|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | |  |
| **OpenHarmony 产品兼容性规范 3.0** | | |  |
| **OpenHarmony 产品兼容性规范 3.0** | | |
| **文档版本** | **01** | |
| **发布日期** | **2021-10-20** | |
|  | | | | |
|  | OpenHarmony | |  |  |

|  |
| --- |
| 版权所有 © OpenHarmony项目贡献者 2021。 保留一切权利。  本材料所载内容受著作权法的保护，著作权由OpenHarmony项目贡献者或其许可人拥有，但注明引用其他方的内容除外。未经OpenHarmony项目贡献者或其许可人事先书面许可，任何人不得将本材料中的任何内容以任何方式进行复制、经销、翻印、播放、以超级链路连接或传送、存储于信息检索系统或者其他任何商业目的的使用。  商标声明  为开放原子开源基金会的商标（非详尽清单），未经开放原子开源基金会书面事先明示许可，任何第三方不得以任何形式使用。  注意  OpenHarmony项目贡献者会不定期对本文档的内容进行更新。  本文档仅作为使用指导，文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。 |

目 录

[1 概述（发布到网站时独立于PCS文档之外） 1](#_Toc256000000)

[1.1 OpenHarmony系统简介 1](#_Toc256000001)

[1.2 OpenHarmony兼容性目标 2](#_Toc256000002)

[1.3 OpenHarmony产品兼容性规范文档说明 2](#_Toc256000003)

[2 介绍 4](#_Toc256000004)

[3 基础系统兼容性 6](#_Toc256000005)

[3.1 轻量系统（mini system）兼容性 6](#_Toc256000006)

[3.1.1 硬件 6](#_Toc256000007)

[3.1.1.1 CPU 6](#_Toc256000008)

[3.1.1.2 图形显示 6](#_Toc256000009)

[3.1.1.3 内存和存储 6](#_Toc256000010)

[3.1.1.4 通信 7](#_Toc256000011)

[3.1.1.5 摄像头 7](#_Toc256000012)

[3.1.1.6 音频 7](#_Toc256000013)

[3.1.1.7 USB 7](#_Toc256000014)

[3.1.1.8 加解密和安全隔离运行环境 7](#_Toc256000015)

[3.1.2 软件 7](#_Toc256000016)

[3.1.2.1 最小集API兼容性 7](#_Toc256000017)

[3.1.2.2 运行时兼容性 10](#_Toc256000018)

[3.1.2.3 应用兼容性 10](#_Toc256000019)

[3.1.2.4 应用包格式 10](#_Toc256000020)

[3.1.2.5 JS UI框架 11](#_Toc256000021)

[3.1.3 分布式 11](#_Toc256000022)

[3.1.4 性能和功耗 11](#_Toc256000023)

[3.1.5 安全 11](#_Toc256000024)

[3.1.6 系统和软件升级 11](#_Toc256000025)

[3.1.7 开发工具和开发选项 11](#_Toc256000026)

[3.1.8 DFX（Design for X） 11](#_Toc256000027)

[3.2 小型系统（small system）兼容性 11](#_Toc256000028)

[3.2.1 硬件 11](#_Toc256000029)

[3.2.1.1 CPU 11](#_Toc256000030)

[3.2.1.2 图形显示 12](#_Toc256000031)

[3.2.1.3 内存和存储 12](#_Toc256000032)

[3.2.1.4 通信 12](#_Toc256000033)

[3.2.1.5 摄像头 12](#_Toc256000034)

[3.2.1.6 音频 12](#_Toc256000035)

[3.2.1.7 USB 12](#_Toc256000036)

[3.2.1.8 加解密和安全隔离运行环境 12](#_Toc256000037)

[3.2.2 软件 12](#_Toc256000038)

[3.2.2.1 最小集API兼容性 12](#_Toc256000039)

[3.2.2.2 运行时兼容性 16](#_Toc256000040)

[3.2.2.3 应用兼容性 17](#_Toc256000041)

[3.2.2.4 应用包格式 17](#_Toc256000042)

[3.2.2.5 JS UI框架 17](#_Toc256000043)

[3.2.3 分布式 17](#_Toc256000044)

[3.2.4 性能和功耗 17](#_Toc256000045)

[3.2.5 安全 17](#_Toc256000046)

[3.2.6 系统和软件升级 17](#_Toc256000047)

[3.2.7 开发工具和开发选项 17](#_Toc256000048)

[3.2.8 DFX（Design for X） 17](#_Toc256000049)

[3.3 标准系统（standard system）兼容性 18](#_Toc256000050)

[3.3.1 硬件 18](#_Toc256000051)

[3.3.1.1 CPU 18](#_Toc256000052)

[3.3.1.2 图形显示 18](#_Toc256000053)

[3.3.1.3 内存和存储 18](#_Toc256000054)

[3.3.1.4 通信 18](#_Toc256000055)

[3.3.1.5 摄像头 18](#_Toc256000056)

[3.3.1.6 音频 18](#_Toc256000057)

[3.3.1.7 USB 18](#_Toc256000058)

[3.3.1.8 加解密和安全隔离运行环境 18](#_Toc256000059)

[3.3.2 软件 19](#_Toc256000060)

[3.3.2.1 最小集API兼容性 19](#_Toc256000061)

[3.3.2.2 运行时兼容性 23](#_Toc256000062)

[3.3.2.3 应用兼容性 23](#_Toc256000063)

[3.3.2.4 应用包格式 23](#_Toc256000064)

[3.3.2.5 JS UI框架 23](#_Toc256000065)

[3.3.3 分布式 23](#_Toc256000066)

[3.3.4 性能和功耗 23](#_Toc256000067)

[3.3.5 安全 24](#_Toc256000068)

[3.3.6 系统和软件升级 24](#_Toc256000069)

[3.3.7 开发工具和开发选项 24](#_Toc256000070)

[3.3.8 DFX（Design for X） 24](#_Toc256000071)

[4 可选系统能力兼容性 25](#_Toc256000072)

[4.1 图形图像 26](#_Toc256000073)

[4.2 电池管理服务 26](#_Toc256000074)

[4.3 电源管理服务 26](#_Toc256000075)

[4.4 USB服务 27](#_Toc256000076)

[4.5 AI引擎 27](#_Toc256000077)

[4.6 媒体 27](#_Toc256000078)

[4.7 音频 28](#_Toc256000079)

[4.8 相机 29](#_Toc256000080)

[4.9 图像编解码 29](#_Toc256000081)

[4.10 电路域电话短信服务 30](#_Toc256000082)

[4.11 通话管理服务 30](#_Toc256000083)

[4.12 输入法服务框架 30](#_Toc256000084)

[4.13 定时&时间时区服务 30](#_Toc256000085)

[4.14 密钥和凭据 30](#_Toc256000086)

[4.15 应用权限管理 31](#_Toc256000087)

[4.16 应用完整性校验 32](#_Toc256000088)

[4.17 多模输入服务 32](#_Toc256000089)

[4.18 Ability assistant工具 32](#_Toc256000090)

[5 硬件兼容性 33](#_Toc256000091)

[5.1 内存和存储 33](#_Toc256000092)

[6 软件兼容性 34](#_Toc256000093)

[6.1 API兼容性 34](#_Toc256000094)

[6.2 HDI兼容性 35](#_Toc256000095)

[6.3 运行时兼容性 35](#_Toc256000096)

[6.4 应用兼容性 35](#_Toc256000097)

[6.5 应用包格式 35](#_Toc256000098)

[6.6 JS UI框架 36](#_Toc256000099)

[7 分布式兼容性 37](#_Toc256000100)

[7.1 分布式硬件 37](#_Toc256000101)

[7.2 分布式软总线 37](#_Toc256000102)

[7.3 分布式数据管理 39](#_Toc256000103)

[7.4 分布式任务调度 39](#_Toc256000104)

[8 性能和功耗兼容性 40](#_Toc256000105)

[9 系统和软件升级 41](#_Toc256000106)

[10 开发工具和开发选项 42](#_Toc256000107)

[11 应用兼容性分发规则 43](#_Toc256000108)

[12 认证测试套件 45](#_Toc256000109)

[13 修订记录 46](#_Toc256000110)

[13.1 版本更新记录 46](#_Toc256000111)

[13.2 此版本兼容性条款修订记录 46](#_Toc256000112)

# 概述（发布到网站时独立于PCS文档之外）

[1.1 OpenHarmony系统简介](#_ZH-CN_TOPIC_0000001196218683)

[1.2 OpenHarmony兼容性目标](#_ZH-CN_TOPIC_0000001196138789)

[1.3 OpenHarmony产品兼容性规范文档说明](#_ZH-CN_TOPIC_0000001150258904)

