**OpenHarmony竞赛训练营**

**赛题：\_\_\_\_\_题目九：病床巡检终端\_\_\_\_\_**

|  |  |
| --- | --- |
| 学校名称： | 华中科技大学 |
| 团队名称： | 名称暂定团队 |
|  |  |
| 队长： | 刘浩毅 |
| 队员1： | 赵建 |

OpenHarmony竞赛训练营组委会

2023年10月 23日

目 录

[1 简介](#_Toc149066666)

[1.1 背景 3](#_Toc149066667)

[1.2 目的 3](#_Toc149066668)

[1.2.1赛题目标: 3](#_Toc149066669)

[1.2.2技术挑战： 4](#_Toc149066670)

[1.2.3需求分析： 4](#_Toc149066671)

[1.2.4文档目的： 4](#_Toc149066672)

[2 设计描述 4](#_Toc149066673)

[2.1 总体设计 4](#_Toc149066674)

[2.2 实现思路 4](#_Toc149066675)

[2.3 系统结构 5](#_Toc149066676)

[2.3.1 模块划分 5](#_Toc149066677)

[2.3.2 系统架构说明 6](#_Toc149066678)

[2.3.3 文件结构 8](#_Toc149066679)

[2.4 模块功能描述 9](#_Toc149066680)

[2.4.1 页面显示功能描述 9](#_Toc149066681)

[2.4.2 功能模块描述 10](#_Toc149066682)

[2.4.3 其它模块功能描述 11](#_Toc149066683)

[2.5 业务/实现流程说明 12](#_Toc149066684)

[2.6 接口描述 12](#_Toc149066685)

[2.6.1 调用接口 12](#_Toc149066686)

[2.6.2 提供接口 13](#_Toc149066687)

[2.7 UI设计 13](#_Toc149066688)

[3 其他](#_Toc149066689)

[3.1 成员分工 13](#_Toc149066690)

[3.2 困难与思考 13](#_Toc149066691)

[3.3 参考 14](#_Toc149066692)

# 简介

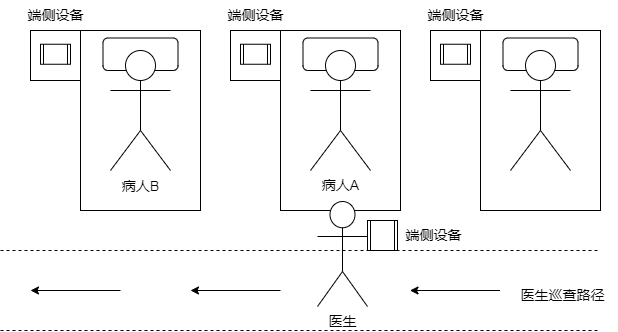
## 背景

在OpenHarmony的行业生态赛题九：病床巡检终端中介绍了这样的一种背景：

数字化技术正在蓬勃发展，医院的智慧化转型正在加速。物联终端接入、信息化诊疗的普及都在加速医院智慧化升级的步伐。心率、血氧、体温等基础病人数据是医生和护理人员衡量病人状态的关键指标，如何更快速更方便的获取病床上病人的这些数据，也成为智慧化医院所考虑的一项重要工作。

它描述了这样的问题：

如下图所示病房内有多张病床，每张病床配备有端侧设备收集病人的心率、血氧、体温等生理数据。医生手持端侧设备进行病房查房，端侧设备间可通过蓝牙进行连接。当其走到病人A床尾时，由于A端侧设备距离最近，自动通过病人A的端侧设备获取该病人的生理数据，进行检视判断。当其走到病人B床尾时，通过病人B的端侧设备获取该病人的生理数据。现用两台开发板设备模拟不同病人的端侧设备，用另一台开发板模拟医生手持设备实现该功能。



