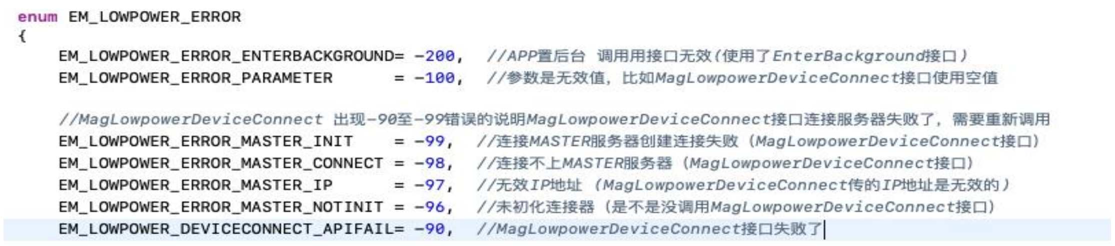
低功耗 API(IOS 版本)

1.- (int)MagLowpowerDeviceConnect:(NSString*) strIP;

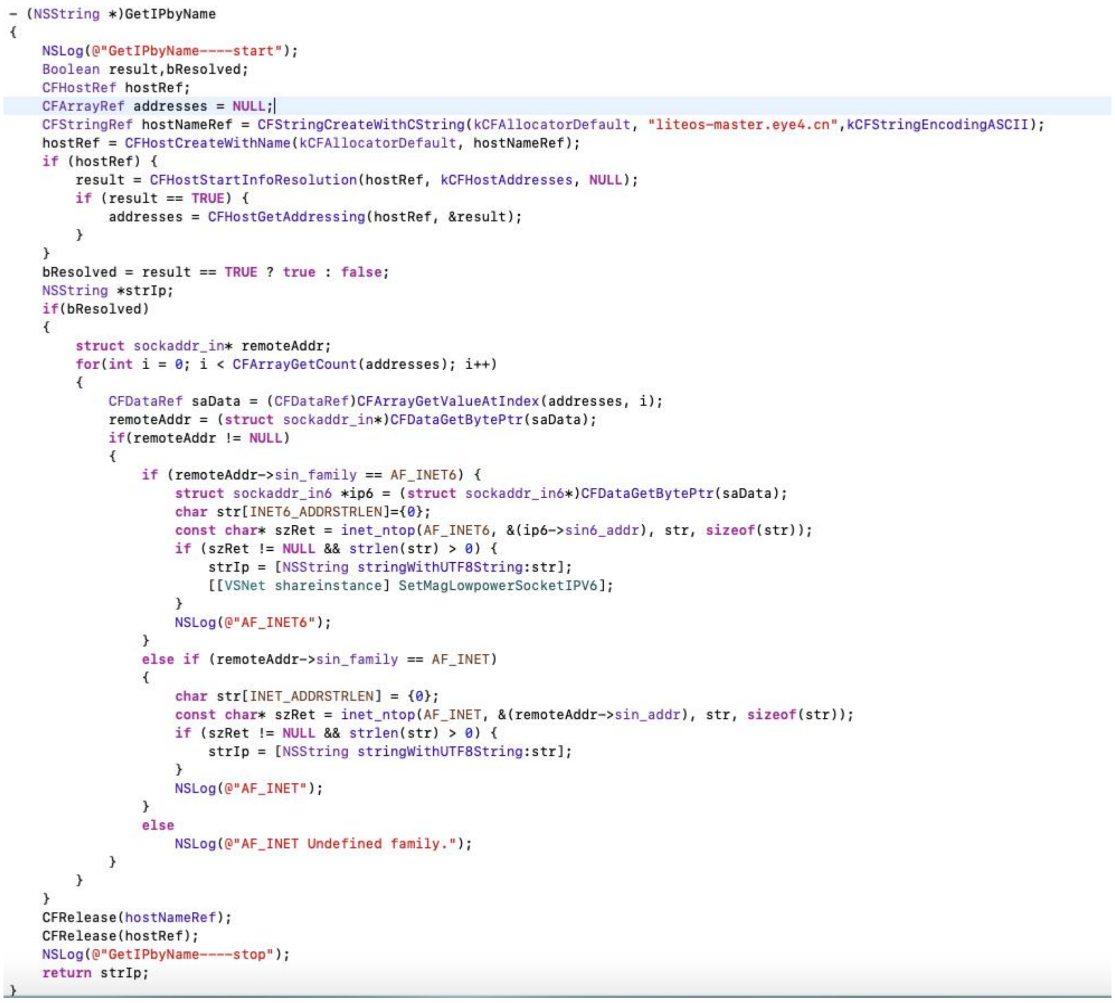
连接低功耗服务器，返回 1 代表成功，其它代表是失败的。（在 MagLowpowerInitDevice 前一定要调用此接口，连上服务器）