## OpenHarmony系统简介

OpenHarmony是一款面向全场景、全连接、全智能时代的开源操作系统，采用部件化设计，支持在128KiB到xGiB RAM资源的设备上运行，设备开发者可基于目标硬件能力自由选择系统部件进行集成。

为了保证在不同硬件上易集成，OpenHarmony定义了四种基础系统类型，设备开发者通过选择基础系统类型完成必选部件集配置后，便可实现其最小系统的开发。这四种基础系统类型的参考定义如下：

* 轻量系统（mini system）

面向MCU类处理器例如Arm Cortex-M、RISC-V 32位的设备，硬件资源极其有限，支持的设备最小内存为128KiB，可以提供多种轻量级网络协议，轻量级的图形框架，以及丰富的IOT总线读写部件等。可支撑的产品如智能家居领域的连接类模组、传感器设备、穿戴类设备等。

* 小型系统（small system）

面向应用处理器的设备，支持的设备最小内存为1MiB，可以提供更高的安全能力、标准的图形框架、视频编解码的多媒体能力。可支撑的产品如智能家居领域的IP Camera、电子猫眼、路由器以及行车记录仪等。

* 标准系统（standard system）

面向应用处理器的设备，支持的设备最小内存为128MiB，可以提供增强的交互能力、GPU以及硬件合成能力、更多控件以及动效更丰富的图形能力、完整的应用框架。可支撑的产品如带屏IOT设备、轻智能手机等。

* 大型系统（large system）

面向应用处理器的设备，支持的设备最小内存为1GiB，提供多模交互能力、GPU及硬件合成能力、丰富控件及动效的图形能力、完整的应用框架。可支撑的产品如智慧屏、智能手表、智能手机等。



如上几种基础系统中所支持的最小内存，[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82)的次方单位采用[IEC 60027-2](https://zh.wikipedia.org/wiki/IEC_60027-2)标准中定义的[二进制前缀](https://zh.wikipedia.org/wiki/%E4%BA%8C%E8%BF%9B%E5%88%B6%E5%89%8D%E7%BC%80)。

OpenHarmony也提供了一系列可选的系统部件，方便设备开发者按需配置，以支撑其特色功能的扩展或定制开发。系统将这些可选的系统部件组合为一系列描述为特性或功能的系统能力，以方便设备开发者理解和选择。

## OpenHarmony兼容性目标

鉴于OpenHarmony部件可以按需拼装，即允许不同设备提供不同的系统能力，并且由于OpenHarmony源代码开源，可能导致同一个功能的源码经过不同设备开发修改后在软硬件规格方面也会有差异。但每个用户和开发者都需要一个共同的生态系统，即在此生态系统中的设备是兼容可互通的，以及设备上可运行应用的用户体验是一致的。因此，需要为设备开发定义统一的兼容性设计要求，以保证所有基于OpenHarmony开发的设备都能够兼容这个共同的生态系统，并且可以通过明确的设备认证流程来加入这个生态。

从用户和开发者视角的OpenHarmony兼容性目标如下：

* 用户视角
  1. 单设备功能可用：
     1. 单设备功能/特性具有延续性；
     2. 合法渠道安装的应用功能可用。
  2. 设备组合功能可用：
     1. 基于使用场景，可以通过组合相关设备完成场景功能；
     2. 基于使用场景，有统一的场景体验，不依赖具体厂家硬件差异；
     3. 基于使用场景，以及当前设备组合状态，符合预期的一致业务体验；
     4. 设备加入/退出已有场景组合时，业务体验可以预期；
     5. 设备可以在不同使用场景下，参与设备组合并完成功能。
* 开发者视角
  1. 应用开发者：应用可根据兼容性正确分发，跨设备/跨版本兼容运行，对用户提供一致的体验。
  2. 设备开发者：
     1. 用于指导集成OpenHarmony系统的统一的软硬件设计规范；
     2. 明确的设备认证/发布要求与指导；
     3. 统一的基础协议相关指导及兼容性要求以支撑实现与其他设备互通。

OpenHarmony通过如下三个部分来达成如上兼容性目标：

* [OpenHarmony开源项目](https://gitee.com/openharmony)的源代码。开发者可以免费获得并移植到设备。
* 产品兼容性规范文档（PCS）（待PCS发布后插入超链接），定义设备兼容性的标准。开发者在设备开发过程中需要对此文档定义的规则进行遵循。
* 产品兼容性认证测试套件（XTS）（待xts发布后插入超链接），提供验证设备兼容性的执行机制。开发者在设备开发过程中可以借助此套件对兼容性进行评估。

## OpenHarmony产品兼容性规范文档说明

OpenHarmony产品兼容性规范文档（Product Compatibility Specifications，以下简称PCS）主要从API兼容性、分布式接口兼容性方面对生态设备的设计进行约束，以努力促进OpenHarmony应用在所有OpenHarmony生态设备上的兼容性运行和分布式互通。同时，也会从软硬件基础规格等方面对生态设备的设计进行约束，以努力保证用户在OpenHarmony生态设备上的同一个使用场景的体验一致性。需要明确的是，PCS文档的作用是整理已经明确的兼容性要求，并非综合全面的设备兼容性开发指导。由于OpenHarmony代码是开源的，设备开发时建议优先选择开源实现作为首选实现方案，如果要修改开源实现，请下载相关开源代码并详细阅读对应的兼容性条款并遵循其规则。

PCS文档通过明确的条款规范了设备必须满足哪些规范才能与最新版本的OpenHarmony兼容。认证测试套件是针对PCS文档中明确的条款的对应测试套件，并非综合全面的设备兼容性测试，即通过兼容性测试只是是遵循PCS条款的必要条件。

PCS文档跟随OpenHarmony的版本发布而更新。

# 介绍

本文档用于定义OpenHarmony产品兼容性规范（Product Compatibility Specifications，以下简称PCS），明确API兼容性、分布式接口兼容性、软硬件基础规格等方面的具体要求，明确源代码的修改和使用规则。

**用词约定**

本文中出现的“必须（MUST）”、“禁止（MUST NOT）”、“必需（REQUIRED）”、“会（SHALL）”、“不会（SHALL NOT）”、“应（SHOULD）”、“不应（SHOULD NOT）”、“建议（RECOMMENDED）”、“强烈建议（STRONGLY RECOMMENDED）”、“可以（MAY）”和“可选（OPTIONAL）”用词遵循RFC2119中定义的IETF标准。

**文档结构**

第3 基础系统兼容性节定义特定系统类型的兼容性规范，第4 可选系统能力兼容性节定义可选系统能力的兼容性规范，第5 硬件兼容性节及之后定义普遍适用于所有OpenHarmony系统类型的兼容性规范。

针对规范所需“基础系统ID”定义如下：

* M：适用于轻量系统（mini system）。
* S：适用于小型系统（small system）。
* STD：适用于标准系统（standard system）。
* L：适用于大型系统（large system）。
* ALL：适用于所有系统。

针对“条件必选”兼容性规范定义如下：

* C：Conditional，有条件必选规范条款，如果设备选择集成了某个系统能力时需要遵循。

针对“通用”兼容性规范定义如下：

* G：General，通用规范条款，所有系统可引用。

针对规范定义所需的“TYPE”定义如下：

* HARDWARE：适用于硬件兼容性的条款。
* SOFTWARE：适用于软件兼容性的条款。
* UPDATE：适用于设备与APP升级的兼容性条款。
* DISTRIBUTE：适用于分布式兼容性的条款。
* PERFORMANCE：适用于性能兼容性的条款。
* POWER：适用于功耗兼容性的条款。
* SECURITY：适用于安全的兼容性条款。
* MEDIA：适用于多媒体兼容性的条款。
* TOOLS：关于开发工具与选项的兼容性条款。
* CERTIFICATION：关于认证测试的兼容性条款。
* DFX：Design for X，关于的可靠性、可测试性、可服务性、可供应性、可伸缩性、能效与环境设计等方面的兼容性条款。

**强制规范定义**

对每个“必须（MUST）”规范定义的条款格式：

1. 针对基础系统定义的ID格式为【基础系统ID-TYPE-XXXX】(例如【M-HARDWARE-0100】)。
2. 针对条件必选定义的ID格式为【C-基础系统ID[ | 基础系统ID...]-TYPE-XXXX】(例如【C-M|S-HARDWARE-0100】)。
3. 通用规范条款定义的ID格式为【G-TYPE-\*】(例如【G-HARDWARE-0100】)。

**强烈建议规范定义**

对每个“强烈建议（STRONGLY RECOMMENDED）”规范定义的条款格式：

1. 针对基础系统定义的ID格式为【基础系统ID-TYPE-SR-XXXX】（例如【M-HARDWARE-SR-0100】）。
2. 针对条件必选定义的ID格式为【C-基础系统ID[ | 基础系统ID...]-TYPE-SR-XXXX】（例如【C-M|S-HARDWARE-SR-0100】）。
3. 通用规范条款定义的ID格式为【G-TYPE-SR-XXXX】(例如针对硬件兼容性的强烈建议为【G-HARDWARE-SR-0100】)。