本文档为病床巡检终端的开发设计文档，致力于完成病床巡检终端的整体设计并让其它开发者能更快的上手理解我们的工作。

## 目的

### 赛题目标:

1.实现设备间分布式组网，并且根据距离与最近设备建立连接

2.实现数据同步，获取到最近设备的相应模拟数据

3.界面可视化，实现医生查房系统

### 技术挑战：

1.通过分布式软总线进行设备配对组网

2.根据蓝牙距离判断需同步设备并进行数据同步

### 需求分析：

1. **设备间分布式组网**：病床上的端侧设备需要能够通过分布式软总线进行配对组网。这意味着设备需要支持分布式通信，能够发现和连接其他设备。这样，医生手持设备就可以与最近的病床端侧设备建立连接。

2. **蓝牙距离判断**：医生手持设备在巡检过程中，需要根据蓝牙信号的强度判断与最近的病床端侧设备的距离。可以通过比较不同设备之间的蓝牙信号强度来确定最近的设备。一旦确定最近的设备，医生手持设备需要与该设备进行数据同步，获取病人的心率、血氧、体温等生理数据。

3. **数据同步**：医生手持设备需要与最近的病床端侧设备建立数据同步。一旦连接建立，医生手持设备可以请求病人的生理数据，并将其显示在界面上供医生查看。数据同步需要确保数据的准确性和及时性。

4. **界面可视化**：为了方便医生查房，需要实现一个界面可视化的医生查房系统。该系统可以显示病人的基础生理数据，如心率、血氧、体温等。医生可以通过界面进行操作和查看数据，以便做出准确的判断和决策。

### 文档目的：

本文档希望给项目提供清晰、准确、完整的项目开发信息和指导，帮助项目团队成员以及其它开发者理解项目的需求、设计和实现，以及项目的目标和计划，特别是该怎么样实现以上的项目目标以及满足项目需求。

# 设计描述

## 总体设计

根据以上需求分析，我们对应的设计了应用的4种功能来对应的满足以上的4点需求：

1. **分布式设备管理功能：**提供分布式设备管理能力，能实现分布式组网。
2. **蓝牙功能：**提供蓝牙的BLE扫描以及广播功能实现距离判断能力。
3. **数据同步功能：**提供应用多设备间的分布式数据对象协同能力。
4. **Ui界面功能：**提供给用户界面可视化以及交互能力，并且因为医生端和病人端的显而易见的不同，要将其进行拆分。

除此之外，还需要一些功能来辅助整个应用的调试，使用，如打印日志的功能，以及申请权限的功能。

## 实现思路

对于主要的4种功能，我们结合OpenHarmony对应的模块，实现了它们的功能：

1. 设备间分布式组网：使用了@ohos.distributedHardware.deviceManager模块解决，该模块提供分布式设备管理能力。系统应用可调用接口实现如下功能：

* 注册和解除注册设备上下线变化监听
* 发现周边不可信设备
* 认证和取消认证设备
* 查询可信设备列表
* 查询本地设备信息，包括设备名称，设备类型和设备标识
* 发布设备发现

2. 蓝牙距离判断：使用了@ohos.bluetoothManager模块解决，该模块提供了基础的传统蓝牙能力以及BLE的扫描、广播等功能。

3. 数据同步：使用了@ohos.data.distributedDataObject模块解决，该模块提供管理基本数据对象的相关能力，包括创建、查询、删除、修改、订阅等；同时支持相同应用多设备间的分布式数据对象协同能力。

4. 界面可视化：使用ArkTS语言开发，通过UI的配置将应用中分为病床端和医生端，病床端显示当前监察数据，医生端进行设备管理以及同步数据的显示，实现了界面可视，便于使用。

## 系统结构

### 模块划分

因为应用的相对简单，我们主要考虑了将视图与控制器结合，并另外设置模型来实现我们的功能。对应的从的模块划分来说我们设置了视图模块，模型模块，其它模块。细分来说，视图模块包括：

