	ADV_SCAN_RSP	4	1	0	12	0xFA5ACD6F21D2	05 09 51 46 44 5A	0xA94A33	-69	OK

10.3 蓝牙 ibeacon 的应用

10.3.1 蓝牙 ibeacon 的微信摇一摇

现在我们来谈一谈微信摇一摇周边，“摇一摇周边”是什么了？摇一摇周边是微信针对低功耗蓝牙硬件（支持 iBeacon 协议）提供的连接入口。在手机蓝牙打开的状态下，当用户在微信中打开摇一摇室，如果周围有 iBeacon 设备，会自动出现周边入口。摇一摇周边是微信基于低功耗蓝牙技术的 O2O 入口级应用，作为微信在线下的全新功能，为线下商户提供近距离连接用户的能力，并支持线下商户向周边用户提供个性化营销、互动及信息推荐等服务。

2015 年 1 月 28 日，“微信摇一摇周边”开启自助申请入口的测试。测试期间，商户可通过摇周边的商户申请平台进行自助接入。

2015 年 4 月 12 日，在微信公开课第三季长沙站现场，微信团队宣布“微信摇一摇周边”正式对外开放。拥有微信认证的公众帐号商户，均可通过摇周边的商户申请平台或者微信公众平台后台申请入驻。联合微信支付、公众帐号、微信卡包，摇周边为更多商家提供了便捷连接用户和精准近场服务的能力 [1]。