# 基础系统兼容性

采用OpenHarmony系统开发的设备，必须选择一种基础系统类型进行必需系统部件的集成，以支持其最小系统的实现。

选择某一种基础系统类型开发的设备必须遵循本章定义的对应基础系统兼容性规范。

[3.1 轻量系统（mini system）兼容性](#_ZH-CN_TOPIC_0000001150258888)

[3.2 小型系统（small system）兼容性](#_ZH-CN_TOPIC_0000001150099062)

[3.3 标准系统（standard system）兼容性](#_ZH-CN_TOPIC_0000001196218695)

## 轻量系统（mini system）兼容性

### 硬件

#### CPU

【M-HARDWARE-SR-0100】主CPU的DMIPS大于100 MIPS。

#### 图形显示

如果提供图形显示能力，必须遵循4.1 图形图像定义的硬件兼容性规范。

#### 内存和存储

必须遵循5.1 内存和存储中定义的硬件兼容性规范。

**最小内存和存储**

为了定义可以运行符合OpenHarmony兼容性规范的最小系统，以及完成设备的基本业务功能：

【M-HARDWARE-SR-0200】设备整体RAM空间≥128KiB。

【M-HARDWARE-SR-0201】设备整体可读写非易失存储空间≥2MiB。

【M-HARDWARE-SR-0202】用户可读写非易失存储空间≥128KiB。

#### 通信

必须遵循7.2 分布式软总线中定义的硬件兼容性规范。

#### 摄像头

如果设备支持摄像头，必须遵循4.8 相机定义的硬件兼容性规范。

#### 音频

如果设备支持音频输入，必须遵循4.7 音频定义的硬件兼容性规范。

#### USB

如果设备支持USB，必须遵循4.4 USB服务定义的硬件兼容性规范。

#### 加解密和安全隔离运行环境

如果设备支持硬件加解密模块，必须遵循4.14 密钥和凭据定义的硬件加密兼容性规范。

如果设备支持TEE，必须遵循4.14 密钥和凭据定义的硬件TEE兼容性规范。

### 软件

#### 最小集API兼容性

【M-SOFTWARE-0100】必须保证基础系统的最小集API的兼容性，API兼容性定义参见6.1 API兼容性。轻量系统的最小集API清单如下：

轻量系统最小集API列表

| 必选子系统 | 需要兼容的API |
| --- | --- |
| 内核 | liteos\_m内核提供的CMSIS和POSIX标准API |
| 驱动框架 | / |
| 语言编译器运行时 | 系统基础库libc、libc++提供的API |
| 公共基础库 | 如下头文件中定义的API：  utils/native/lite/include/kv\_store.h  如果使用了文件系统, 需如下头文件定义的API，文件数目和文件长度受内核和预留flash尺寸约束：  utils/native/lite/include/utils\_file.h |
| DFX | 如下头文件中定义的API：  base/hiviewdfx/hilog\_lite/interfaces/native/kits/hilog\_lite/log.h  base/hiviewdfx/hilog\_lite/interfaces/native/kits/hilog\_lite/hiview\_log.h |
| 启动恢复 | 如下头文件中定义的API：  base/startup/syspara\_lite/interfaces/kits/parameter.h |
| 电源管理 | / |
| 帐号服务 | / |
| 分布式软总线 | / |
| 分布式数据管理 | / |
| 分布式任务调度 | / |



表中“需要兼容的API”列注明“/”表示该必选子系统当前还不涉及对应用开放的API。

【M-SOFTWARE-0101】为了保证设备之间提供一致的与设备和版本相关的描述信息，系统对这些信息定义了统一的格式规范，设备开发必须遵循这些规范。设备信息规范定义如下表：

设备信息接口列表

| API接口 | 字符范围 | 名称，说明和要求 | 举例 |
| --- | --- | --- | --- |
| GetDeviceType() | 基础字符 | 设备类型。  也叫品类，最大32字符。 | linkiot  ipcamera |
| GetManufacture() | 基础字符 | 公司英文名简称。  最大32字符 | HUAWEI |
| GetBrand() | 基础字符 | 品牌英文名称。  最大32字符 | HUAWEI |
| GetMarketName() | utf8编码 | 外部产品系列名称。最大32字符。用户可见 | Mate 30 |
| GetProductSeries() | 基础字符 | 产品系列英文名称。  最大32字符。用以对不同类型产品进行分类管理。 | TAS |
| GetProductModel() | 基础字符 | 型号。  设备关键信息之一，用以标识设备的类型，用户可见，通常打印在设备铭牌上，用以区分不同产品。该值也是进行OpenHarmony认证所需的关键数据，需要采用英文描述，最大32字符。 | TAS-AL00 |
| GetSoftwareModel() | 基础字符 | 内部软件子型号。  最大32字符。多硬件共软件时区分软件分支 | TAS-AL00 |
| GetHardwareModel() | 基础字符 | 硬件版本号。最大32字符 | TASAL00CVN1 |
| GetSerial() | 基础字符 | 设备序列号。最大64字符。  非配置，从API读取 | 随设备差异 |
| GetOsFullName() | 基础字符 | 操作系统及版本号。名称与版本号之间以"-"分隔。最大64字符 | OpenHarmony-2.0.1.27 |
| GetDisplayVersion() | utf8编码 | 用户可见的软件版本号。注意是整个系统的软件版本号而非OpenHarmony版本号。最多64字符 | 1.0.0.6 |
| GetBootloaderVersion() | 基础字符 | Bootloader版本号。最多64字符 | u-boot-v2019.07 |
| GetSecurityPatchTag() | 基础字符 | 安全补丁标签。标识当前OS的安全补丁级别。最多64字符 | 2021-01-01 |
| GetAbiList() | 基础字符 + "," | 用逗号隔开，仅有生态且生态中包含native应用的系统使用。最多64字符 | riscv-liteos  arma7\_hard\_neon-vfpv4-liteos  arma7\_soft-liteos  arma7\_softfp\_neon-vfpv4-liteos |
| GetSdkApiVersion() | 整型 | 系统软件API version。  设备当前版本API级别，一般是整数，仅有生态的系统使用 | 3 |
| GetFirstApiVersion() | 整型 | 设备首版本的系统软件API version。  一般是整数，仅有生态的系统使用 | 3 |
| GetIncrementalVersion() | 基础字符 | 差异版本号。  在设备型号确定的情况下，即在"设备类型"+"公司"+"品牌"+"产品系列"+"操作系统及版本号"+"型号"+"内部硬件子型号"+"内部软件子型号"均相同的情况下，此值必须可以唯一标识软件版本。最多64字符 | 1.0.0.6 |
| GetVersionId() | 基础字符 + "/" + ":" | 版本Id。最大127字符。由多个字段拼接而成：  $(设备类型) + '/' + $(公司英文名简称) + '/' + $(品牌英文名称) + '/' + $(产品系列英文名称) + '/' + $( 操作系统及版本号 ) + '/' + $(型号) + '/' + $(内部软件子型号) + '/' + $(系统软件API level ) + '/' + $(差异版本号) + '/' + $(构建类型)  在所有厂家的所有设备范围中，可以唯一标识版本 | NA |
| GetBuildType() | 基础字符 + ":" | 构建类型。  同一基线代码的不同构建类型，比如debug/release、log/nolog  可以用多个标识，分号分隔，最多32字符 | release:nolog |
| GetBuildUser() | utf8编码 | 构建user，最多32字符 | NA |
| GetBuildHost() | utf8编码 | 构建host，最多32字符 | NA |
| GetBuildTime() | [0-9]+ | 构建时间，Epoch Time，自1970年至今的秒数；例如 1294902266；  最多32字符。 | NA |