1. 首页：提供了APP应用的开始界面。
2. 病床端主页：提供了病床端的主页，并显示了当前病床端的检测数据。
3. 医生端主页：提供了医生端的主页；使用了标题栏组件，可以通过这个组件来打开设备管理界面；并且可以显示当前同步设备的数据。
4. 设备管理界面组件：打开设备管理界面，并根据用户操作执行对应的功能。

模型模块包括：

1.分布式设备管理模块：该模块提供分布式设备管理能力，提供了远程设备的发现，监听状态改变，认证和取消认证等功能。

2.蓝牙模块：该模块提供蓝牙的BLE扫描以及广播功能。

3.数据同步模块：该模块提供应用多设备间的分布式数据对象协同功能。

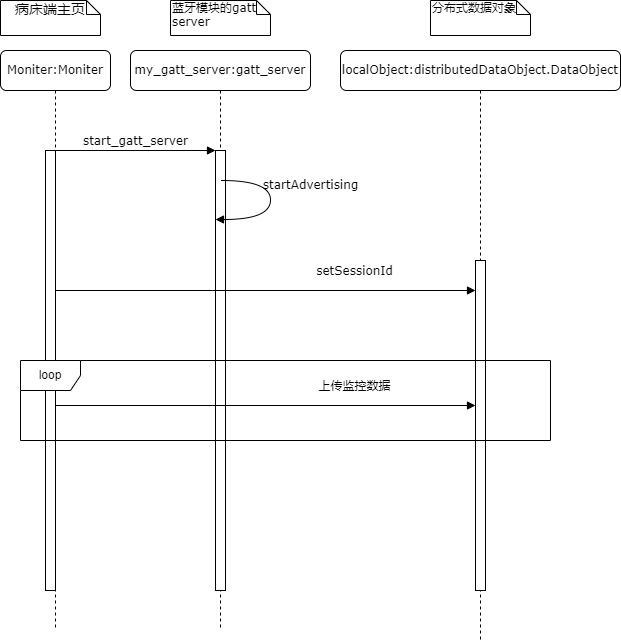
除此之外，另设计一些小的模块来辅助整个应用的调试，使用：

1. 日志模块：该模块提供打印日志到控制台中的能力，方便调试。
2. 常量模块：该模块保存一些常量。

### 系统架构说明

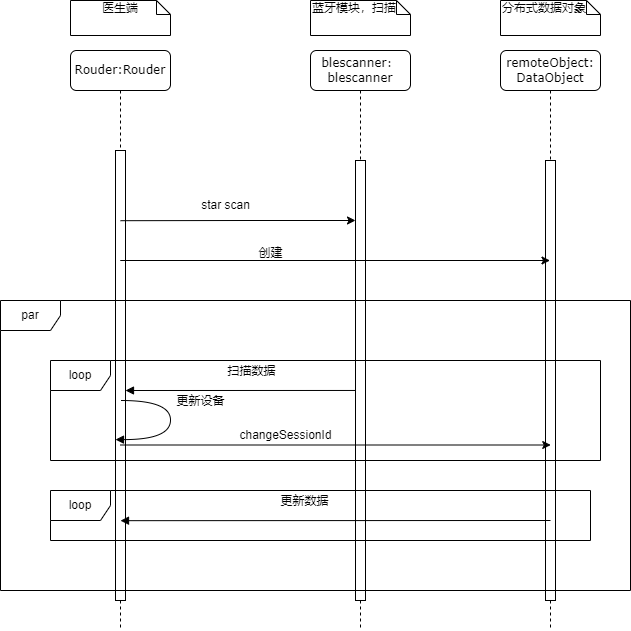
在APP进入的首页中，并不涉及太多功能，除第一次使用时向用户获取权限的功能外，只能够转入医生端或病床端。

