IOS Bluetooth Program

Pei.ZHANG pei.zhang@student.yncrea.fr

Project Objective

Write a simple Bluetooth demo program to connect the microcontroller. Can send data to control the movement of the wheelchair, or can receive data to get the location of the wheelchair.

Completed part

The program can now scan for surrounding Bluetooth devices. After pressing to stop scanning, all connectable devices will be displayed in the menu bar. Click on the device you want to connect to connect. After that, the program will find the corresponding service and characteristics according to uuid. If not found, it will automatically disconnect. After the connection is successfully established, the received data will be displayed in the text box. If you click the move button, a data will be sent to the microcontroller (it has no meaning, it is only used as a test). All data are represented in hexadecimal asicc codes.

Unfinished part

Convert the received data into the form of coordinates, and add a virtual joystick to control the movement of the wheelchair, and you can also add the function of voice control

project description

This program uses a central model

Import CoreBluetooth framework, #import <CoreBluetooth / CoreBluetooth.h>

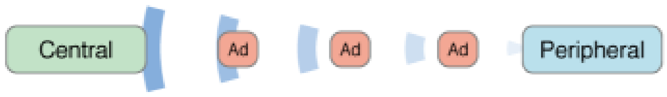
Comply with CBCentralManagerDelegate, CBPeripheralDelegate agreement

central: The device connected to the hardware.

peripheral: peripheral, connected hardware.

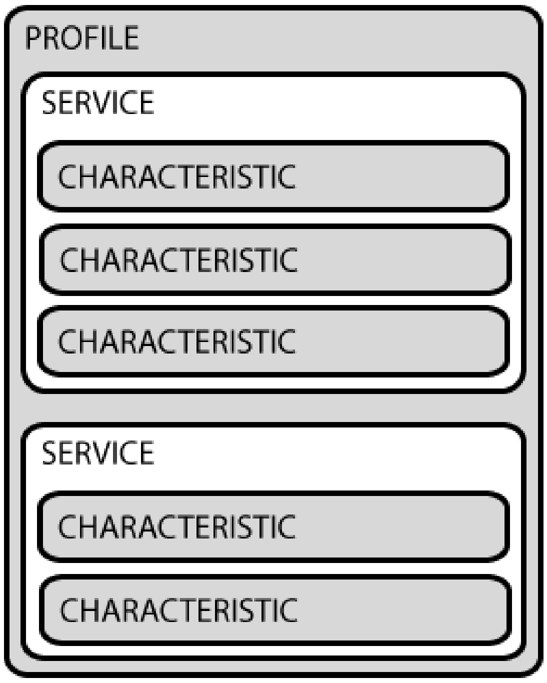
Note: Peripherals are always broadcasting.When the central object you create is scanning peripherals, you can discover peripherals.

as the picture shows:



Note: A peripheral device contains multiple services, and each service contains multiple features. Features include feature values and feature descriptions. Each service contains multiple fields. The permissions of the fields are read, write, notify.

as the picture shows:



Central Model Process

1. Establish the central role [[CBCentralManager alloc] initWithDelegate: self queue: nil]
2.Scan peripherals: cancelPeripheralConnection
3. Discover the peripheral: didDiscoverPeripheral
4. connect peripherals: connectPeripheral
  4.1 connection failed: didFailToConnectPeripheral
  4.2 disconnected: didDisconnectPeripheral
  4.3 Connected successfully: didConnectPeripheral
5. Scan services in peripherals: discoverServices
  5.1 Discovering and Obtaining Services in Peripherals: didDiscoverServices
6. Scan the characteristics of corresponding services of peripherals: discoverCharacteristics
  6.1 Discover and obtain the characteristics of the corresponding service of the peripheral: didDiscoverCharacteristicsForService
  6.2 Write data to the corresponding feature writeValue: forCharacteristic: type:
7. Notification of subscription characteristics setNotifyValue: forCharacteristic:
  7.1 Read data based on characteristics: didUpdateValueForCharacteristic

Note: After the iOS version is updated, the Bluetooth certificate may also need to be updated or it cannot be used normally. For details, please refer to the official document.

Demo Program:

https://github.com/keroro1998/Knight-Rider

Apple's official Bluetooth development guide:

https://developer.apple.com/library/archive/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/AboutCoreBluetooth/Introduction.html#//apple_ref/doc/uid/TP40013257-CH1-SW1

iOS development series-UI simple TableView use(Development of the menu bar)：
https://blog.csdn.net/xiaozhuanddapang/article/details/70210843
https://github.com/chengxuyuanxiaoka/SimpleTableView

Virtual joystick development guide(Unfinished part)

https://www.jianshu.com/p/176d06d8cdc6

https://blog.csdn.net/jieyar/article/details/8893243

https://blog.csdn.net/WMX843230304WMX/article/details/78301817