上表中：

1）基础字符：[a-zA-Z0-9\_-.()]；utf8编码字符串的最大长度都是最大的字节长度。

2）API接口列：加粗斜体表示其返回值在产品生命周期不变化，普通字体表示软件升级后可能变化。

#### 运行时兼容性

必须遵循6.3 运行时兼容性中定义的运行时兼容性规范。

#### 应用兼容性

如果设备支持安装三方应用，则设备开发必须遵循6.4 应用兼容性中定义的应用兼容性规范。

#### 应用包格式

如果设备支持安装三方应用，则设备开发必须遵循6.5 应用包格式中定义的应用兼容性规范。

#### JS UI框架

带屏设备如果选择集成JS UI框架，则设备开发必须遵循6.6 JS UI框架定义的兼容性规范。

### 分布式

如果采用分布式软总线协议进行组网，必须遵循7.2 分布式软总线定义的兼容性规范。

### 性能和功耗

必须遵循8 性能和功耗兼容性定义的兼容性规范。

### 安全

强烈建议集成4.14 密钥和凭据系统能力并遵循其定义的兼容性规范。

强烈建议集成4.15 应用权限管理系统能力并遵循其定义的兼容性规范。

强烈建议集成4.16 应用完整性校验系统能力并遵循其定义的兼容性规范。

### 系统和软件升级

必须遵循9 系统和软件升级定义的兼容性规范。

### 开发工具和开发选项

必须遵循10 开发工具和开发选项定义的兼容性规范。

### DFX（Design for X）

【M-DFX-0100】打印流水日志必须使用hilog接口，并且要按照接口要求定义模块标识，确保日志可追溯。

## 小型系统（small system）兼容性

### 硬件

#### CPU

【S-HARDWARE-SR-0100】主CPU的DMIPS大于900 MIPS。

【S-HARDWARE-SR-0101】建议支持NEON加速。

【S-HARDWARE-SR-0102】建议支持浮点计算FPU。

#### 图形显示

如果提供图形显示能力，必须遵循4.1 图形图像定义的硬件兼容性规范。

#### 内存和存储

必须遵循5.1 内存和存储中定义的硬件兼容性规范。

**最小内存和存储**

为了定义可以运行符合OpenHarmony兼容性规范的最小系统，以及完成设备的基本业务功能：

【S-HARDWARE-SR-0200】设备整体RAM空间≥64MiB。

【S-HARDWARE-SR-0201】设备整体ROM空间≥16MiB。

【S-HARDWARE-SR-0202】用户剩余可用RAM空间≥5MiB。

【S-HARDWARE-SR-0203】用户可读写非易失存储空间≥256KiB。

#### 通信

必须遵循7.2 分布式软总线中定义的硬件兼容性规范。

#### 摄像头

如果设备支持摄像头，必须遵循4.8 相机定义的硬件兼容性规范。

#### 音频

如果设备支持音频输入，必须遵循4.7 音频定义的硬件兼容性规范。

#### USB

如果设备支持USB，必须遵循4.4 USB服务定义的硬件兼容性规范。

#### 加解密和安全隔离运行环境

如果设备支持硬件加解密模块，必须遵循4.14 密钥和凭据定义的硬件加密兼容性规范。

如果设备支持TEE，必须遵循4.14 密钥和凭据定义的硬件TEE兼容性规范。

### 软件

#### 最小集API兼容性

【S-SOFTWARE-0100】必须保证基础系统的最小集API的兼容性，API兼容性定义参见6.1 API兼容性。小型系统的最小集API清单如下：

小型系统最小集API列表

| 必选子系统 | 需要兼容的API |
| --- | --- |
| 内核 | 内核提供的POSIX标准API |
| 驱动框架 | / |
| 语言编译器运行时 | 系统基础库libc、libc++提供的API |
| 公共基础库 | 如下头文件中定义的API：  utils/native/lite/include/kv\_store.h |
| DFX | 如下头文件中定义的API：  base/hiviewdfx/hilog\_lite/interfaces/native/kits/hilog/log.h |
| 启动恢复 | 如下头文件中定义的API：  base/startup/syspara\_lite/interfaces/kits/parameter.h |
| 电源管理 | / |
| 帐号服务 | / |
| 分布式软总线 | / |
| 分布式数据管理 | / |
| 分布式任务调度 | / |
| Ability框架 | 如下头文件中定义的API：  foundation/aafwk/aafwk\_lite/interfaces/kits/ability\_lite/\*.h  foundation/aafwk/aafwk\_lite/interfaces/kits/want\_lite/want.h |
| 用户程序框架 | 如下头文件中定义的API：  foundation/appexecfwk/appexecfwk\_lite/interfaces/kits/bundle\_lite/\*.h |
| JS UI框架（带屏设备） | 1. JS模块定义的API：  @system.app  @ohos.app  @system.configuration  @ohos.configuration  @system.router  @ohos.router  @system.prompt  @ohos.prompt  2. 内置JS API：  console.debug|log|info|warn|error(message)  setTimeout(handler[, delay[, ...args]])  clearTimeout(timeoutID)  setInterval(handler[, delay[, ...args]])  clearInterval(intervalID)  3. JS UI组件：  div、list、list-item、stack、swiper、chart、image、image-animator、input、marquee、picker-view、progress、qrcode、slider、switch、text |



表中“需要兼容的API”列注明“/”表示该必选子系统当前还不涉及对应用开放的API。

【S-SOFTWARE-0101】为了保证设备之间提供一致的与设备和版本相关的描述信息，系统对这些信息定义了统一的格式规范，设备开发必须遵循这些规范。设备信息规范定义如下表：

设备信息接口列表

| API接口 | 字符范围 | 名称，说明和要求 | 举例 |
| --- | --- | --- | --- |
| GetDeviceType() | 基础字符 | 设备类型。  也叫品类，最大32字符。 | linkiot  ipcamera |
| GetManufacture() | 基础字符 | 公司英文名简称。  最大32字符 | HUAWEI |
| GetBrand() | 基础字符 | 品牌英文名称。  最大32字符 | HUAWEI |
| GetMarketName() | utf8编码 | 外部产品系列名称。最大32字符。用户可见 | Mate 30 |
| GetProductSeries() | 基础字符 | 产品系列英文名称。  最大32字符。用以对不同类型产品进行分类管理。 | TAS |
| GetProductModel() | 基础字符 | 型号。  设备关键信息之一，用以标识设备的类型，用户可见，通常打印在设备铭牌上，用以区分不同产品。该值也是进行HarmonyOS认证所需的关键数据，需要采用英文描述，最大32字符。 | TAS-AL00 |
| GetSoftwareModel() | 基础字符 | 内部软件子型号。  最大32字符。多硬件共软件时区分软件分支 | TAS-AL00 |
| GetHardwareModel() | 基础字符 | 硬件版本号。最大32字符 | TASAL00CVN1 |
| GetSerial() | 基础字符 | 设备序列号。最大64字符。  非配置，从API读取 | 随设备差异 |
| GetOsFullName() | 基础字符 | 操作系统及版本号。名称与版本号之间以"-"分隔。最大64字符 | OpenHarmony-2.0.1.27 |
| GetDisplayVersion() | utf8编码 | 用户可见的软件版本号。注意是整个系统的软件版本号而非OpenHarmony版本号。最多64字符 | 1.0.0.6 |
| GetBootloaderVersion() | 基础字符 | Bootloader版本号。最多64字符 | u-boot-v2019.07 |
| GetSecurityPatchTag() | 基础字符 | 安全补丁标签。标识当前OS的安全补丁级别。最多64字符 | 2021-01-01 |
| GetAbiList() | 基础字符 + "," | 用逗号隔开，仅有生态且生态中包含native应用的系统使用；最多64字符 | riscv-liteos  arma7\_hard\_neon-vfpv4-linux  arma7\_soft-linux  arma7\_softfp\_neon-vfpv4-linux  arma7\_hard\_neon-vfpv4-liteos  arma7\_soft-liteos  arma7\_softfp\_neon-vfpv4-liteos |
| GetSdkApiVersion() | 整型 | 系统软件API version。  设备当前版本API级别，一般是整数，仅有生态的系统使用 | 3 |
| GetFirstApiVersion() | 整型 | 设备首版本的系统软件API version。  一般是整数，仅有生态的系统使用 | 3 |
| GetIncrementalVersion() | 基础字符 | 差异版本号。  在设备型号确定的情况下，即在"设备类型"+"公司"+"品牌"+"产品系列"+"操作系统及版本号"+"型号"+"内部硬件子型号"+"内部软件子型号"均相同的情况下，此值必须可以唯一标识软件版本；最多64字符 | 1.0.0.6 |
| GetVersionId() | 基础字符 + "/" + ":" | 版本Id。最大127字符。由多个字段拼接而成：  $(设备类型) + '/' + $(公司英文名简称) + '/' + $(品牌英文名称) + '/' + $(产品系列英文名称) + '/' + $( 操作系统及版本号 ) + '/' + $(型号) + '/' + $(内部软件子型号) + '/' + $(系统软件API level ) + '/' + $(差异版本号) + '/' + $(构建类型)  在所有厂家的所有设备范围中，可以唯一标识版本 | NA |
| GetBuildType() | 基础字符 + ":" | 构建类型。  同一基线代码的不同构建类型，比如debug/release、log/nolog  可以用多个标识，分号分隔；最多32字符 | release:nolog |
| GetBuildUser() | utf8编码 | 构建user，最多32字符 | NA |
| GetBuildHost() | utf8编码 | 构建host，最多32字符 | NA |
| GetBuildTime() | [0-9]+ | 构建时间，Epoch Time，自1970年至今的秒数；例如 1294902266；  最多32字符。 | NA |