在**病床端**，涉及的模块如下图，其中可能涉及的常量模块、日志模块并未画出。病床端，主要进行蓝牙模块gattserver的启动，以及及时进行数据的更新。



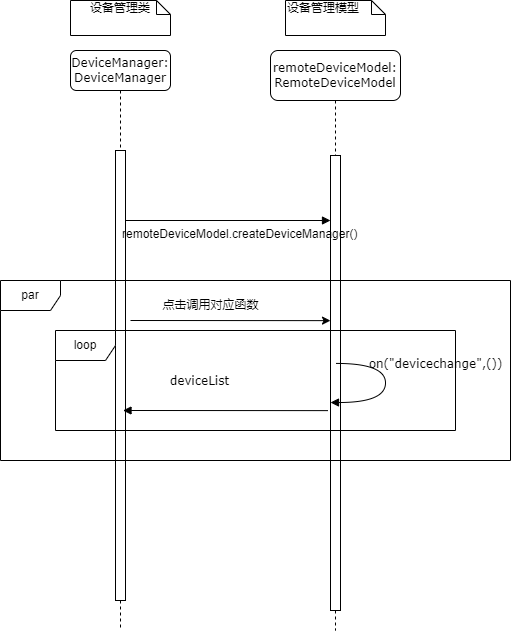
病床端uml图

在**医生端**，涉及的模块如下图。医生端提供了向**设备管理页面的跳转**按钮。还进行了蓝牙的扫描，当信号强度最高的可信设备改变时，会更换设备，为了避免设备频繁切换，在这里我们还实现了防抖动设计。另外数据同步以及显示，也是病床端不可能缺少的功能。特别的是，我们为医生端收到的心率绘制了心率曲线。



医生端uml图

在**设备管理**页面，涉及的模块主要为设备管理模型，该页面实现了发现设备列表的显示，以及已认证设备列表的显示。除此之外，通过该页面上的交互，还可以调用设备管理模型中对应函数，实现发布本机，停止发布本机，进行设备发现，停止设备发现，进行设备认证，取消设备认证的操作。



设备管理端uml图

### 文件结构

如下为我们设计的病床巡检终端文件目录结构

1. ─main
2. │  module.json5  //配置文件
3. │
4. ├─ets
5. │  ├─common
6. │  │      ListDeviceView.ets    //单一设备渲染
7. │  │      TitleBarComponent.ets   //包含设备管理页面入口的标题栏组件
8. │  │
9. │  ├─entryability
10. │  │      EntryAbility.ts  //程序入口类
11. │  │
12. │  ├─model
13. │  │      blescanner.ts    //ble扫描类
14. │  │      gattserver.ets   //gatt server类
15. │ │ data\_collector.ets //数据接口
16. │  │      RemoteDeviceModel.ets  //分布式设备管理类
17. │  │
18. │  ├─pages
19. │  │      DeviceManager.ets  //设备管理页
20. │  │      Monitor.ets  //病床端主页
21. │  │      Rounder.ets  //医生端主页
22. │  │      welcome.ets  //首页
23. │  │
24. │  └─util
25. │          constant.ets //常量
26. │          Logger.ts    //日志类
27. │
28. └─resources
29. ├─base
30. │  ├─element
31. │  ├─media
32. │  │
33. │  └─profile
34. │          main\_pages.json
35. ├─en\_US
36. ├─rawfile
37. └─zh\_CN

## 模块功能描述

### 页面显示功能描述

* 首页：welcome.ets

|  |
| --- |
| Welcome |
| aboutToAppear()//进行权限的验证 |
| build()//显示UI，构建两个按钮，分别进入病床端和医生端主页 |

* 病床端主页：Monitor.ets

|  |
| --- |
| Moniter |
| localObject:distributedDataObject.DataObject//分布式数据对象，进行数据同步 |
| my\_gatt\_server:gatt\_server//gattserver进行BLE蓝牙广播 |
| aboutToAppear()//启动gattserver，并开启localObject中数据更新 |
| build()//显示当前监测数据， |
| aboutToDisappear()//关闭广播 |