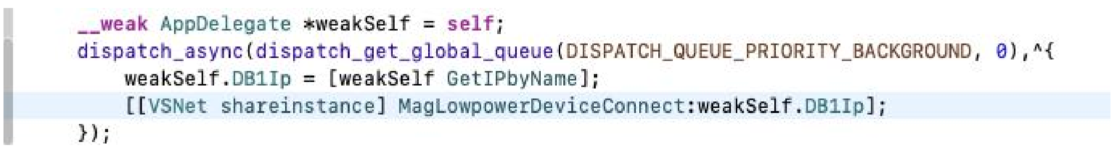
```
enum EM_LOWPOWER_ERROR
{
    EM_LOWPOWER_ERROR_ENTERBACKGROUND= -200,   //APP置后台 调用用接口无效(使用了EnterBackground接口)
    EM_LOWPOWER_ERROR_PARAMETER      = -100,   //参数是无效值，比如MagLowpowerDeviceConnect接口使用空值

    //MagLowpowerDeviceConnect 出现-90至-99错误的说明MagLowpowerDeviceConnect接口连接服务器失败了，需要重新调用
    EM_LOWPOWER_ERROR_MASTER_INIT    = -99,   //连接MASTER服务器创建连接失败（MagLowpowerDeviceConnect接口）
    EM_LOWPOWER_ERROR_MASTER_CONNECT = -98,   //连接不上MASTER服务器（MagLowpowerDeviceConnect接口）
    EM_LOWPOWER_ERROR_MASTER_IP      = -97,   //无效IP地址（MagLowpowerDeviceConnect传的IP地址是无效的）
    EM_LOWPOWER_ERROR_MASTER_NOTINIT = -96,   //未初化连接器（是不是没调用MagLowpowerDeviceConnect接口）
    EM_LOWPOWER_DEVICECONNECT_APIFAIL= -90,   //MagLowpowerDeviceConnect接口失败了
```

例如————————————————————————————————



```
- (NSString *)GetIPbyName
{
    NSLog(@"GetIPbyName----start");
    Boolean result,bResolved;
    CFHostRef hostRef;
    CFArrayRef addresses = NULL;
    CFStringRef hostNameRef = CFStringCreateWithCString(kCFAllocatorDefault, "liteos-master.eye4.cn",kCFStringEncodingASCII);
    hostRef = CFHostCreateWithName(kCFAllocatorDefault, hostNameRef);
    if (hostRef) {
        result = CFHostStartInfoResolution(hostRef, kCFHostAddresses, NULL);
        if (result == TRUE) {
            addresses = CFHostGetAddressing(hostRef, &result);
        }
    }
    bResolved = result == TRUE ? true : false;
    NSString *strIp;
    if(bResolved)
    {
        struct sockaddr_in* remoteAddr;
        for(int i = 0; i < CFArrayGetCount(addresses); i++)
        {
            CFDataRef saData = (CFDataRef)CFArrayGetValueAtIndex(addresses, i);
            remoteAddr = (struct sockaddr_in*)CFDataGetBytePtr(saData);
            if(remoteAddr != NULL)
            {
                if (remoteAddr->sin_family == AF_INET6) {
                    struct sockaddr_in6 *ip6 = (struct sockaddr_in6*)CFDataGetBytePtr(saData);
                    char str[INET6_ADDRSTRLEN]={0};
                    const char* szRet = inet_ntop(AF_INET6, &(ip6->sin6_addr), str, sizeof(str));
                    if (szRet != NULL && strlen(str) > 0) {
                        strIp = [NSString stringWithUTF8String:str];
                        [[VSNet shareinstance] SetMagLowpowerSocketIPV6];
                    }
                    NSLog(@"AF_INET6");
                }
                else if (remoteAddr->sin_family == AF_INET)
                {
                    char str[INET_ADDRSTRLEN] = {0};
                    const char* szRet = inet_ntop(AF_INET, &(remoteAddr->sin_addr), str, sizeof(str));
                    if (szRet != NULL && strlen(str) > 0) {
                        strIp = [NSString stringWithUTF8String:str];
                    }
                    NSLog(@"AF_INET");
                }
                else
                    NSLog(@"AF_INET Undefined family.");
            }
        }
    }
    CFRelease(hostNameRef);
    CFRelease(hostRef);
    NSLog(@"GetIPbyName----stop");
    return strIp;
}
```



