HZLBlueTooth_V1.0 SDK for iOS

Date: 08 30, 2019

Author: Liang Fang

SDK Version: 1.0

MCU: 2.3

Content

HZLBlueTooth Development Guide

Introduction

This guide will teach you how to use HZLBlueTooth SDK for iOS to write iOS applications that can acquire brainwave data from Macrotellect 's Hardware (BrainLink Pro & BrainLink Lite). This will enable your iOS apps to receive and use brainwave data such as BLEMIND and BLEGRAVITY acquired via Bluetooth, Macrotellect 's Hardware and File source encapsulated as HZLBlueTooth. HZLBlueTooth SDK for iOS supports upgrading Hardware

Function:

Receive brainwave data. Only one Bluetooth device can be connected at a time

Files included:

- API Reference (this document)
- SDK static library and headers
- libHzlBlueTooth_V1.0.a
- HZLBlueData.h
- Blue3OrBlue4Manager.h
- IOS_HZLBlue4.0Demo example project for iOS

Supported devices:

- Bluetooth 4.0 BLE
  - BrainLink_Pro
  - Jii
- Bluetooth 3.0
  - BrainLink_Lite
  - Mind Link

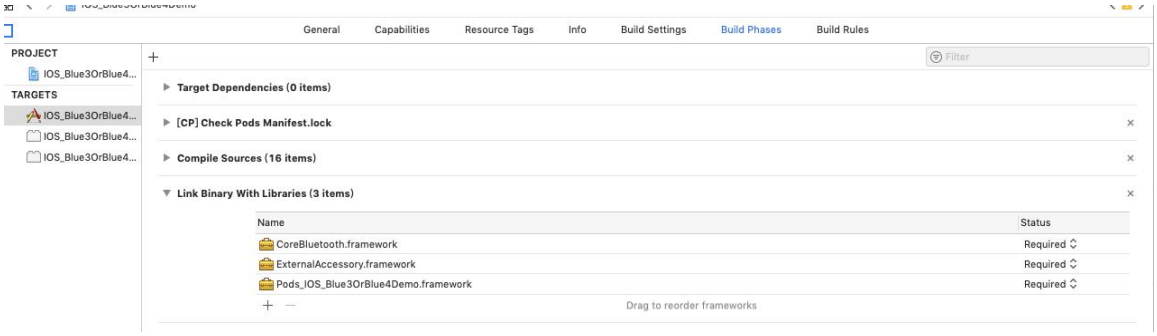
iOS Verrsion:

- iOS 9.0 +

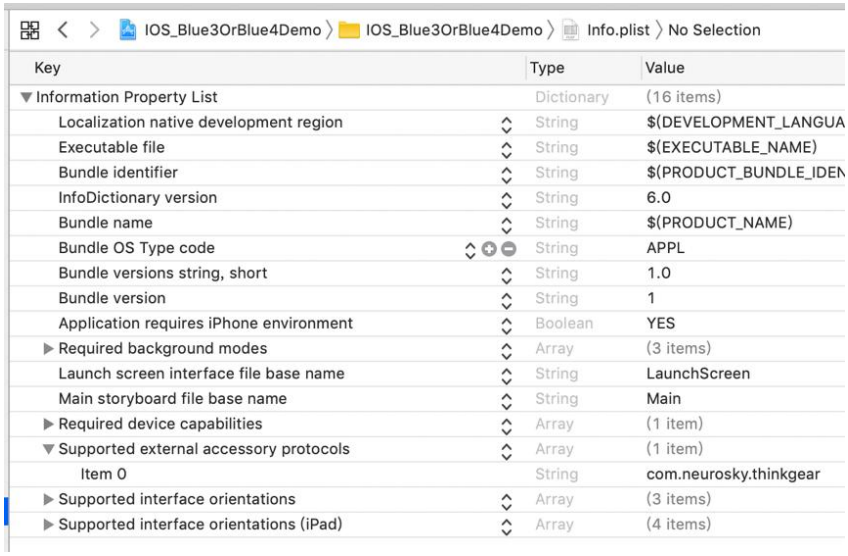
Your First Project: IOS_Blue3OrBlue4Demo