上表中：

1）基础字符：[a-zA-Z0-9\_-.()]；utf8编码字符串的最大长度都是最大的字节长度。

2）API接口列：加粗斜体表示其返回值在产品生命周期不变化，普通字体表示软件升级后可能变化。

#### 运行时兼容性

必须遵循6.3 运行时兼容性中定义的运行时兼容性规范。

【S-SOFTWARE-SR-0200】建议C++库采用libc++，支持C++17标准。

#### 应用兼容性

必须遵循6.4 应用兼容性中定义的应用兼容性规范。

#### 应用包格式

必须遵循6.5 应用包格式中定义的应用兼容性规范。

#### JS UI框架

带屏设备如果选择集成JS UI框架，则设备开发必须遵循6.6 JS UI框架定义的兼容性规范。

### 分布式

必须遵循7.1 分布式硬件定义的兼容性规范。

必须遵循7.2 分布式软总线定义的兼容性规范。

必须遵循7.4 分布式任务调度定义的兼容性规范。

### 性能和功耗

必须遵循8 性能和功耗兼容性定义的兼容性规范。

### 安全

强烈建议集成4.14 密钥和凭据系统能力并遵循其定义的兼容性规范。

强烈建议集成4.15 应用权限管理系统能力并遵循其定义的兼容性规范。

强烈建议集成4.16 应用完整性校验系统能力并遵循其定义的兼容性规范。

### 系统和软件升级

必须遵循9 系统和软件升级定义的兼容性规范。

### 开发工具和开发选项

必须遵循10 开发工具和开发选项定义的兼容性规范。

### DFX（Design for X）

【S-DFX-0100】打印流水日志必须使用hilog接口，并且要按照接口要求定义模块标识，确保日志可追溯。

## 标准系统（standard system）兼容性

### 硬件

#### CPU

【STD-HARDWARE-SR-0100】建议支持图形处理GPU。

#### 图形显示

如果提供图形显示能力，必须遵循4.1 图形图像定义的硬件兼容性规范。

#### 内存和存储

必须遵循5.1 内存和存储中定义的硬件兼容性规范。

**最小内存和存储**

为了定义可以运行符合OpenHarmony兼容性规范的最小系统，以及完成设备的基本业务功能：

【STD-HARDWARE-SR-0200】建议设备整体RAM空间≥128MiB。

【STD-HARDWARE-SR-0201】建议设备整体ROM空间≥2GiB。

#### 通信

必须遵循7.2 分布式软总线中定义的硬件兼容性规范。

#### 摄像头

如果设备支持摄像头，必须遵循4.8 相机定义的硬件兼容性规范。

#### 音频

如果设备支持音频输入，必须遵循4.7 音频定义的硬件兼容性规范。

#### USB

如果设备支持USB，必须遵循4.4 USB服务定义的硬件兼容性规范。

#### 加解密和安全隔离运行环境

如果设备支持硬件加解密模块，必须遵循4.14 密钥和凭据定义的硬件加密兼容性规范。

如果设备支持TEE，必须遵循4.14 密钥和凭据定义的硬件TEE兼容性规范。

### 软件

#### 最小集API兼容性

【STD-SOFTWARE-0100】必须保证基础系统的最小集API的兼容性，API兼容性定义参见6.1 API兼容性。标准系统的最小集API清单如下：

标准系统最小集API列表

| 必选子系统 | 需要兼容的API |
| --- | --- |
| 内核 | Linux内核提供的标准接口 |
| 驱动框架 | / |
| 语言编译器运行时 | 系统基础库libc、libc++提供的API |
| 公共基础库 | 如下JS模块定义的API：  @ohos.utils |
| DFX | 如下头文件中定义的API：  base/hiviewdfx/hilog/interfaces/native/kits/include/hilog/log.h  base/hiviewdfx/hiappevent/interfaces/native/kits/include/hiapp\_event.h |
| 启动恢复 | 如下JS模块定义的API：  @ohos.deviceinfo  @ohos.systemparameter |
| 电源管理 | 如下JS模块定义的API：  @ohos.batteryInfo  @ohos.power  @ohos.runninglock  @ohos.brightness |
| 帐号服务 | 如下JS模块定义的API：  @ohos.account.distributedaccount |
| 分布式软总线 | 如下JS模块定义的API：  @ohos.wifi |
| 分布式数据管理 | 如下JS模块定义的API：  @ohos.data |
| 分布式任务调度 | 如下JS模块定义的API：  @ohos.distributedschedule |
| Ability框架 | 如下JS模块定义的API：  @ohos.ability.wantconstant  @ohos.ability.featureability |
| 用户程序框架 | 如下JS模块定义的API：  @ohos.bundle  @ohos.app.abilitymanager |
| JS UI框架 | 1. JS模块定义的API：  @system.app  @ohos.app  @system.configuration  @ohos.configuration  @system.router  @ohos.router  @system.prompt  @ohos.prompt  2. 内置JS API：  console.debug|log|info|warn|error(message)  setTimeout(handler[, delay[, ...args]])  clearTimeout(timeoutID)  setInterval(handler[, delay[, ...args]])  clearInterval(intervalID)  3. JS UI组件：  div、list、list-item、stack、swiper、chart、image、image-animator、input、marquee、picker-view、progress、qrcode、slider、switch、text |



表中“需要兼容的API”列注明“/”表示该必选子系统当前还不涉及对应用开放的API。

【STD-SOFTWARE-0101】为了保证设备之间提供一致的与设备和版本相关的描述信息，系统对这些信息定义了统一的格式规范，设备开发必须遵循这些规范。设备信息规范定义如下表：

设备信息接口列表

| API接口 | 字符范围 | 名称，说明和要求 | 举例 |
| --- | --- | --- | --- |
| getDeviceType() | 基础字符 | 设备类型。  也叫品类，最大32字符。 | linkiot  ipcamera |
| getManufacture() | 基础字符 | 公司英文名简称。  最大32字符 | HUAWEI |
| getBrand() | 基础字符 | 品牌英文名称。  最大32字符 | HUAWEI |
| getMarketName() | utf8编码 | 外部产品系列名称。最大32字符。用户可见 | Mate 30 |
| getProductSeries() | 基础字符 | 产品系列英文名称。  最大32字符。用以对不同类型产品进行分类管理。 | TAS |
| getProductModel() | 基础字符 | 型号。  设备关键信息之一，用以标识设备的类型，用户可见，通常打印在设备铭牌上，用以区分不同产品。该值也是进行HarmonyOS认证所需的关键数据，需要采用英文描述，最大32字符。 | TAS-AL00 |
| getSoftwareModel() | 基础字符 | 内部软件子型号。  最大32字符。多硬件共软件时区分软件分支 | TAS-AL00 |
| getHardwareModel() | 基础字符 | 硬件版本号。最大32字符 | TASAL00CVN1 |
| getSerial() | 基础字符 | 设备序列号。最大64字符。  非配置，从API读取 | 随设备差异 |
| getOsName() | 基础字符 | 操作系统及版本号。名称与版本号之间以"-"分隔。最大32字符 | OpenHarmony-2.0.1.27 |
| getDisplayVersion() | utf8编码 | 用户可见的软件版本号。注意是整个系统的软件版本号而非OpenHarmony版本号。最多64字符 | 1.0.0.6 |
| getBootloaderVersion() | 基础字符 | Bootloader版本号，最多64字符。 | u-boot-v2019.07 |
| getSecurityPatchTag() | 基础字符 | 安全补丁标签。标识当前OS的安全补丁级别，最多64字符。 | 2021-01-01 |
| getAbiList() | 基础字符 + "," | 用逗号隔开，仅有生态且生态中包含native应用的系统使用；最多64字符 | arma7\_hard\_neon-vfpv4-linux  arma7\_soft-linux  arma7\_softfp\_neon-vfpv4-linux |
| getSdkApiVersion() | 整型 | 系统软件API version。  设备当前版本API级别，一般是整数，仅有生态的系统使用 | 3 |
| getFirstApiVersion() | 整型 | 设备首版本的系统软件API version。  一般是整数，仅有生态的系统使用 | 3 |
| getIncrementalVersion() | 基础字符 | 差异版本号，最多64字符。  在设备型号确定的情况下，即在"设备类型"+"公司"+"品牌"+"产品系列"+"操作系统及版本号"+"型号"+"内部硬件子型号"+"内部软件子型号"均相同的情况下，此值必须可以唯一标识软件版本 | 1.0.0.6 |
| getVersionId() | 基础字符 + "/" + ":" | 版本Id。最大127字符。由多个字段拼接而成：  $(设备类型) + '/' + $(公司英文名简称) + '/' + $(品牌英文名称) + '/' + $(产品系列英文名称) + '/' + $( 操作系统及版本号 ) + '/' + $(型号) + '/' + $(内部软件子型号) + '/' + $(系统软件API level ) + '/' + $(差异版本号) + '/' + $(构建类型)  在所有厂家的所有设备范围中，可以唯一标识版本 | NA |
| getBuildType() | 基础字符 + ":" | 构建类型。  同一基线代码的不同构建类型，比如debug/release、log/nolog  可以用多个标识，分号分隔；最多32字符 | release:nolog |
| getBuildUser() | utf8编码 | 构建user，最多32字符 | NA |
| getBuildHost() | utf8编码 | 构建host，最多32字符 | NA |
| getBuildTime() | [0-9]+ | 构建时间，Epoch Time，自1970年至今的秒数；例如 1294902266；  最多32字符。 | NA |