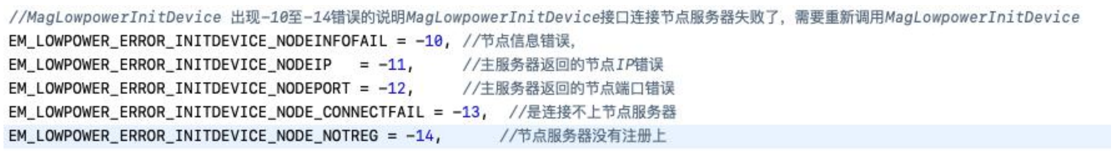
```
    __weak AppDelegate *weakSelf = self;
    dispatch_async(dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_BACKGROUND, 0),^{
        weakSelf.DB1Ip = [weakSelf GetIPbyName];
        [[VSNet shareinstance] MagLowpowerDeviceConnect:weakSelf.DB1Ip];
    });
```

2.– (void)MagLowpowerDeviceDisconnect;

断开低功耗服务器的连接。

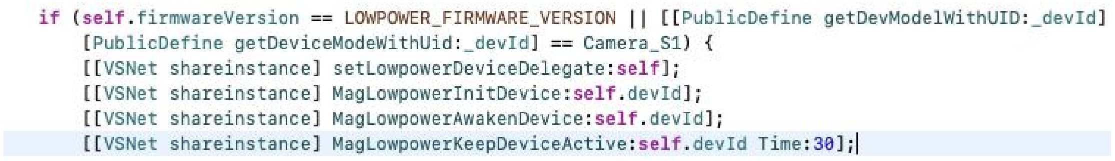
3.– (int)MagLowpowerInitDevice:(NSString *)deviceIdentity;

初始化注册低功耗设备，返回 1 代表成功，其它代表是失败的。



```
//MagLowpowerInitDevice 出现-10至-14错误的说明MagLowpowerInitDevice接口连接节点服务器失败了，需要重新调用MagLowpowerInitDevice
EM_LOWPOWER_ERROR_INITDEVICE_NODEINFOFAIL = -10, //节点信息错误,
EM_LOWPOWER_ERROR_INITDEVICE_NODEIP    = -11,      //主服务器返回的节点IP错误
EM_LOWPOWER_ERROR_INITDEVICE_NODEPORT = -12,      //主服务器返回的节点端口错误
EM_LOWPOWER_ERROR_INITDEVICE_NODE_CONNECTFAIL = -13,  //是连接不上节点服务器
EM_LOWPOWER_ERROR_INITDEVICE_NODE_NOTREG = -14,      //节点服务器没有注册上
```

例如：——————————————————————————————



```
    if (self.firmwareVersion == LOWPOWER_FIRMWARE_VERSION || [[PublicDefine getDevModelWithUID:_devId]
        [PublicDefine getDeviceModeWithUid:_devId] == Camera_S1) {
        [[VSNet shareinstance] setLowpowerDeviceDelegate:self];
        [[VSNet shareinstance] MagLowpowerInitDevice:self.devId];
        [[VSNet shareinstance] MagLowpowerAwakenDevice:self.devId];
        [[VSNet shareinstance] MagLowpowerKeepDeviceActive:self.devId Time:30];
```

———————————————————————————————

4.– (int)MagLowpowerAwakenDevice:(NSString *)deviceIdentity

唤醒设备返回 1 代表成功，其它代表是失败的。

5.– (int)MagLowpowerGetDeviceStatus:(NSString *)deviceIdentity

向服务器查询低功耗设备状态

6.–(int)MagLowpowerKeepDeviceActive:(NSString *)deviceIdentity Time:(int) time

保活设备通讯，此接口用于 P2P 连接上后保活设备不让其睡眠，返回 1 代表成功，其它代表是失败的

7.–(int)MagLowpowerRemoveKeepDeviceActive:(NSString *)deviceIdentity;