* 医生端主页：Rounder.ets；

|  |
| --- |
| Rouder |
| remoteDeviceModel: RemoteDeviceModel//分布式设备管理类 |
| ble\_scanner:ble\_scanner//ble扫描 |
| remoteObject//分布式数据对象 |
| aboutToAppear()//进行蓝牙扫描，并对扫描事件进行监听，在信号最强的设备发生改变时更改同步数据的设备，同时实现了防抖动的设计 |
| build()//显示UI，使用了标题栏组件，将remoteDeviceModel作为参数传入，并进行了当前同步数据的显示 |
| onDestroy()//进行了remoteDeviceModel的各监听的停止 |

* 设备管理页：DeviceManager.ets；

|  |
| --- |
| DeviceManager |
| localDevice: deviceManager.DeviceInfo //当前设备信息  trustedDeviceList: Array<deviceManager.DeviceInfo>//已认证设备列表 |
| discoverDeviceList: Array<deviceManager.DeviceInfo>//发现的设备列表 |
| isAllowDiscovered: Boolean//该设备是否允许被发现 |
| aboutToAppear()//调用remoteDeviceModel.createDeviceManager()进行设备管理模块的创建 |
| build()//显示设备管理页面，渲染图标，实现了当前设备是否允许被发现的选择，以及发现设备列表的显示和已信任设备列表的显示 |

### 功能模块描述

* 分布式设备管理模块：RemoteDeviceModel.ets

导入了分布式设备管理库'@ohos.distributedHardware.deviceManager';

在RemoteDeviceModel中的函数会进行回调函数的传入与调用，便于及时更新数据。

|  |
| --- |
| RemoteDeviceModel |
| deviceList: Array<deviceManager.DeviceInfo>//认证设备列表 |
| discoverList: Array<deviceManager.DeviceInfo>//发现设备列表 |
| deviceManager: deviceManager.DeviceManager//设备管理器 |
| createDeviceManager(): Promise<void>//进行deviceManager的创建  getLocalDeviceInfo(): void//获得当前设备信息  getTrustedDeviceList()//获得已认证设备列表  publishDeviceDiscovery(): void//进行当前设备的发布  unPublishDeviceDiscovery(): void//停止当前设备的发布  registerDeviceStateListener(): void//进行设备状态改变的监听，并更新对应的列表信息  startDeviceDiscovery(): void//开始设备发现  stopDeviceDiscovery(): void//停止设备发现  unregisterDeviceListCallback(): void//停止对设备状态改变的监听  authenticateDevice(device): void//对传入的设备进行设备认证  unAuthenticateDevice(device): void//对传入的设备取消设备认证 |

* 分布式数据对象模块：在Monitor.ets,Rounder.ets都有使用

导入了分布式数据对象库'@ohos.data.distributedDataObject';

创建了distributedDataObject.DataObject对象，并使用了setSessionId()方法建立对应的连接，以及on(“change”,())进行数据更新的监听以及处理。

* 蓝牙模块：gattserver.ets,blescanner.ts

导入了蓝牙管理库'@ohos.bluetoothManager';

|  |
| --- |
| gatt\_server |
| gattServer: bluetoothManager.GattServer//gattServer类 |
| server\_status:Boolean |
| start\_gatt\_server()//开启gatt server |
| stop\_gatt\_server()//关闭gatt server |

|  |
| --- |
| ble\_scanner |
| onReceiveEvent()//接受到数据时发生事件 |
| start\_scan()//开启扫描 |