Step 1:

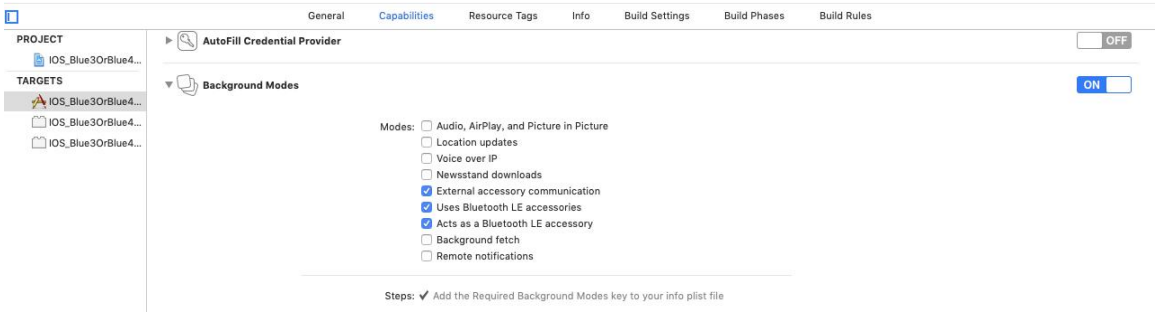
1.1 Import the IOS framework libraries CoreBluetooth.framework
and ExternalAccessory.framework in the Build Phases of TARGETS in the Xcode project:



Add com.neurosky.thinkgear In the Info.plist:（ios13 needs to add the Bluetooth permission privacy - Bluetooth always usage description and privacy - Bluetooth peripheral usage description）



1.2 If you want Bluetooth to work in the background, please set it as follows，
Don't set it if you don't need it:



Step 2:

Import header file

```
#import "HZLBlueData.h"
#import "Blue3OrBlue4Manager.h"
```