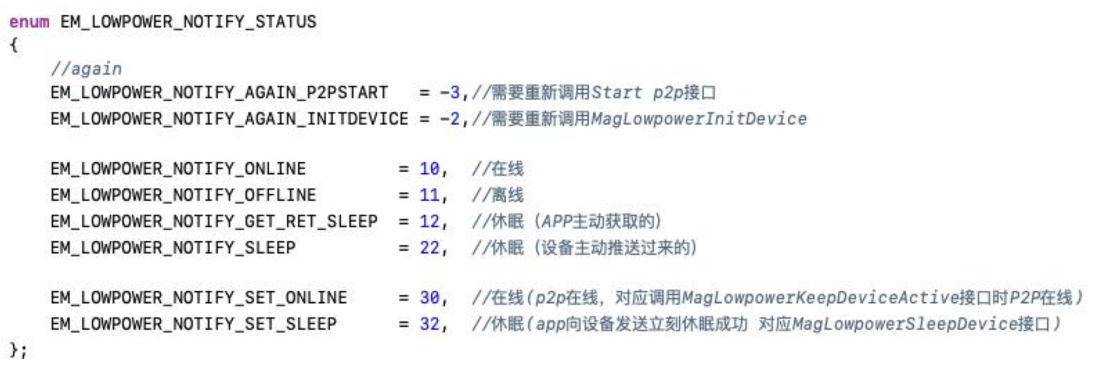
移除保活设备通讯

8.– (void)setLowpowerDeviceDelegate: (id <LowpowerDeviceProtocol>) delegate

设置低功耗设备代理，用于接收设备状态

二、接收设备状态

–(void) DeviceStateNotify:(NSString*)strUID state:(int) nState

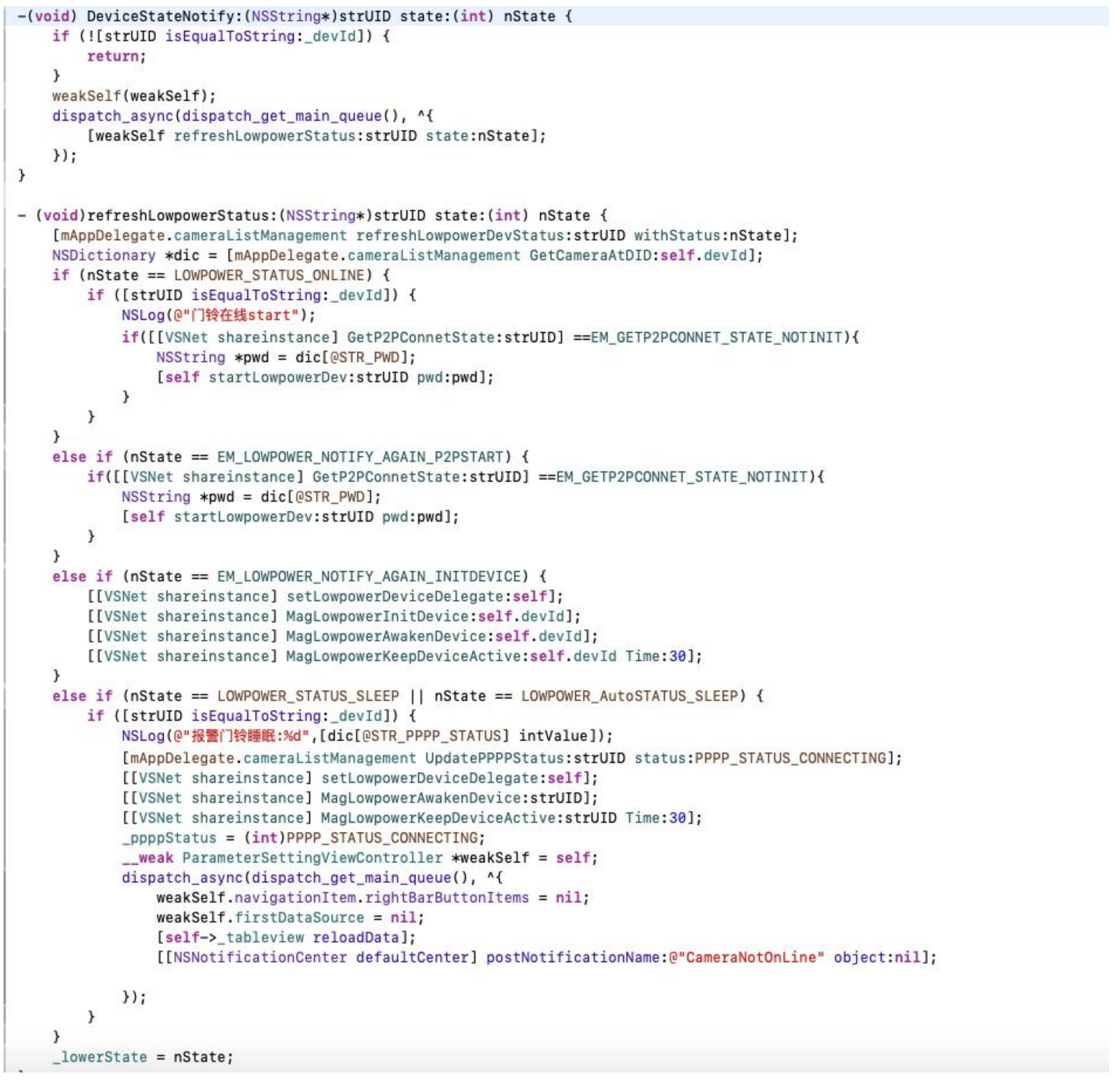


```
enum EM_LOWPOWER_NOTIFY_STATUS
{
    //again
    EM_LOWPOWER_NOTIFY_AGAIN_P2PSTART   = -3,//需要重新调用Start p2p接口
    EM_LOWPOWER_NOTIFY_AGAIN_INITDEVICE = -2,//需要重新调用MagLowpowerInitDevice

    EM_LOWPOWER_NOTIFY_ONLINE          = 10,  //在线
    EM_LOWPOWER_NOTIFY_OFFLINE         = 11,  //离线
    EM_LOWPOWER_NOTIFY_GET_RET_SLEEP   = 12,  //休眠（APP主动获取的）
    EM_LOWPOWER_NOTIFY_SLEEP           = 22,  //休眠（设备主动推送过来的）

    EM_LOWPOWER_NOTIFY_SET_ONLINE      = 30,  //在线(p2p在线，对应调用MagLowpowerKeepDeviceActive接口时P2P在线)
    EM_LOWPOWER_NOTIFY_SET_SLEEP       = 32,  //休眠(app向设备发送立刻休眠成功 对应MagLowpowerSleepDevice接口)
};
```