具体蓝牙测距的实现：

1.设备配置：

* 病床设备端将自身设置为蓝牙低功耗（BLE）设备，并启用GATT服务器功能。
* 医生设备端作为中央设备，启用蓝牙扫描功能，并配置为GATT客户端。

2.广播数据包：

* 病床设备端使用GATT服务器广播数据包，其中包含以下字段：
* 制造商标识（Manufacturer ID）：用于识别设备制造商，此项目使用的设备制造商id为0x4444，与serviceUUID一起用来协助过滤出病床设备。
* 服务标识（Service ID）：用于标识设备提供的特定服务。
* 添加服务，此项目使用的serviceUUID为 00001889-0000-1000-8000-00805f9b34fb，即0x1889，对应的serviceValue表示设备的id。

3.医生设备端扫描与过滤：

* 医生设备端启动蓝牙扫描功能，以搜索周围的BLE设备。
* 扫描到的设备广播报文中，医生设备端过滤出具有匹配的制造商标识和服务标识的设备。

4.获取、解析广播报文，确定设备距离：

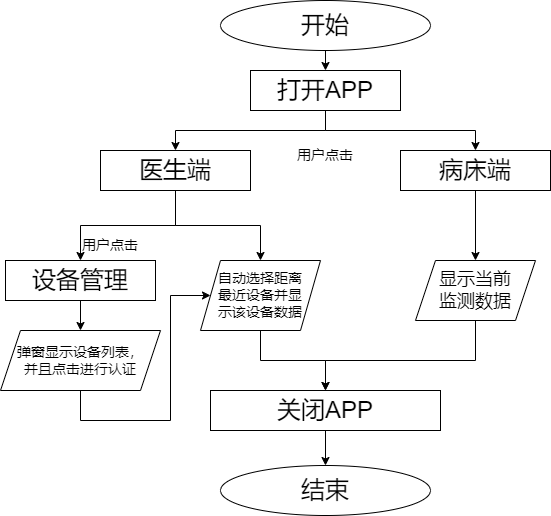
* 医生设备端通过蓝牙扫描获取到的广播报文中，定位到病床设备端的广播数据包。
* 医生设备端解析病床设备端的广播数据包，提取出相关字段。
* 使用已提取的设备标识符字段，医生设备端可以确定病床设备端与自身的距离。 基于信号强度指示（RSSI）值可以进行距离的确定。

### 其它模块功能描述

* 日志模块：Logger.ts；进行日志的打印。
* 常量模块：constant.ets；保存一些常量。

## 业务/实现流程说明

应用的使用流程如下图所示：



业务流程图

## 接口描述

### 调用接口

* 分布式设备管理模块

导入了分布式设备管理库@ohos.distributedHardware.deviceManager；

主要使用了deviceManager下的createDeviceManager()方法,以及DeviceManager类下的startDeviceDiscovery(), stopDeviceDiscovery(), getTrustedDeviceListSync(), on('deviceStateChange, ()), on('deviceFound', ()), on('discoverFail', ()), on('serviceDie', ()), authenticateDevice()等方法

* 分布式数据对象模块

导入了分布式数据对象库@ohos.data.distributedDataObject';

使用了distributedDataObject的create()方法，以及DataObject类下的setSessionId(), on('change') 等方法

* 蓝牙模块

导入了蓝牙管理库@ohos.bluetoothManager;

在blescanner.ts中主要使用了bluetoothManager下BLE类的startBLEScan(), on("BLEDeviceFind", ())方法，实现了BLE蓝牙的扫描，以及扫描事件发生的监听。

在gattserver.ets中主要使用了bluetoothManager下BLE类的createGattServer()创建了GattServer类对象,并使用了bluetoothManager下gattServer类的startAdvertising(), stopAdvertising()方法实现了广播。

### 提供接口

数据采集模块接口采用依赖注入模式，位于data\_collector.ets中，只需根据不同设备实现不同的collector即可。目前实现了模拟采集数据的collector，即随机生成数据的rand\_gen()函数，该函数随机生成数据并返回数据，在Moniter.ets中该函数被调用。对应的可以在data\_collector.ets中编写函数返回收集到的数据，在Moniter.ets中调用。数据以key-value对返回，包括heart\_beat（心率）, oximetry（血氧）, temperature（温度）, icon\_color（图标颜色）等。