Function one：Receive data

```
// Bluetooth multi-connection
// blueNames: connectable device
NSArray *blue3Name = @[@"BrainLink",@"BrainLink_Pro",@"BrainLink_Lite"];
[Blue3OrBlue4Manager logEnable:YES];
[[Blue3OrBlue4Manager shareInstance] configureBlue3MFiOrBlue4Names:blue3Name];

// Connect bluetooth callback successfully
  __weak FactoryViewController *weakSelf = self;
[Blue3OrBlue4Manager shareInstance].blueConBlock =   ^(BlueType conBT){
      // Determine connected devices
      if (conBT == BlueType_3) {
            NSLog(@"Bluetooth 3.0 Pro device connected");
      }
      else if(conBT == BlueType_4Pro)
      {
            NSLog(@"Bluetooth 4.0 Pro device connected");
      }
      else if (conBT == BlueType_4Jii){
            NSLog(@"Bluetooth 4.0 jii device connected");
      }
};


// Bluetooth disconnect callback
[Blue3OrBlue4Manager shareInstance].blueDisBlock =   ^(BlueType disBT){
      if (disBT == BlueType_3) {
            NSLog(@"Bluetooth 3.0 device disConnected ");
      }
      else if(disBT == BlueType_4Pro)
```

```
        {
            NSLog(@"Bluetooth 4.0 Pro device disConnected");
        }
        else if(disBT == BlueType_4Jii)
        {
            NSLog(@"Bluetooth 4.0 jii device disConnected");
        }

        weakSelf.signallv.image = [UIImage imageNamed:@"noSignal"];

        weakSelf.attentionlabel.text = @"";
        weakSelf.medlabel.text = @"";
        weakSelf.electricityLabel.text = @"";
        weakSelf.favrouteRateLabel.text = @"";
        weakSelf.otherLabel.text = @"";

        weakSelf.circleRateLabel.text = @"";

        weakSelf.rawLabel.text = @"";
        weakSelf.pDataLabbel.text= @"";
    };


    [Blue3OrBlue4Manager shareInstance].hzlblueDataBlock = ^(HZLBlueData *blueData, BlueType conBT){

        if (conBT == BlueType_4Pro) {
            NSString *peripID = [blueData.identifier substringWithRange:NSMakeRange(blueData.identifier.length - 5, 4)];
            if (blueData.bleDataType == BLEMIND) {

                weakSelf.attentionlabel.text = [NSString stringWithFormat:@"%@=%d",peripID,blueData.attention];
                weakSelf.medlabel.text = [NSString stringWithFormat:@"%@=%d",peripID,blueData.meditation];
                weakSelf.electricityLabel.text = [NSString stringWithFormat:@"%@=%d",peripID, blueData.batteryCapacity];
                weakSelf.favrouteRateLabel.text = [NSString stringWithFormat:@"%@=%d",peripID,blueData.ap];

                weakSelf.otherLabel.text = [NSString stringWithFormat: @"%@=Delta:%d Theta:%d LowAlpha:%d HighAlpha:%d LowBeta:%d HighBeta:%d LowGamma:%d HighGamma:%d Hardwareversion:%d grid=%d",peripID,blueData.delta,blueData.theta,blueData.lowAlpha,blueData.highAlpha,blueData.lowBeta,blueData.highBeta,blueData.lowGamma,blueData.highGamma,blueData.hardwareVersion,blueData.grind];
```

```
                    // when the signal value is 0, the bluetooth device is worn
                    // note: if the bluetooth device is connected but not worn, Greater than 0 and less than or equal to 100
                    if(blueData.signal == 0){
                        weakSelf.signallv.image = [UIImage imageNamed:@"signal_zhengChang"];
                    }else{
                        weakSelf.signallv.image = [UIImage imageNamed:@"signal3"];
                    }
                }
                if (blueData.bleDataType == BLEGRAVITY) {
                    weakSelf.circleRateLabel.text = [NSString stringWithFormat:@"%@=x:%d y:%d z:%d ",peripID,blueData.xvlaue,blueData.yvlaue,blueData.zvlaue];
                }
                if(blueData.bleDataType == BLERaw)
                {
                    weakSelf.rawLabel.text = [NSString stringWithFormat:@"Blue3=Raw:%d Blinkeye:%d",blueData.raw,blueData.blinkeye];
                }


            }
            else if (conBT == BlueType_4Jii){
                NSString *peripID = [blueData.identifier substringWithRange:NSMakeRange(blueData.identifier.length - 5, 4)];
                if (blueData.bleDataType == BLEMIND){

                    weakSelf.attentionlabel.text = [NSString stringWithFormat:@"%@=%d",peripID,blueData.attention];
                    weakSelf.medlabel.text = [NSString stringWithFormat:@"%@=%d",peripID,blueData.meditation];
                    weakSelf.electricityLabel.text = [NSString stringWithFormat:@"%@=%d",peripID, blueData.batteryCapacity];
                    weakSelf.favrouteRateLabel.text = [NSString stringWithFormat:@"%@=%d ",peripID,blueData.ap];

                    if(blueData.signal == 0){
                        weakSelf.signallv.image = [UIImage imageNamed:@"signal_zhengChang"];
                    }else{
                        weakSelf.signallv.image = [UIImage imageNamed:@"signal3"];
                    }

                }


            }
            else if (conBT == BlueType_3){
                if (blueData.bleDataType == BLEMIND){
```

```
                weakSelf.attentionlabel.text = [NSString stringWithFormat:@"Blue3=%d",blueData.attention];
                weakSelf.medlabel.text = [NSString stringWithFormat:@"Blue3=%d",blueData.meditation];

                weakSelf.otherLabel.text = [NSString stringWithFormat: @"Blue3=Delta:%d Theta:%d LowAlpha:%d HighAlpha:%d LowBeta:%d HighBeta:%d LowGamma:%d HighGamma:%d ",blueData.delta,blueData.theta,blueData.lowAlpha,blueData.highAlpha,blueData.lowBeta,blueData.highBeta,blueData.lowGamma,blueData.highGamma];

                if(blueData.signal == 0){
                    weakSelf.signallv.image = [UIImage imageNamed:@"signal_zhengChang"];
                }else{
                    weakSelf.signallv.image = [UIImage imageNamed:@"signal3"];
                }
            }

            if(blueData.bleDataType == BLERaw){
                weakSelf.rawLabel.text = [NSString stringWithFormat:@"Blue3=Raw:%d Blinkeye:%d",blueData.raw,blueData.blinkeye];
            }

        }
    };

  [[Blue3OrBlue4Manager shareInstance] connectBlue3OrBlue4];

    // Active bluetooth disconnect
  [[Blue3OrBlue4Manager shareInstance]disConnectBlue3OrBlue4];
```

