

1.SDK集成指南

1.接入包作用

2.SDK接入

2.注意事项

1.BLE通讯

2.网络通讯

3.MFI通讯

3.接口说明

1.设备协议

2.BLE通讯

1.设备搜索

2.BLE设备

2.网络通讯

3.MFI通讯

1.SDK集成指南

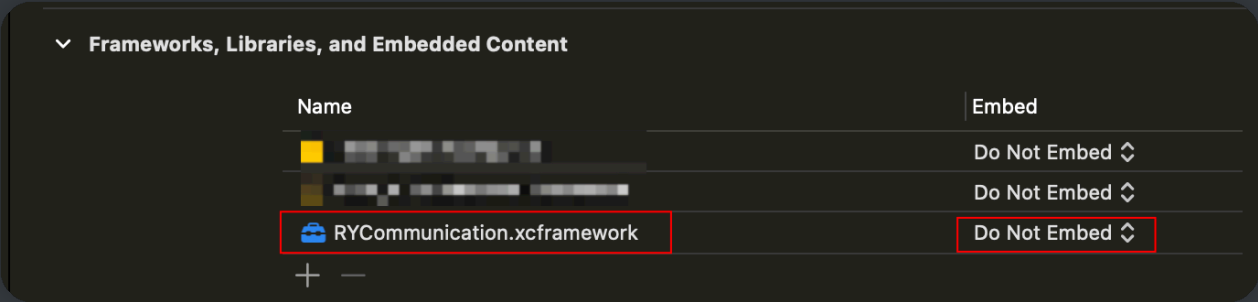
1.接入包作用

作为基础库，在iOS平台下为外设通讯提供基础服务。

2.SDK接入

1. 说明

RYCommunication属于静态库，所以接入项目时不需要将库绑定至应用

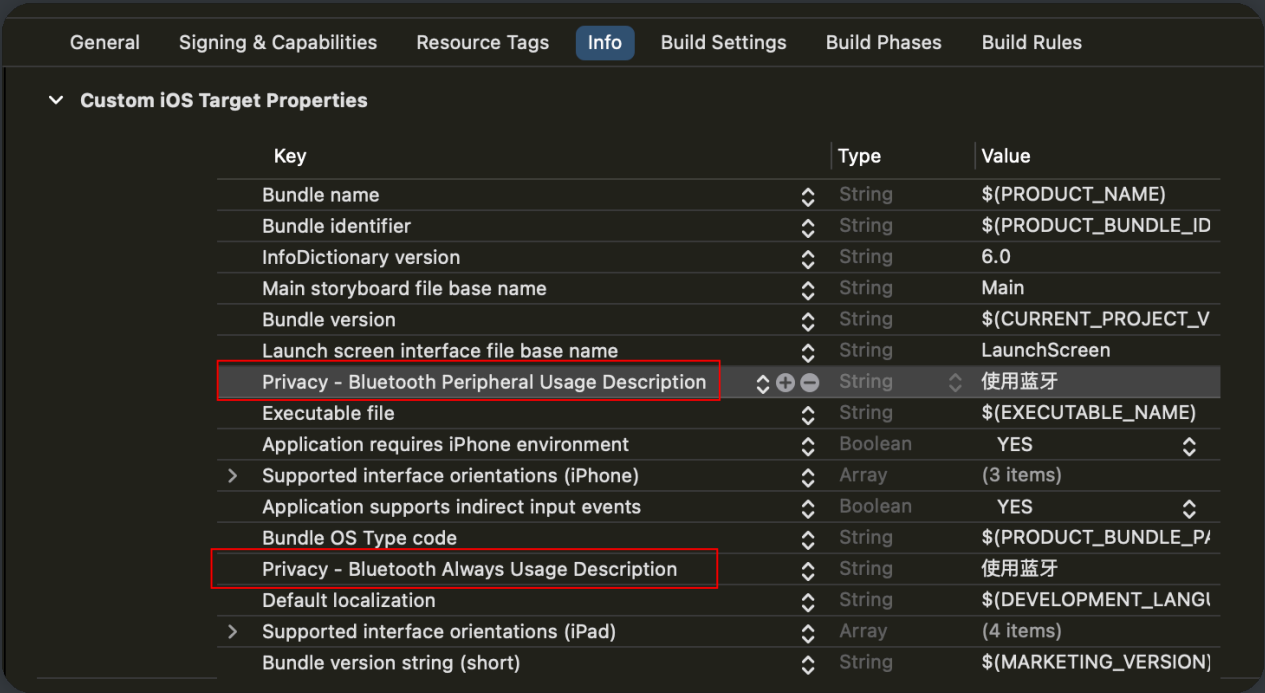


2.注意事项

1.BLE通讯

1. Info.plist配置

使用BLE设备，需要在配置文件添加蓝牙使用描述



2.网络通讯

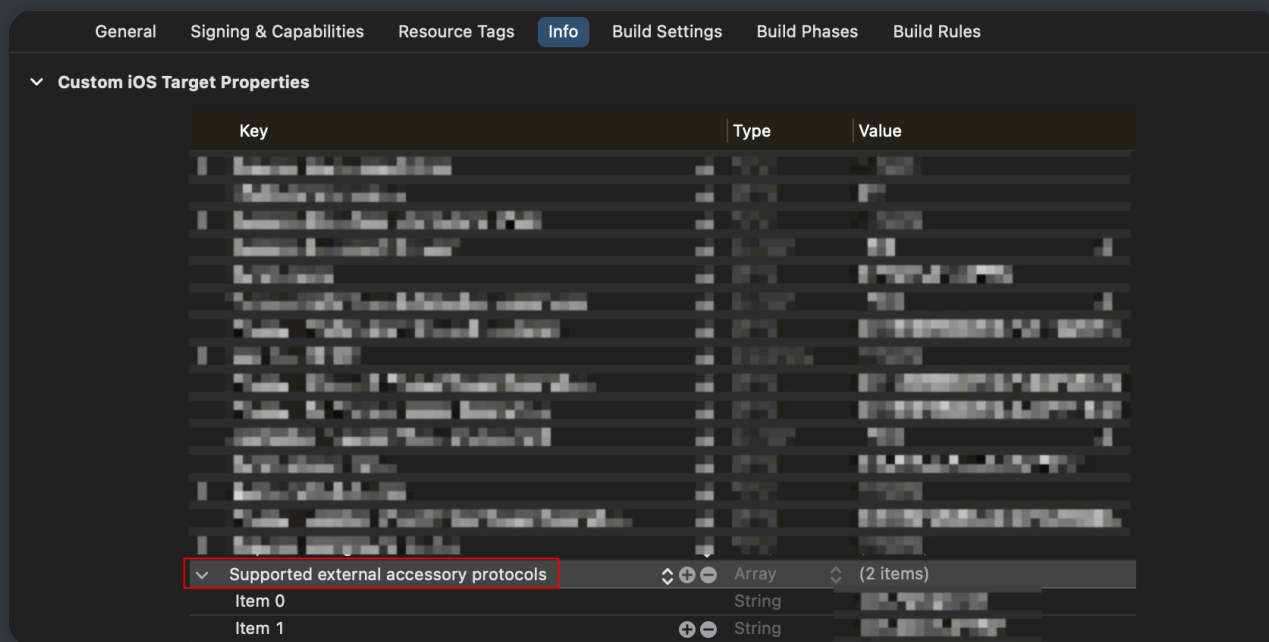
1. 应用权限

应用需要使用Socket通讯，需要保证用户开启网络权限，并允许使用无线局域网。调用苹果原生网络请求框架进行网络请求可自动触发权限申请

3.MFI通讯

1. Info.plist配置

使用MFI设备，需要在配置文件声明应用支持的Supported external accessory protocols



3.接口说明

1.设备协议

RYAccessory作为设备协议，规范设备对象的基本操作

1. 连接打印机

```
/**
 *  \~chinese
 *
 *  连接
 *
 *  @param successBlock 连接成功回调
 *  @param failBlock 连接失败回调
 *
 *  \~english
 *
 *  connect
```

```

*
* @param successBlock called when connect success
* @param failBlock called when connect fail
*
*/
- (void)connect:(void(^ _Nullable)(void))successBlock fail:(void(^ _Nullable)(NSError *))failBlock;