## UI设计

使用ArkTS语言开发，响应式数据刷新，使用框架能力进行多语言适配。将应用分为病床端和医生端，病床端显示当前监察数据，医生端进行设备管理以及同步数据的显示。

控件方面主要使用了按钮组件实现页面的跳转，事件的触发；使用弹窗控件实现了设备管理弹窗的显示与设备的操作；使用列表控件，按序渲染数据等；使用图片控件实现了图片的加载显示，以及颜色改变。

而布局方面主要使用了线性布局，在行和列中分配图标的位置。

# 其他

## 成员分工

* 刘浩毅：担任队长，对应用进行了整体设计，并负责了主体代码的编写，特别是蓝牙模块的资料搜索以及代码编写，并进行了UI的绘制，除此外还进行了开发板的调试使用以及一些资料的整理。
* 赵建：负责了分布式设备管理部分的代码编写，以及文档和ppt的制作，并进行了开发板的调试使用以及相关资料的查找整理。

## 困难与思考

在开发过程我们已经遇到并解决了许多问题，以下为我们遇到的一些困难以及产生的思考：

1. 资料缺失：资料比较缺失是OpenHarmoy开发目前最大的问题，即使已经有相当多的OpenHarmony开发者进行了分享，参考还是较少。在开发过程中常遇到的一个问题是，遇到一个问题，进行搜索，但完全没有直接相关的解答，常常是漫游网络后，找到在别的文档中偶然提及的一句问题的解决方案。这样的情况下，虽然官方文档较为完善，但往往我们并不能直接的找到问题的解决方案。但是社区解答问题比较及时，一定程度上弥补了问题。
2. 团队合作：在团队开发过程中，我们也产生了一些失误。表现在双方的同步不够及时，往往是一个人踩过的坑另一个人又接着踩过去，造成了重复的损耗。团队成员应编写开发日志，将遇到的问题都记录下来，并及时交流更新解决。
3. 硬件问题：开发板有时会遇到一些莫名其妙的问题，如usb调试只有在传输文件时立刻掉线，后来才发现可能是usb口过热导致的，等等。这种问题较为棘手，有时重启就会正常，有时甚至需要换设备才能解决，难以复现，难以定位追踪。

## 参考

[OpenHarmony 3.2 Release应用开发](https://docs.openharmony.cn/pages/v3.2/zh-cn/application-dev/application-dev-guide.md/)文档

[Laval社区 (csdn.net)](https://laval.csdn.net/)

[zh-cn/application-dev/reference/apis/js-apis-device-manager.md · OpenHarmony/docs - Gitee.com](https://gitee.com/openharmony/docs/blob/master/zh-cn/application-dev/reference/apis/js-apis-device-manager.md#publishinfo9)

[full-SDK替换指南 | 中俊为 | OpenHarmony文档中心](http://open.weharmonyos.com/zh-cn/application-dev/quick-start/full-sdk-switch-guide.html)

[访问控制授权申请-访问控制-安全-开发-HarmonyOS应用开发](https://developer.harmonyos.com/cn/docs/documentation/doc-guides-V3/accesstoken-guidelines-0000001493744016-V3)

[应用权限列表-访问控制-安全-开发-HarmonyOS应用开发](https://developer.harmonyos.com/cn/docs/documentation/doc-guides-V3/permission-list-0000001544464017-V3#ZH-CN_TOPIC_0000001523648786__ohospermissiondistributed_datasync)

[【小源笔记】第六期 | 解决应用申请权限失败导致安装报错问题-开源基础软件社区-51CTO.COM](https://ost.51cto.com/posts/19833)

[harmonyos - 「鸿蒙学习笔记」Stage模型--UIAbility - 个人文章 - SegmentFault 思否](https://segmentfault.com/a/1190000044191975)