上表中：

1）基础字符：[a-zA-Z0-9\_-.()]；utf8编码字符串的最大长度都是最大的字节长度。

2）API接口列：加粗斜体表示其返回值在产品生命周期不变化，普通字体表示软件升级后可能变化。

#### 运行时兼容性

必须遵循6.3 运行时兼容性中定义的运行时兼容性规范。

【STD-SOFTWARE-SR-0200】建议C++库采用libc++，支持C++17标准。

【STD-SOFTWARE-0200】为了方便应用的书写，JS运行时必须支持ES2017中的async/await特性。

#### 应用兼容性

必须遵循6.4 应用兼容性中定义的应用兼容性规范。

【STD-SOFTWARE-0300】任何替代ohos核心应用程序的版本都必须遵守ohos核心应用程序提供的相同want，系统实现者必须支持ohos SDK提供的所有want。

【STD-SOFTWARE-0301】系统实现者不应对使用ohos SDK提供的want的系统应用附加任何特权，不应阻止三方应用绑定并承担对这些want的控制。

【STD-SOFTWARE-0302】ohos兼容设备必须发送公共事件want以响应适当的系统事件来通知三方应用硬件或软件环境的改变。



Want定义了一个Ability的启动信息格式，本地或者跨设备调用Ability时，需要按此格式封装启动信息，并通过startAbility/connectAbility接口携带。

【STD-SOFTWARE-0303】系统实现者不应在commonevent命名空间中添加新的公共事件类型，不应更改或扩展ohos核心应用使用的任何公共事件。

#### 应用包格式

必须遵循6.5 应用包格式中定义的应用兼容性规范。

#### JS UI框架

带屏设备如果选择集成JS UI框架，则设备开发必须遵循6.6 JS UI框架定义的兼容性规范。

### 分布式

必须遵循7.1 分布式硬件定义的兼容性规范。

必须遵循7.2 分布式软总线定义的兼容性规范。

必须遵循7.3 分布式数据管理定义的兼容性规范。

必须遵循7.4 分布式任务调度定义的兼容性规范。

### 性能和功耗

必须遵循8 性能和功耗兼容性定义的兼容性规范。

### 安全

强烈建议集成4.14 密钥和凭据系统能力并遵循其定义的兼容性规范。

强烈建议集成4.15 应用权限管理系统能力并遵循其定义的兼容性规范。

强烈建议集成4.16 应用完整性校验系统能力并遵循其定义的兼容性规范。

### 系统和软件升级

必须遵循9 系统和软件升级定义的兼容性规范。

### 开发工具和开发选项

必须遵循10 开发工具和开发选项定义的兼容性规范。

**开发工具**

【STD-TOOLS-0100】必须支持使用hdc设备连接器提供的命令行和通信协议与设备进行交互。

### DFX（Design for X）

【STD-DFX-0100】打印流水日志必须使用hilog接口，并且要按照接口要求定义模块标识，确保日志可追溯。

# 可选系统能力兼容性

OpenHarmony提供了一系列可选的系统部件，方便设备开发者按需配置，以支撑其特色功能的扩展或定制开发。系统将这些可选的系统部件组合为一系列描述为特性或功能的系统能力，以方便设备开发者理解和选择。本章为每种可选择的系统能力定义兼容性规范。

如果系统中集成了某个系统能力，则必须遵循该系统能力定义的兼容性规则，并强烈建议考虑其相应的兼容性建议。

【C-ALL-SOFTWARE-0100】如果设备选择集成了某一种系统能力，则必须保证其通过“@Syscap”注释标记的API的兼容性，API兼容性定义参见5 API兼容性。



* OpenHarmony定义的系统能力，以ohos.utils.system.SystemCapability.xxx.yyy的形式在开发者网站的API文档中进行声明。
* OpenHarmony发布的API，以“@Syscap”注释的形式来标记其对应的系统能力。

[4.1 图形图像](#_ZH-CN_TOPIC_0000001196138799)

[4.2 电池管理服务](#_ZH-CN_TOPIC_0000001196138785)

[4.3 电源管理服务](#_ZH-CN_TOPIC_0000001196218667)

[4.4 USB服务](#_ZH-CN_TOPIC_0000001196138773)

[4.5 AI引擎](#_ZH-CN_TOPIC_0000001150099068)

[4.6 媒体](#_ZH-CN_TOPIC_0000001150099084)

[4.7 音频](#_ZH-CN_TOPIC_0000001150258912)

[4.8 相机](#_ZH-CN_TOPIC_0000001196218679)

[4.9 图像编解码](#_ZH-CN_TOPIC_0000001150258878)

[4.10 电路域电话短信服务](#_ZH-CN_TOPIC_0000001150258892)

[4.11 通话管理服务](#_ZH-CN_TOPIC_0000001150099098)

[4.12 输入法服务框架](#_ZH-CN_TOPIC_0000001196218673)

[4.13 定时&时间时区服务](#_ZH-CN_TOPIC_0000001150099066)

[4.14 密钥和凭据](#_ZH-CN_TOPIC_0000001196218653)

[4.15 应用权限管理](#_ZH-CN_TOPIC_0000001150258902)

[4.16 应用完整性校验](#_ZH-CN_TOPIC_0000001196138757)

[4.17 多模输入服务](#_ZH-CN_TOPIC_0000001150099076)

[4.18 Ability assistant工具](#_ZH-CN_TOPIC_0000001150258922)

## 图形图像

**显示能力**

如果提供显示能力：

【C-ALL-HARDWARE-0200】必须在硬件profile中设置支持显示的display字段标识。

**支持2D和3D加速**

如果支持图形显示，并且支持图形的2D或3D硬件加速：

【C-ALL-HARDWARE-0201】默认情况下必须启用硬件加速。

如果使用Linux内核并且提供3D的硬件加速能力：

【C-ALL-HARDWARE-0202】必须同时支持OpenGL ES 1.1 and 2.0。

【C-ALL-HARDWARE-0203】必须支持可在libGLESv2.so库中导出相应的OpenGL ES 2.0定义的函数符号。

**软件兼容性**

对于有图形图像输出需求的设备：

【C-STD-SOFTWARE-0200】对所有UI对象操作必须在UI线程中进行。

【C-STD-SOFTWARE-0201】必须支持 OpenGL ES 3.2。

【C-STD-SOFTWARE-0202】必须支持Vulkan 1.0。

## 电池管理服务

【C-STD-SOFTWARE-0300】如果设备支持电池，则必须提供电池电量信息和充放电状态的查询能力和接口。

## 电源管理服务

【C-STD-SOFTWARE-0400】必须提供系统休眠、唤醒和低功耗管理能力。

【C-STD-SOFTWARE-0401】必须在用户明确执行设备进入不活动状态的操作之后进入休眠状态。

【C-STD-SOFTWARE-0402】如果设备已实现 ACPI 中所定义的S3/S4 电源状态，必须满足仅在应用不需要系统资源时进入S3/S4状态。当应用需要系统资源时，则必须退出S3/S4状态。

## USB服务

**USB device mode**

当设备实现支持USB device模式时候：

【C-ALL-HARDWARE-0500】端口必须可连接到具有标准A型或C型的主机模式的USB端口。

【C-ALL-HARDWARE-0501】必须通过GetHardwareProfile()中在USB标准设备描述符中报告iSerialNumber的正确值。

**USB host mode**

当设备实现支持主机模式的USB端口：

【C-ALL-HARDWARE-0502】必须支持连接标准 USB 外围设备。

【C-ALL-HARDWARE-0503】必须通过GetHardwareProfile()中usbhost提供是否支持USBHOST的标记。

## AI引擎

【C-S-SOFTWARE-0600】AI sdk需按算法调用顺序，封装client对外提供接口；对于异步插件对应的sdk，需要实现client提供的回调接口IClientCb。

