蓝牙开发说明

`iOS` `BLE`

蓝牙基础开发

> 本文针对iOS开发

蓝牙的特点：

与WIFI通讯相比，蓝牙**低耗**，对于设备电量要求比较低；**速度快**,极限好像是20ms,超过这个时间，虽然代码不再执行，但低层代码还在不停的发送数据，容易使得手机发烫，除了特殊需求（音乐实时律动,手指取色发送），一般不要发送太快。

蓝牙相关的框架：

CoreBluetooth.frameWork

蓝牙相关的类：

CBCentralManager 中心设备 --> 手机

CBPeripheral 周边设备 --> 蓝牙设备

蓝牙的基本属性（特指在iOS开发中，常用到的一些属性）

- **UUID**：唯一标识符，可用来区分设备，具体请自行Google。
- **RSSI**: 信号强弱值，防丢器之类的可以用这个。
- **name**: 设备名称。
- **service UUID**（重点）:服务。一个Server 会包含多个characteristic，用UUID来区分。

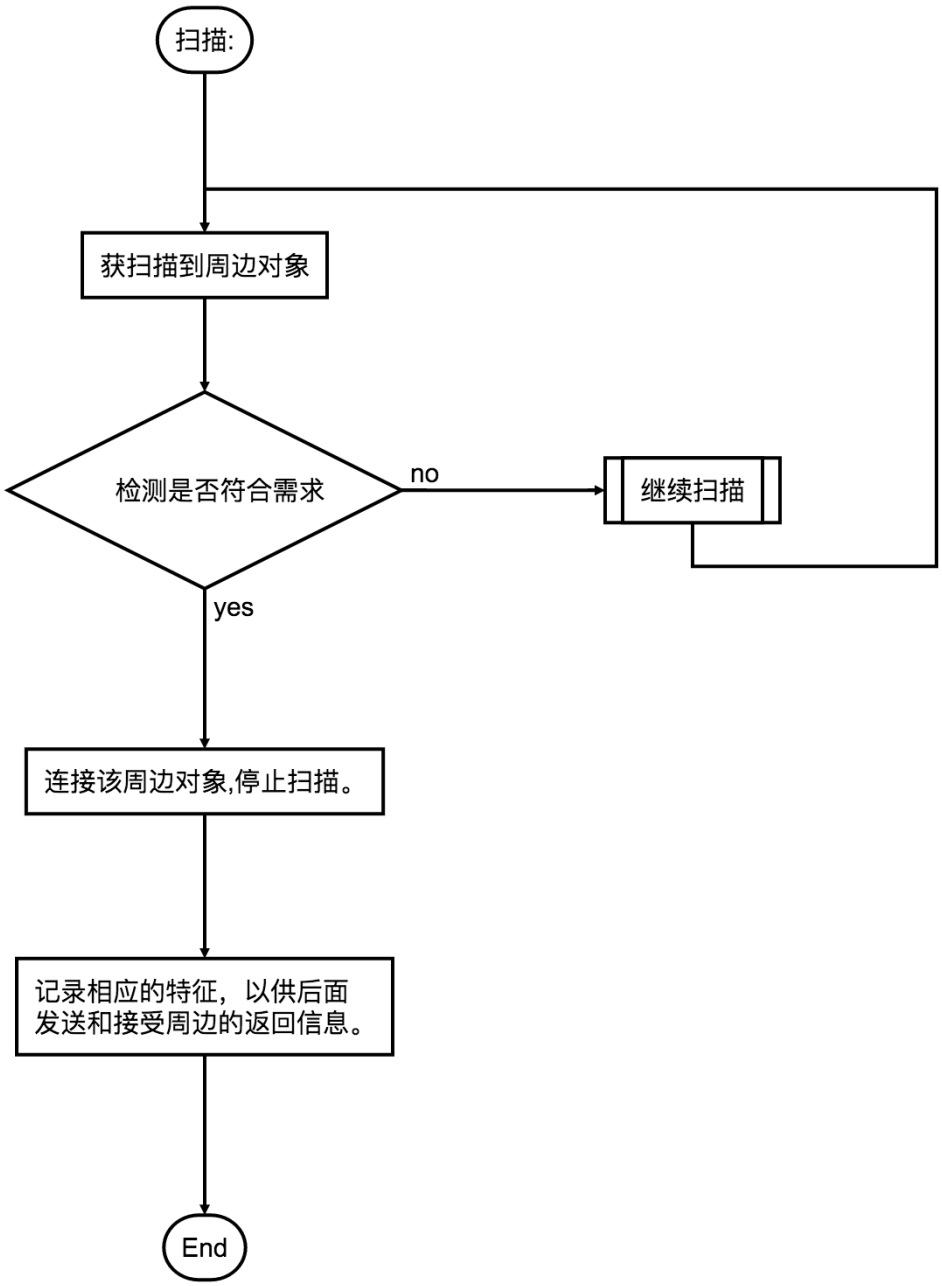
> 这个是有国际规范标注的，具体请自行Google，硬件的工程师应该比较了解。

- **characteristic**（重点）:特征。用UUID来区分。

> Characteristic 里面有个notify,只有打开才能接受到蓝牙的返回数据。Characteristic 里面也有很多属性，但本文不展开讲解。

具体例子讲解

蓝牙开发的流程



基本流程代码讲解

1. **创建手机中心** ，首先你要有一个中心CBCentralManager 这个中心代表你的设备，它提供了一系列的代理方法来管理与之相连的周边对象CBPeripheral。

```
CBCentralManager * BLEManager = [[CBCentralManager alloc] initWithDelegate:self queue:nil];
```

`BLEManager.delegate = self;`

2. **开始扫描** ，注意了扫描是耗性能、耗电、容易发热的，所以当找到并连接上设备的时候需要停止扫描。

   `if ([self isLECapableHardware]) {//判断是否打开蓝牙`

   `[BLEManager scanForPeripheralsWithServices:nil options:nil];}`

   `}`

3. **结束扫描** ，不听的扫描会相当的消耗电量和Cpu。

   ```
   [BLEManager stopScan];
   ```

4. **获取周边对象** ，扫描开始后，你会在代理方法里面获取到扫描的周边对象（再次提醒，**周边对象**就是代表本手机以外的蓝牙设备，比如手环、蓝牙灯、防丢器什么的。）

5. **获取服务** ，通过service UUID在目标周边对象的所有服务中找到你需要的服务。

