CH9140 Framework 库接口函数说明

版本：1A

http://wch.cn

一、概述

CH9140 Framework是iOS系统下的BLE串口接口库，用于为CH914X低功耗蓝牙转串口的芯片模组，提供BLE串口设备的自动搜索连接、打开关闭、串口参数设置、MODEM操作、串口读写等操作API函数。

支持iOS11及以上系统，蓝牙控制器4.0及以上版本。

二、使用流程

1. 通过 shareManager 方法获取 CH9140BluetoothManager 单例对象，设置 isDebug 为 true 开启 debug 模式，查看更多日志输出
2. 设置 CH9140BluetoothManager 的 delegate 对象
3. 通过 startEnumDevicesWithOptions 扫描设备
4. 代理中实现 ch9140ManagerDidDiscoverPeripheral 获取扫描到的设备
5. 通过 openDevice 打开设备
6. 代理中实现 ch9140DidPeripheralConnectUpateState 获取蓝牙连接状态变更
7. 代理中实现 ch9140DidSerialModemNotify：注册监听上报串口以及 Modem 状态的通知
8. 通过 setSerialBaud:dataBit:stopBit:parity:paritywithBlock:设置波特率及其他参数
9. 通过 setSerialModem:dtr:rts:withBlock:设置流控开关及 Modem 状态
10. 调用方法写入数据 ch9140DidWrite
11. 代理中实现该方 ch9140DidRead:获取读取的数据
12. 调用 stopEnumDevices 方法停止扫描
13. 调用 closeDevice 方法关闭设备

三、Framework 中对外接口方法介绍

3.1. 库初始化

```
+ (CH9140BluetoothManager *)shareManager;
```

该方法是个单例，会生成全局 CH9140BluetoothManager 管理对象。

3.2. 开始枚举设备

```
-(void)startEnumDevicesWithOptions:(NSDictionary *)options;
```

该函数会在规定时间扫描附近设备并通过结构体数组返回扫描结果。

options 是个字典 key-value 形式。

3.3. 列出扫描出的设备

```
- (void)ch9140ManagerDidDiscoverPeripheral:(CBPeripheral *)peripheral
                                      RSSI:(NSNumber *)RSSI
                                     error:(NSError*)error;
```

该方法在代理中实现。当扫描到设备的时候会通知代理的该方法。
peripheral- 扫描发现的外设。
RSSI-外设的信号值。
error- 扫描出错的错误信息。

3.4. 停止枚举设备

```
-(void)stopEnumDevices;
```

该方法会停止中心设备的扫描操作

3.5. 打开设备

```
-(void)openDevice:(CBPeripheral *)peripheral;
```

该方法会连接指定的设备，连接后设备的状态会发生改变，会通过代理发送状态变更通知。

3.6. 关闭设备

```
-(void)closeDevice:(CBPeripheral *)peripheral;
```

该方法会关闭指定的设备连接.设备连接关闭后，设备连接的状态会发生改变，会通过代理发送状态变更通知。

3.7. 注册监听上报串口以及 Modem 状态的通知

```
-(void)ch9140DidSerialModemNotify:(ModemNotifyItem *)modemNotifyItem
```

该方法注册监听上报串口及 Modem 状态的通知,只需要在代理类中实现该方法，上报串口以及 Modem 状态会通知该方法。

3.8. 设置波特率以及其他参数

```
-(void)setSerialBaud:(NSInteger)baudRate
            dataBit:(NSInteger)dataBit
            stopBit:(NSInteger)stopBit
             parity:(NSInteger)parity
          withBlock:(SerialBaudBlock)setSerialBaudBlock;
```

该方法设置串口的波特率以及其他参数
baudRate-波特率： 1200、2400、4800、9600、19200、38400、57600、115200、230400
dataBit-数据位：5/6/7/8
stopBit-停止位：1/2
parity-校验位 ：0(无校验)/1(奇校验)/2(偶校验)/3(标志)/4(空白位)
该方法需要传递一个 block 来回调通知设置的结果

```
typedef void (^SerialBaudBlock) (Boolean returnValue,NSString *info);
```

returnValue - 设置的结果成功/失败
info - 附带的信息

3.9. 设置流控开关及 Modem 状态

```
-(void)setSerialModem:(Boolean)flow
                dtr:(NSInteger)dtr
                rts:(NSInteger)rts
          withBlock:(SerialModemBlock)serialModemBlock;
```

该方法设置串口的波特率以及其他参数
flow-硬件流控开发 true 开 false 管

dtr-1 有效低电平，0 无效高电平
rts- 1 有效低电平，0 无效高电平

该方法需要传递一个 block 来回调通知设置的结果
typedef void (^SerialModemBlock) (Boolean returnValue,NSString *info);
returnValue - 设置的结果成功/失败
info - 附带的信息

3.10. 监听蓝牙的状态变更

-(void)ch9140DidPeripheralConnectUpateState:(CBPeripheral *)peripheral
error:(NSError *)error;

只需要在代理中实现该方法，当蓝牙状态发生变更的时候，可以通知代理蓝牙连接状态变更

CBPeripheralStateDisconnected -取消连接
CBPeripheralStateConnecting,-连接中
CBPeripheralStateConnected,-已连接

3.11.向外设写入数据

-(NSInteger)ch9140DidWrite:(NSData *)data;
写入二进制数据，返回值是写入的数据长度

3.12. 根据写入类型获取写入的数据的最大长度

-(NSInteger)ch9140MaxWriteValueLengthForType:(CBCharacteristicWriteType)type;
CBCharacteristicWriteWithResponse -有响应写,
CBCharacteristicWriteWithoutResponse - 无响应写,

3.13. 获取读取到的数据

-(NSInteger)ch9140DidRead:(NSData *)data;
该函数用于获取读取特征值。只需要再代理中实现该方法，当有值存在会回调该方法

3.14. 获取类库的版本号

-(NSString *)managerVersion;
返回当前 framework 的版本号