那么“摇一摇周边”的业务特点：

1) 精准定位线下用户，提供个性化服务

利用 iBeacon 设备特有的精准定位能力，而且设备体积小，成本低，易安装；用户摇一摇时，可以根据位置和其他相关信息，提供高度个性化的服务，提升体验；

2) 开放的页面内容，为用户提供更酷更丰富的体验

HTML5 页面采用 URL 模式接入，商家可自定义所有互动形式；支持微信 JS-SDK。基于 iBeacon 的接口将陆续开放；

3) 大微信体系

摇一摇入口拥有日均千万级以上的访问用户；与微信公众平台，微信支付，卡券，微信连 WiFi 等产品无缝打通。

微信摇一摇周边目前平台开放了哪些接口？目前微信硬件开发如下几个接口：

1) 设备管理接口：通过该接口可以申请设备 ID，查询设备列表，配置设备与页面的绑定关系；

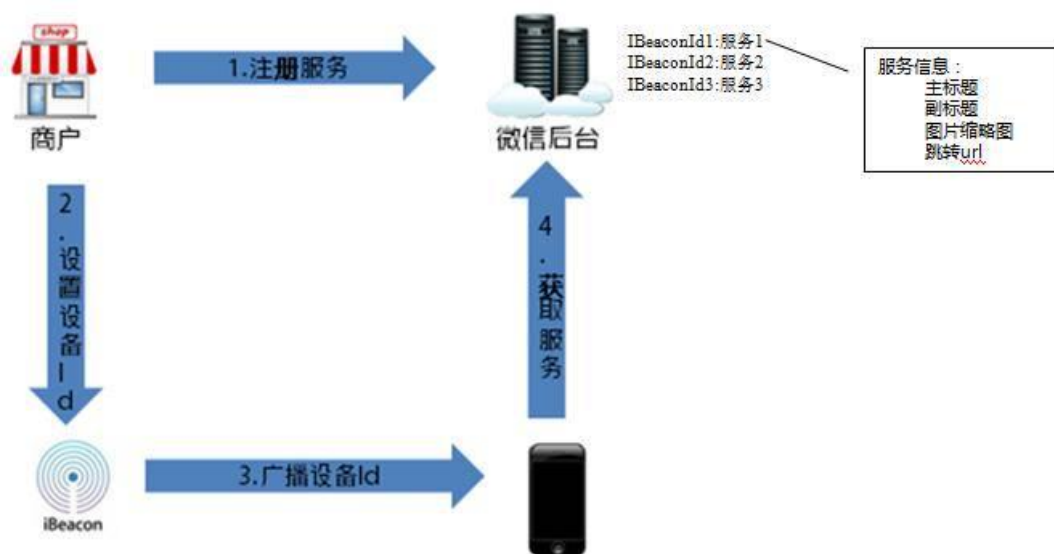
2) 页面管理接口：可以新增，编辑，查询以及删除页面；

3) 信息获取接口：可以获取摇一摇的设备以及用户信息，包括： UMM（UUID、Major、 Minor），设备距离以及用户的 Company ID。

4) 数据统计接口：以设备为纬度，可以统计单个设备周边操作的人数，次数，点击要周边的人数，次数；

5) 一键关注接口: 商户可以在摇出来的页面直接调用摇一摇关注接口, 实现关注公众号的功能。

目前微信摇周边功能申请是无需费用, 由于 iBeacon 协议是开放协议, 商户可以自行通过任意渠道购买支持 iBeacon 设备 (譬如搜索“iBeacon 设备”或者在电商网站购买), 微信无任何限制。如果对 iBeacon 设备完全不了解, 可以参考 iBeacon 设备厂商。对应公众号: 已经通过资质认证的订阅号或服务号即可申请。对应企业号: 已经认证成功的企业号即可申请。下图为商户或企业接入微信摇一摇的基本步骤:



对应申请步骤表述如下: 公众号: 登录 MP 平台->添加功能插件->添加“摇一摇周边”功能插件;
企业号: 登录企业号->服务中心->摇一摇周边

一: 注册: 也可直接在 PC 侧登录“摇一摇周边”的申请入口: <https://zb.weixin.qq.com>;



登录后, 填写基本资料, 提交申请;



审核周期为 3 个工作日;

二: 服务提供者把第一步拿到的 IBeaconId 设置到 IBeacon 设备上, 让 IBeacon 设备广播该 IBeaconId, 下面我们以微信提供的一组测试 ID 为例, 来设置我们的设备, 测试设备 ID 如下图所示:

可将下文的测试ID配置在iBeacon中, 体验“摇一摇·周边”产品流程。

UUID: FDA50693-A4E2-4FB1-AFCF-C6EB076478

25

Major: 10

Minor: 7

01. 代码中, 我们需要修改 UUID 号和 major 和 minor 的值:
02. #define APP_BEACON_INFO_LENGTH 0x17
03. #define APP_ADV_DATA_LENGTH 0x15
04. #define APP_DEVICE_TYPE 0x02
05. #define APP_MEASURED_RSSI 0xC3
06. #define APP_COMPANY_IDENTIFIER 0x004c //改成 apple 公司的 ID
07. #define APP_MAJOR_VALUE 0x00, 0x0A //修改成测试 major
08. #define APP_MINOR_VALUE 0x00, 0x07 // 修改成测试的 minor
09. #define APP_BEACON_UUID 0xFD, 0xA5, 0x06, 0x93, \
10. 0xA4, 0xE2, 0x4F, 0xB1, \
11. 0xAF, 0xCF, 0xC6, 0xEB, \
12. 0x07, 0x64, 0x78, 0x25 //修改成测试 UUID
- 13.

第三步. 设置好参数后, 下载程序。用户在该 IBeacon 设备的信号范围内打开微信摇一摇周边, 微信 App 拿到该 IBeaconId;

第四步. 微信通过第三步拿到的 IBeaconId, 向微信后台拉取相应的服务, 展示在摇出来的结果上。



第五步. 用户点击摇出来结果, 在微信内嵌的浏览器上, 会带上用户信息跳转到服务提供者第一步申请服务时填的 url, 进入应用页面:



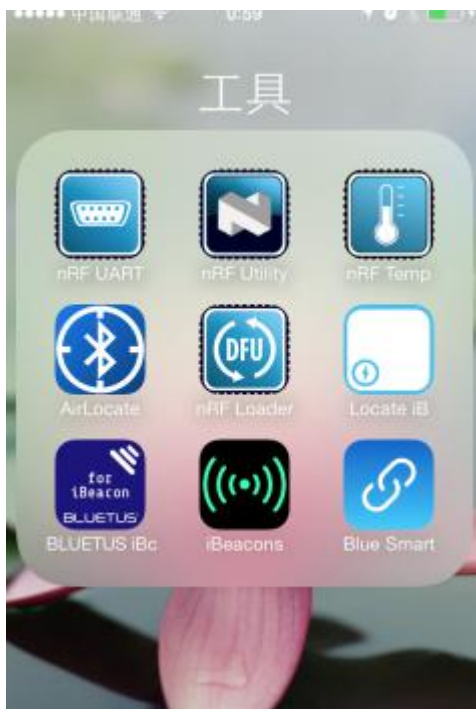
通过上面的设置,可以把 iBeacon 应用在微信摇一摇周边中,摇一摇周边是微信面向线下场景的全新 O2O 入口级应用,可以广泛用于餐饮、广告、展览会议、博物馆景区、商城等领域。商户可根据不同行业场景向周边用户提供如摇红包、摇优惠、摇关注、摇签到、摇投票、摇导航、摇互动、摇支付等个性化营销、互动及信息推荐等服务。因此具有广泛的应用前景。

10.3.2 蓝牙测距

手机上装了很多 App store 上的蓝牙 4.0 BLE 的软件,其中 AirLocate 是苹果公司推出 iBeacon 的测试软件,Locate iBeacons 是一款第三方的软件,能实现 iBeacon 的距离定位和校准,下面我们来看看 Locate iBeacons 实现的距离显示和室内定位。

1: 软件安装:

手机上安装 Locate iBeacons, 你的 iPhone 至少是 4s 或以上, 系统必须是 iOS 7.0 以上, iPhone 4s iOS 7.0 以后的系统才支持蓝牙 4.0 BLE。



打开 keil 工程代码, 修改下面几个参数

修改 UUDI, MAJOR, MINOR, 设置如下:

UUDI: e2c56db5-dffb-48d2-b060-d0f5a71096e0

MAJOR:1

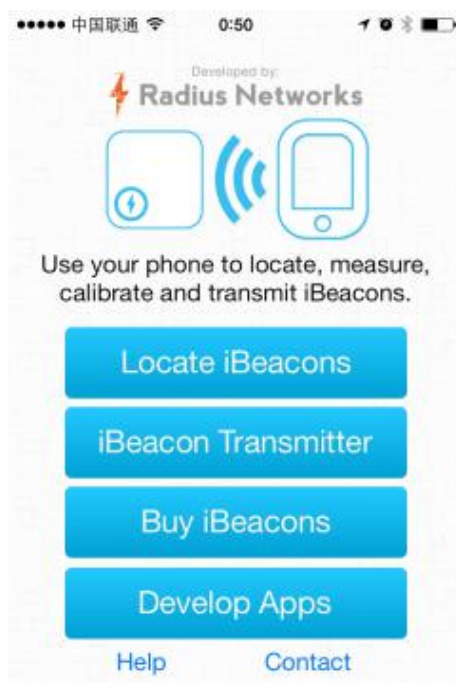
MINOR:2

```

32 #define PERIPHERAL_LINK_COUNT      0          /**< Number of peripheral links
33
34 #define IS_SRVC_CHANGED_CHARACT_PRESENT 0      /**< Include support for
35
36 #define APP_CFG_NON_CONN_ADV_TIMEOUT 0         /**< Time in seconds before
37 #define NON_CONNECTABLE_ADV_INTERVAL MSEC_TO_UNITS(100, UNIT_0_625_MS) /**< The s
38
39 #define APP_BEACON_INFO_LENGTH      0x17       /**< Total length of beacon info
40 #define APP_ADV_DATA_LENGTH         0x15       /**< Length of advertising data
41 #define APP_DEVICE_TYPE              0x02       /**< 0x02
42 #define APP_MEASURED_RSSI            0xC3       /**< The RSSI value
43 #define APP_COMPANY_IDENTIFIER       0x004c     /**< Company identifier
44 #define APP_MAJOR_VALUE              0x00, 0x01 /**< Major value
45 #define APP_MINOR_VALUE              0x00, 0x02 /**< Minor value
46 #define APP_BEACON_UUID              0xe2, 0xc5, 0x6d, 0xb5, \
47                                     0xdf, 0xfb, 0x48, 0xd2, \
48                                     0xb0, 0x60, 0xd0, 0xf5, \
49                                     0xa7, 0x10, 0x96, 0xe0 /**< Proprietary
50
51 #define DEAD_BEEF                    0xDEADBEEF /**< Value to detect if
52
53 #define APP_TIMER_PRESCALER          0          /**< Value of the timer
54 #define APP_TIMER_OP_QUEUE_SIZE      4          /**< Size of the timer
55
56 #if defined(USE_UICR_FOR_MAJ_MIN_VALUES)
57 #define MAJ_VAL_OFFSET_IN_BEACON_INFO 18        /**< Position of the
58 #define UICR_ADDRESS                  0x10001080 /**< Address of the
59 #endif
60
61 static ble_gap_adv_params_t m_adv_params;      /**< Parameters for advertising
62 static uint8_t m_beacon_info[APP_BEACON_INFO_LENGTH] = /**< Information
63 {
64     APP_DEVICE_TYPE, // Manufacturer specific information. Specifies the device

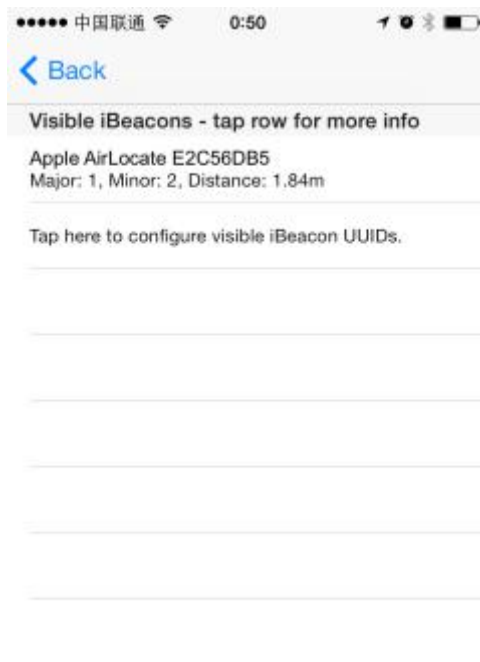
```

修改够编译通过。



点击 **Locate iBeacons** 进入如下界面，如果你的基站已经打开，即可看到如图所示的设备，下图中 就直接显示了你的手机与 **iBeacon** 基站间的距离，这个距离可能随着不同的硬件设备而有所差异，所以需要校准，请看下一步如何校准。这里为

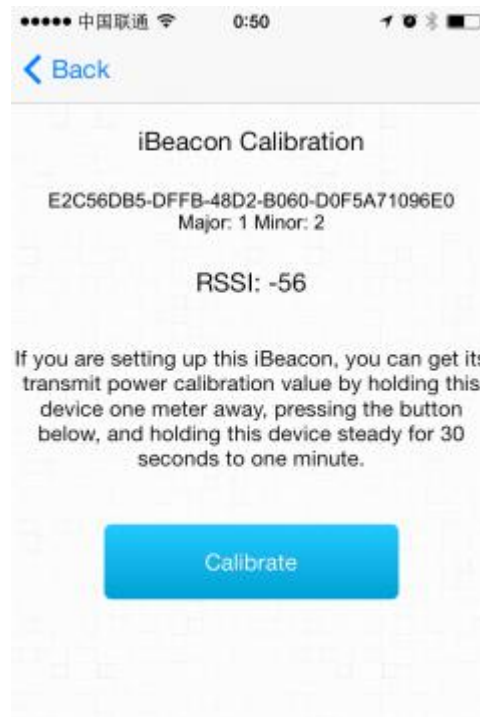
什么一打开这个软件就实现了距离显示了呢，主要是 该软件内默认添加了苹果公司发布的 **UUID**，我们亦可以修改基地的 **uuid**，然后在该软件中添加该 **uuid**，同样也能实现该功能的，这个做起来就要深入开发了。