```

2. 断开连接

```

/**
 * \~chinese
 *
 * 断开连接
 *
 * \~english
 *
 * disconnect
 *
 */
- (void)disconnect;

```

3. 监听连接断开

```

/*!
 * \~chinese
 *
 * 连接关闭回调
 *
 * \~english
 *
 * called when connect is closed
 *
 */
@property (nonatomic, copy) void (^ _Nullable closedBlock)(NSError * _Nullable);

```

4. 写入数据

```
/**
 *  \~chinese
 *
 *  写入数据
 *
 *  @param data 要写入的数据
 *  @param block 写入进度回调
 *
 *  \~english
 *
 *  write data
 *
 *  @param data data will write
 *  @param block called when write process has updated
 *
 */
- (void)write:(NSData *)data progress:(void (^)(_Nullable)NSProgress
*))block;
```

5. 设备名称

```
@property (nonatomic, copy) NSString *name;
```

6. 数据解析器

```
@property (nonatomic, strong) id<RYDataResolver> _Nullable
resolver;
```

2.BLE通讯

1.设备搜索

RYCentralManager作为蓝牙中心，主要负责BLE设备的发现

1. 对象构造

```
/// 获取蓝牙中心单例  
+ (instancetype)share;
```

2. 蓝牙是否开启

```
/// 蓝牙是否开启  
@property (nonatomic,readonly, assign) BOOL isBlePowerOn;
```

3. 开始搜索

```
/// 开始搜索，返回搜索任务标识  
/// - Parameters:  
///   - timeout: 超时时间  
///   - discoverBlock: 搜索设备回调  
///   - powerOffBlock: 蓝牙关闭回调  
///   - scanTimeoutBlock: 搜索超时回调  
///   - printerFilter: 设备过滤  
- (NSString *)startScan:(NSTimeInterval)timeout  
    discoverBlock:(void(^ _Nullable)(RYBleAccessory  
*device))discoverBlock  
    powerOffBlock:(void(^ _Nullable)(void))powerOffBlock  
    scanTimeoutBlock:(void(^ _Nullable)(void))scanTimeoutBlock  
    printerFilter:(BOOL (^ _Nullable)(CBPeripheral *,  
NSNumber *, NSDictionary<NSString *,id> *))printerFilter;
```

4. 停止搜索

```
/// 停止搜索  
/// - Parameter identity: 搜索任务标识  
- (void)stopScan:(NSString *)identity;
```

2.BLE设备

RYCentralManager搜索后可以直接获取BLE设备对象，可以直接操作该对象进行通讯操作

2.网络通讯

1. 网络设备

RYSocketAccessory对象可以直接使用ip和端口进行创建

```
/**
 *  \~chinese
 *
 *  创建socket设备对象
 *
 *  @param ip ip
 *  @param port port
 *
 *  \~english
 *
 *  Create socket accessory object
 *
 *  @param ip ip
 *  @param port port
 *
 */
- (instancetype)initWith:(NSString *)ip port:(NSInteger)port;
```

3.MFI通讯

MFI设备原生对象EAAccessory需要通过ExternalAccessory框架获取，更多信息见ExternalAccessory官方文档

1. 手机搜索连接MFI设备

ExternalAccessory框架，EAAccessoryManager提供接口，用于搜索周边的MFI，并连接至手机。注意是连接到手机，此时应用并没有连接设备

```
- (void)showBluetoothAccessoryPickerWithNameFilter:(NSPredicate *)predicate completion:(EABluetoothAccessoryPickerCompletion)completion;
```

2. 应用获取连接到手机的MFI设备

ExternalAccessory框架，EAAccessoryManager提供接口

```
@property(n nonatomic, readonly) NSArray<EAAccessory *>
    *connectedAccessories;
```

3. MFI设备

```
/**
 *  \~chinese
 *
 *  创建MFI设备对象
 *  @param accessory 系统外设对象
 *  @param protocolString 设备协议
 *
 *  \~english
 *
 *  Create MFI RYExternalAccessory object
 *
 *  @param accessory EAAccessory
 *  @param protocolString MFI protocol
 *
 */
- (instancetype)initWithAccessory:(EAAccessory *)accessory
forProtocol:(NSString *)protocolString;
```