6. **获取特征** ， 同样通过characteristic UUID 获取到相应的特征，保存成全局变量，后面发送会用到。

7. **发送数据** ， 发送数据需要至少需要五个参数：手机、连接上的周边对象、特征、数据、是否返回。

8. **接收返回数据** ，在这里接收、处理周边对象的返回数据，注意这里一共提供了几个参数，具体使用看需求而定。

> **强烈建议大家把 CBCentralManager.h 里面的代理方法全部看一遍（就是翻译一遍，别偷懒）。**

tips

- **编辑数据**

```
//拆分高低位
Byte bindIndexLow = needIndex & 0xff;
Byte bindIndexHigh = (needIndex >> 8) & 0xff;

int i == 0
char char_array[30] = {0x00};//定义一个字节数组

char_array[i++] = 0x55;  //16进制
char_array[i++] = 20;  //10进制
char_array[i++] = 0x8F;  //
char_array[i++] = 0x07;   //

NSData* data = [NSData dataWithBytes:(const void *)char_array length:sizeof(char) * i]
;

[peripheral writeValue:data forCharacteristic:calibrationCharacteristic
type:CBCharacteristicWriteWithResponse];
```

- **检验工具 lightblue**

墙裂推挤使用lightblue 检测蓝牙设备，它能读取到蓝牙的所有参数，特别需要注意的是里面的serveruuid 和 特征，里面的notify和read。理论上来说 lightblue 上能完成的操作 app也应该能完成 同时这也是开发中用来检验硬件收发数据是否正常，如果lightblue 跟硬件通讯正常，问题极有可能出在app上。

- **重连**

  iOS的框架自带重连机制，千万不要因为设备断开就手动调用立马扫描，不要扫描，不要扫描。具体看 Core Bluetooth Programming Guide

- **多连**

  iOS框架支持多连，同样只要connect就行，当然要注意区分不同的设备

- **防劫持**

  很多防丢器和一些家庭级的蓝牙会有防劫持功能，具体得看需求和硬件而定,一般来说要求手机在限定时间内提交密码。

- **进阶一：队列发送**

  由于蓝牙的发送性能的限制，如果你需要高速发送数据时，尽量把发送封装成一个工具类，把需要发送的数据存到一个数组队列里面，然后定时发送数据。

- **进阶二：丢包重发**

  结合队列发送，如果业务需求要求收发严密，不许丢包。可以对每一包发送的数据进行临时保存，在一定时间内没有接收到返回就发送第二次，发送三次后仍无返回即发送下一包数据或提示用户等业务逻辑处理。