HZLBlueTooth API Reference

HZLBlueData Reference

Overview
The HZLBlueData class is a data model

Enum
```
typedef enum : NSUInteger {
    BlueType_NO = 0,
BlueType_3,
/*The current connections are BrainLink_Lite, etc. (bluetooth 3.0 devices), with BLEMIND, BLEGRAVITY, and BLERaw type data*/
BlueType_4Pro,
/* The current connection is BrainLink_Pro(bluetooth 4.0 device) with BLEMIND, BLEGRAVITY, and BLERaw type data*/
    BlueType_4Jii,
/* The current connection is Jii(bluetooth 4.0 device) */
}BlueType;

        typedef NS_ENUM(NSUInteger,BLEDATATAYPE){
         BLEMIND   =   0,            // basic brain wave data
         BLEGRAVITY,                 // gravity data
         BLERaw,                     // blink data
};
```

Basic Brainwave Data：
- signal,
- attention,
- meditation,
- delta,
- theta,
- lowAlpha,
- highAlpha,
- lowBeta,
- highBeta,
- lowGamma,
- highGamma,
- ap,
- batteryCapacity,
- hardwareVersion,
- grind

Gravity Sensor Data:

- xvlaue,
- yvlaue,
- zvlaue

Raw& Blink Data:

- raw,
- blinkeye

Note：

When Jii is connected, only signal, attention, meditation, batteryCapacity, ap data type is available.
When BrainLink_Lite is connected, only signal，attention，meditation，delta，theta，lowAlpha，highAlpha，lowBeta，highBeta，lowGamma， highGamma，raw，blinkeye data type is available.

Instructions of some Instance Property

- **signal:** It represents the signal value of the Macrotellect 's Hardware. When the signal is 0, it means that the Macrotellect 's Hardware has been put on, and when the signal is 200, it means that the Macrotellect 's Hardware is connected to the iPhone.
- **batteryCapacity:** In percentage terms. minimum value is 0, maximum value is 100
- **ap:** Appreciation value
- **hardwareVersion:** Hardware version. The first version value is 255， when you update the Macrotellect 's Hardware，the version value will be smaller.
- **xvlaue：** gravity value in The x axis (Pitching Angle)
- **yvlaue：** gravity value in The y axis (Yaw Angle)
- **zvlaue：** gravity value in The z axis (Roll Angle)

ConnectBlueManager Reference

Overview

The ConnectBlueManager class handles interaction between a Macrotellect's Hardware and an iOS device.

Instance Property

Successful callback of bluetooth connection

@property (nonatomic,copy)Blue3OrBlue4Connect blueConBlock;

Bluetooth disconnect callback

@property (nonatomic,copy) BlueConnectdismiss blueDisBlock;

Data callback for device.

@property(nonatomic,copy)Blue3OrBlue4DataBlock hzlblueDataBlock;
_E;

Connection status of blue3.0 device

@property (nonatomic,assign)BOOL connected3;

Connection status of blue4.0 device

@property (nonatomic,assign)BOOL connected4;

Method

Print log does not print by default

+ (void)logEnable:(BOOL)enable;

Initialization (singleton)

+ (instancetype)shareInstance;

Parameter Configuration：

Parameter interpretation：

blue3MFiOrBlue4Names: **Able to connect to bluetooth 4.0 device name and bluetooth 3.0 MFI**

-(void)configureBlue3MFiOrBlue4Names:(NSArray *)blue3MFiOrBlue4Names；

Connect bluetooth device

-(void)connectBlue3OrBlue4;

Disconnect bluetooth device

-(void)disConnectBlue3OrBlue4;