【C-S-SOFTWARE-0601】AI sdk接口实现中，需要保存与client交互的相关通用数据。

【C-S-SOFTWARE-0602】AI sdk与plugin需要使用编解码模块，将特定算法数据转换成AI引擎的通用数据类型。

【C-S-SOFTWARE-0603】AI sdk中需要对以编解码返回的出参数据类型进行内存释放。

【C-S-SOFTWARE-0604】AI plugin需要实现server提供的IPlugin接口。

【C-S-SOFTWARE-0605】AI plugin需要使用AI引擎提供的统一数据通道。

如果提供视频输入能力：

【C-ALL-SOFTWARE-SR-0600】强烈推荐支持CV能力。

如果提供语音输入能力：

【C-ALL-SOFTWARE-SR-0601】强烈推荐支持ASR能力。

## 媒体

【C-S|STD-SOFTWARE-0700】必须支持PCM/WAV格式的音频编码，编码采样频率必须支持8K, 16K, 44.1K, 48K。

【C-S|STD-SOFTWARE-0701】必须支持MPEG-2 AAC Main/MPEG-2 AAC LC (Low Complexity)格式的音频编码，编码采样频率支持16K, 44.1K，48K，可选支持11.025K，22.05K，48K，176.4K，192K。

【C-STD-SOFTWARE-0702】必须支持MPEG-4 AAC Profile (AAC LC)格式解码，支持单/双声道，支持8~48KHz采样率，支持8/16/24bit位深，建议支持32bit位深，支持MPEG-4(.mp4,.m4a)容器格式。

【C-STD-SOFTWARE-0703】必须支持H.264格式Baseline Profile Level 3编码规格，必须支持SD（尺寸小于720×480）编码规格，应该支持HD（尺寸大于1280×720）编码规格。

【C-STD-SOFTWARE-0704】必须支持H.264格式Baseline Profile/Main Profile解码，MPEG-4(.mp4)容器格式。

【C-STD-SOFTWARE-SR-0700】建议支持H.265格式解码，支持MPEG-4(.mp4)容器格式。

【C-S|STD-SOFTWARE-SR-0701】建议支持 MPEG-4 AAC LD (Low Delay)/MPEG-4 AAC HE (High Efficiency) AACPlusV1/V2(3GPP)格式的音频编码。

【C-STD-SOFTWARE-SR-0702】建议支持H.265格式Baseline Profile编码规格。

## 音频

**麦克风硬件兼容性**

如果设备支持麦克风：

【C-ALL-HARDWARE-0800】必须通过GetHardwareProfile()中microphone提供是否支持麦克风的标记，以支持从配置信息判断设备是否支持麦克风硬件。

【C-ALL-HARDWARE-0801】必须支持单或双MIC输入或Line-In输入的一种或多种方式。

【C-ALL-HARDWARE-0802】必须支持16bit音频输入。

**音频输出硬件兼容性**

如果设备实现包括扬声器或者其他用于音频输出的音频/多媒体输出端口外设，例如3.5mm音频插孔,HDMI,或使用USB音频类的USB主机模式端口。其他类型的音频输出端口，例如支持基于无线电的音频输出和蓝牙等音频输出不在本章节定义。

【C-ALL-HARDWARE-0803】必须通过GetHardwareProfile()中aout提供是否支持音频输出的标识，以支持从配置信息判断设备支持的音频输出端口种类。

**软件兼容性**

【C-S|STD-SOFTWARE-0800】音频播放必须支持16K，44.1K，48K采样率音频。

【C-S|STD-SOFTWARE-0801】音频播放必须支持单声道和双声道音频。

【C-S|STD-SOFTWARE-0802】音频播放必须支持16位PCM编码格式音频 。

【C-S|STD-SOFTWARE-0803】音频采集必须支持单声道、编码格式为16位PCM、采样率为16K，48K的音频。

【C-S|STD-SOFTWARE-0804】必须支持MP3格式解码, 支持单/双声道，支持8~320kbps的固定码率、变码率模式, 支持MP3(.mp3)容器格式。

## 相机

**硬件兼容性**

【C-ALL-HARDWARE-0900】必须通过GetHardwareProfile()中camera字段提供是否支持摄像头硬件的标记，以支持从配置信息判断设备是否支持摄像头硬件。

**视频输入硬件兼容性**

【C-ALL-HARDWARE-0901】必须支持8-/10-/12-bit RGB Bayer DC timing VI 其中的一种或多种模式。

【C-ALL-HARDWARE-0902】必须支持MIPI, LVDS/sub-LVDS, and HiSPi 一种或多种模式。

【C-ALL-HARDWARE-0903】必须支持一款或多款CMOS sensor。

**ISP硬件兼容性**

【C-ALL-HARDWARE-0904】必须支持3A（Auto Exposure/Auto Focus/Auto White Balance）调节。

**软件兼容性**

【C-S|STD-SOFTWARE-0900】必须提供摄像头设备能力查询功能，包含摄像头个数，方向，类型，宽高比和分辨率信息。

【C-S|STD-SOFTWARE-0901】必须提供摄像头数据流采集功能，包含预览流、录像流和拍照流。预览流、录像流必须支持YUV420SP格式，拍照流必须支持JPEG编码格式。

【C-S|STD-SOFTWARE-0902】建议提供摄像头3A（Auto Exposure/Auto Focus/Auto White Balance）能力进行开关控制和模式切换的功能。

## 图像编解码

【C-STD-SOFTWARE-1000】必须支持JPEG格式编码。

【C-S-SOFTWARE-1001】必须支持PNG格式编码。

【C-S-SOFTWARE-1002】必须支持WebP格式编码。

【C-S-SOFTWARE-1003】必须支持JPEG格式解码, 支持基础和渐进式解码，支持JPEG(.jpg)容器格式。

【C-S-SOFTWARE-1004】必须支持GIF格式解码，支持GIF(.gif)容器格式。

【C-S-SOFTWARE-1005】必须支持PNG格式解码，支持PNG(.png)容器格式。

【C-S-SOFTWARE-1006】必须支持BMP格式解码，支持BMP(.bmp)容器格式。

【C-S-SOFTWARE-1007】必须支持WebP格式解码，支持WebP(.webp)容器格式。

【C-S-SOFTWARE-SR-01000】建议支持HEIF(HEIC)格式解码，支持HEIF(.heif)、HEIF(.heic)容器格式。

【C-S-SOFTWARE-1008】必须支持Raw格式，支持ARW (.arw)、CR2 (.cr2)、DNG (.dng)、NEF (.nef)、NRW (.nrw)、ORF (.orf)、PEF (.pef)、RAF (.raf)、RW2 (.rw2)、SRW (.srw)容器格式。

## 电路域电话短信服务

【C-STD-SOFTWARE-1100】必须提供电路域电话核心服务包含的SIM卡和搜网能力的完整实现。



1）如果设备实现包含电话硬件，则必须至少支持电话核心服务包含的SIM卡和搜网API实现，其他能力视设备规格决定是否支持，比如语音通话、短信、蜂窝数据等。

2）如果设备实现不包含电话硬件，则电话核心服务相关API可以为空实现。

## 通话管理服务

【C-STD-SOFTWARE-1200】必须提供通话管理功能的完整实现。



通话管理服务主要负责管理CS（Circuit Switch，电路交换）、IMS（IP Multimedia Subsystem，IP多媒体子系统）和OTT（over the top，OTT解决方案）三种类型的通话。

1）如果设备支持通话功能（CS通话、IMS通话、OTT通话中的一种或多种），则必须支持通话管理服务的API实现。

2）如果设备不支持通话功能，则通话管理服务相关API可以为空实现。

## 输入法服务框架

【C-STD-SOFTWARE-1300】必须支持第三方输入法应用。

【C-STD-SOFTWARE-1301】必须完整实现输入法管理框架。

【C-STD-SOFTWARE-1302】必须有预安装的输入法应用。

## 定时&时间时区服务

【C-STD-SOFTWARE-1400】必须限制通过定时器拉起已经退出的进程的行为。



定时器支持以回调或悬挂意图的方式通知创建该定时器的服务或者应用进行超时处理。系统应该限制定时器拉起已经退出的进程的行为，比如应用/服务通过定时器随意拉起非常驻的后台服务或者弹窗行为，以避免对用户体验或者设备功耗造成影响。