校准， 点击第一行， 进入下一步。



校准提示, 把你的 **iphone** 放到距离 **iBeacon** 基站约 **1 米**处, 按下 **Calibrate** , 并且保持 **30 秒**到 **1 分钟**, 过程如下面图。



校准完毕，下面图显示聚焦校准完毕了，显示 **RSSI 为-57db**，这个为参考值。



重新 工程代码，修改基站的 **RSSI**，如下图所示，点击修改，修改成你校准的值：

```
28  lude "bsp.h"
29  lude "app_timer.h"
30
31  ine CENTRAL_LINK_COUNT          0          /**< Number of central links used by the application. When ch
32  ine PERIPHERAL_LINK_COUNT      0          /**< Number of peripheral links used by the application. When
33
34  ine IS_SRVC_CHANGED_CHARACT_PRESENT 0          /**< Include or not the service_changed characteristic. if no
35
36  ine APP_CFG_NON_CONN_ADV_TIMEOUT 0          /**< Time for which the device must be advertising in non-con
37  ine NON_CONNECTABLE_ADV_INTERVAL MSEC_TO_UNITS(100, UNIT_0_625_MS) /**< The advertising interval for non-connectable advertiseme
38
39  ine APP_BEACON_INFO_LENGTH      0x17          /**< Total length of information advertised by the Beacon. */
40  ine APP_ADV_DATA_LENGTH         0x16          /**< Length of manufacturer specific data in the advertisemen
41  ine APP_DEVICE_TYPE             0x02          /**< 0x02 refers to Beacon. */
42  ine APP_MEASURED_RSSI           0xC3          /**< The Beacon's measured RSSI at 1 meter distance in dBm. */
43  ine APP_COMPANY_IDENTIFIER      0x004c          /**< Company identifier for Nordic Semiconductor ASA. as per
44  ine APP_MAJOR_VALUE             0x00, 0x01          /**< Major value used to identify Beacons. */
45  ine APP_MINOR_VALUE             0x00, 0x02          /**< Minor value used to identify Beacons. */
46  ine APP_BEACON_UUID             0xe2, 0xc5, 0xd6, 0xb5, \
47                                  0xdf, 0xfb, 0x48, 0xd2, \
48                                  0xb0, 0x60, 0xd0, 0xf5, \
49                                  0xa7, 0x10, 0x96, 0xe0          /**< Proprietary UUID for Beacon. */
50
51  ine DEAD_BEEF                  0xDEADBEEF          /**< Value used as error code on stack dump, can be used to i
52
53  ine APP_TIMER_PRESCALER         0          /**< Value of the RIC1 PRESCALER register. */
54  ine APP_TIMER_OP_QUEUE_SIZE     4          /**< Size of timer operation queues. */
55
56  #defined (USE_UICR_FOR_MAJ_MIN_VALUES)
57  ine MAJ_VAL_OFFSET_IN_BEACON_INFO 18          /**< Position of the MSB of the Major Value in m_beacon_info
58  ine UICR_ADDRESS                0x10001080          /**< Address of the UICR register used by this example. The m
59  if
60
```

编译后重新下载， 然后回到 **Locate iBeacons** 软件 显示距离如下：