例如：



```
-(void) DeviceStateNotify:(NSString*)strUID state:(int) nState {
    if (![strUID isEqualToString:_devId]) {
        return;
    }
    weakSelf(weakSelf);
    dispatch_async(dispatch_get_main_queue(), ^{
        [weakSelf refreshLowpowerStatus:strUID state:nState];
    });
}

- (void)refreshLowpowerStatus:(NSString*)strUID state:(int) nState {
    [mAppDelegate.cameraListManagement refreshLowpowerDevStatus:strUID withStatus:nState];
    NSDictionary *dic = [mAppDelegate.cameraListManagement GetCameraAtDID:self.devId];
    if (nState == LOWPOWER_STATUS_ONLINE) {
        if ([strUID isEqualToString:_devId]) {
            NSLog(@"门铃在线start");
            if([[VSNet shareinstance] GetP2PConnetState:strUID] ==EM_GETP2PCONNET_STATE_NOTINIT){
                NSString *pwd = dic[@STR_PWD];
                [self startLowpowerDev:strUID pwd:pwd];
            }
        }
    }
    else if (nState == EM_LOWPOWER_NOTIFY_AGAIN_P2PSTART) {
        if([[VSNet shareinstance] GetP2PConnetState:strUID] ==EM_GETP2PCONNET_STATE_NOTINIT){
            NSString *pwd = dic[@STR_PWD];
            [self startLowpowerDev:strUID pwd:pwd];
        }
    }
    else if (nState == EM_LOWPOWER_NOTIFY_AGAIN_INITDEVICE) {
        [[VSNet shareinstance] setLowpowerDeviceDelegate:self];
        [[VSNet shareinstance] MagLowpowerInitDevice:self.devId];
        [[VSNet shareinstance] MagLowpowerAwakenDevice:self.devId];
        [[VSNet shareinstance] MagLowpowerKeepDeviceActive:self.devId Time:30];
    }
    else if (nState == LOWPOWER_STATUS_SLEEP || nState == LOWPOWER_AutoSTATUS_SLEEP) {
        if ([strUID isEqualToString:_devId]) {
            NSLog(@"报警门铃睡眠:%d",[dic[@STR_PPPP_STATUS] intValue]);
            [mAppDelegate.cameraListManagement UpdatePPPPStatus:strUID status:PPPP_STATUS_CONNECTING];
            [[VSNet shareinstance] setLowpowerDeviceDelegate:self];
            [[VSNet shareinstance] MagLowpowerAwakenDevice:strUID];
            [[VSNet shareinstance] MagLowpowerKeepDeviceActive:strUID Time:30];
            _ppppStatus = (int)PPPP_STATUS_CONNECTING;
            __weak ParameterSettingViewController *weakSelf = self;
            dispatch_async(dispatch_get_main_queue(), ^{
                weakSelf.navigationItem.rightBarButtonItems = nil;
                weakSelf.firstDataSource = nil;
                [self->_tableview reloadData];
                [[NSNotificationCenter defaultCenter] postNotificationName:@"CameraNotOnLine" object:nil];

            });
        }
    }
    _lowerState = nState;
```