【C-STD-SOFTWARE-1401】必须通过权限管控修改系统时间时区的行为。

## 密钥和凭据

**硬件兼容性**

如果设备支持硬件加解密模块,必须支持以下规范：

【C-ALL-HARDWARE-1500】必须支持硬件AES256 加解密算法。

【C-ALL-HARDWARE-1501】必须支持硬件HASH-SHA256、HMAC\_SHA256 算法。

【C-ALL-HARDWARE-1502】必须支持硬件RSA、ECC 签名校验算法。

【C-ALL-HARDWARE-1503】必须支持硬件真随机数生成，满足 FIPS140-2 随机测试标准。

如果设备支持TEE，硬件必须支持以下规范：

【C-ALL-HARDWARE-1504】CPU必须支持安全和非安全状态。

【C-ALL-HARDWARE-1505】如果CPU class为cortex-A，必须支持TZASC安全内存配置和TZPC安全外设配置。

【C-ALL-HARDWARE-1505】如果CPU class为cortex-M，必须支持SAU或IDAU安全内存配置。

**软件兼容性**

【C-ALL-SOFTWARE-1500】必须支持安全随机数。

【C-ALL-SOFTWARE-1501】必须支持HASH算法，至少包括SHA256。

【C-ALL-SOFTWARE-1502】必须支持HMAC和HKDF算法，至少包括HMAC-SHA256。

【C-ALL-SOFTWARE-1503】必须支持AES对称加密算法，分组模式支持CBC、GCM，密钥长度支持128、192、256位。

【C-S|STD-SOFTWARE-1504】必须支持AES对称加密算法，分组模式支持ECB、OFB、CFB、CTR、CCM、密钥长度支持128、192、256位。

【C-S|STD-SOFTWARE-1505】必须支持非对称加密算法RSA，包括2048位、3072位、4096位，能够实现秘钥管理、加解密以及签名服务。

【C-S|STD-SOFTWARE-1506】支持椭圆曲线签名算法ECDSA和秘钥协商算法ECDH，至少支持P256曲线；支持椭圆曲线签名算法ED25519和秘钥协商算法X25519、25519曲线。

## 应用权限管理

【C-S|STD-SOFTWARE-1600】必须支持权限模型，并且执行权限模型文档中定义的每个权限，不应省略、更改或忽略任何权限。

【C-STD-SOFTWARE-1601】禁止应用申请restricted权限，除非应用包中含有restricted权限的权限证书。

【C-S|STD-SOFTWARE-1602】必须为用户提供专用的界面或接口用于授权和管理动态权限(user\_grant)，必要时用户可以撤销授权。

【C-S|STD-SOFTWARE-1603】禁止给任何的预装应用授予任何的动态权限(user\_grant)。



如下情况例外：

\* 预装应用使用动态权限前获得用户同意；

\* 预装应用为动态权限相关的默认应用，例如，短信动态权限允许授予给默认短信应用。

【C-STD-SOFTWARE-1604】设备必须为用户提供专门的界面或接口用于授权和管理应用访问其他设备的能力

【C-STD-SOFTWARE-1605】分布式场景下，主体设备必须为用户提供专门的界面或接口用于授权和管理应用访问其他设备的能力。

## 应用完整性校验

【C-ALL-SOFTWARE-1700】必须对应用进行签名，用于应用完整性和来源验证。

【C-ALL-SOFTWARE-1701】在安装应用时，必须对应用的签名进行校验，确保应用来源可信和未被篡改；禁止安装签名校验失败的应用。

## 多模输入服务

【C-STD-SOFTWARE-1800】电源键、Home键、静音键必须系统处理，不能发送给应用，返回键必须发送到应用处理。

【C-STD-SOFTWARE-1801】不能修改触摸输入事件和按键输入事件的投递规则。

## Ability assistant工具

【C-ALL-SOFTWARE-1900】必须遵守aa工具命令行语义。

# 硬件兼容性

[5.1 内存和存储](#_ZH-CN_TOPIC_0000001150258890)

## 内存和存储

内存，简称为RAM。

存储，定义为非易失性存储，分成以下几种类型：

1. ROM，只读存储器。
2. Writeable NVM（nonvolatile memory，以下简称NVM）, 芯片内或芯片外可读写非易失存储器，例如片内内置Flash或外接Flash器件。
3. portable NVM，用于外部可拆卸存储器，如SDCard。

【G-HARDWARE-0100】外部可拆卸存储必须通过mount方式安装到系统根目录，例如SDCard必须被mount成/sdcard。

【G-HARDWARE-0101】应用必须通过显式声明OHOS.permission.READ\_USER\_STORAGE和OHOS.permission.WRITE\_USER\_STORAGE，获得外部存储的读和写的权限。

# 软件兼容性

[6.1 API兼容性](#_ZH-CN_TOPIC_0000001150258908)

[6.2 HDI兼容性](#_ZH-CN_TOPIC_0000001196138801)

[6.3 运行时兼容性](#_ZH-CN_TOPIC_0000001150099102)

[6.4 应用兼容性](#_ZH-CN_TOPIC_0000001150099074)

[6.5 应用包格式](#_ZH-CN_TOPIC_0000001196138759)

[6.6 JS UI框架](#_ZH-CN_TOPIC_0000001150099050)

## API兼容性

【G-SOFTWARE-0100】必须提供OpenHarmony已公开API以及OpenHarmony社区源代码中所有用“ @SystemApi”注解的API的完整实现，并确保其行为与API文档中的定义一致。

【G-SOFTWARE-0101】禁止通过更改任何方法或类签名或者通过移除类或类字段来修改OpenHarmony已公开API。

【G-SOFTWARE-0102】修改API的底层实现时，不应影响任何已公开API的既定行为和API签名。

【G-SOFTWARE-0103】即使省略了OpenHarmony已公开API的某些硬件功能，也必须保持API的存在并以合理的方式运行。

【G-SOFTWARE-0104】不应允许三方应用调用非SDK接口，包括那些用“@hide”注解但未添加“@SystemApi”注解的接口。

【G-SOFTWARE-0105】不应区分设备来确保API行为兼容性，比如采用设备白名单的方法。

【G-SOFTWARE-0106】不应在ohos.\*命名空间中的API添加任何已公开元素（例如类或接口，或现有类或接口的字段或方法）、或“@SystemApi”注解的接口。

【G-SOFTWARE-0107】在ohos命名空间外扩展API时，不应位于归其他组织所有或提及其他组织的命名空间内。

【G-SOFTWARE-0108】禁止更改Native API接口的定义。

【G-SOFTWARE-0109】禁止添加或移除Native库中的公共函数。

## HDI兼容性

【G-SOFTWARE-0200】必须按照“[驱动开发指南](https://gitee.com/openharmony/docs/blob/master/zh-cn/device-dev/driver/Readme-CN.md" \o " )”中开发文档中所述规范进行 HDI开发。

【G-SOFTWARE-0201】如果现有驱动无法满足接口定义，应新增接口实现，不应修改现有实现。

【G-SOFTWARE-0202】如果系统服务部件中需要与某个被规定为可选部件的硬件部件交互，但设备实现不具备该组件，仍必须提供该组件 HDI的完整类定义，并以合理的方式将该 HDI 的行为实现为空操作。

## 运行时兼容性

【G-SOFTWARE-0300】C运行时的API必须遵循C99 standard、POSIX.1.2008规范。

【G-SOFTWARE-0301】JS运行时支持ES5、ES6语法规范。



ES6以下几个特性明确不支持：

with语句

eval

Function

【G-SOFTWARE-0302】JS运行时环境必须运行在严格模式。

## 应用兼容性

Ability的基本组成和运行单元是FA/PA，Ability的生命周期切换本质上是FA/PA的生命周期切换。用户对一个FA/PA的操作（如：在一个界面上打开一个新的界面，或者按Back键返回上一个界面，又或者按Home键回到主菜单），往往是多个FA/PA配合进行生命周期切换才能完成。

【G-SOFTWARE-0400】禁止修改现有want的显示匹配行为或语义。

【G-SOFTWARE-0401】系统实现者不应更改Ability的生命周期或生命周期回调语义。

【G-SOFTWARE-0402】应实现一个可让用户轻松切换到前一个FA的快捷方式。

## 应用包格式

【G-SOFTWARE-0500】必须能够安装和运行由OpenHarmony打包工具生成的hap包。

【G-SOFTWARE-0501】必须支持使用hap包签名方案签名验证“.hap”文件。

【G-SOFTWARE-0502】 扩展 .hap或者config.json清单时，采用的方式不应导致相应文件无法在其他与OpenHarmony兼容的设备上正确安装和运行。

【G-SOFTWARE-0503】应该为用户提供卸载已安装应用的入口。

【G-SOFTWARE-0504】不应允许在没有任何用户提示的情况下静默安装应用。

【G-SOFTWARE-0505】针对显式安装的应用，不应允许应用在没有用户确认的情况下静默卸载应用。

## JS UI框架

【G-SOFTWARE-0600】禁止修改config.json中js标签配置规则。

【G-SOFTWARE-0601】禁止修改HML、CSS、JS语法。

【G-SOFTWARE-0602】禁止修改JS组件，及其属性和接口定义，通过JS接口相关的兼容性测试。

# 分布式兼容性

[7.1 分布式硬件](#_ZH-CN_TOPIC_0000001150099082)

[7.2 分布式软总线](#_ZH-CN_TOPIC_0000001150258896)

[7.3 分布式数据管理](#_ZH-CN_TOPIC_0000001196218657)

[7.4 分布式任务调度](#_ZH-CN_TOPIC_0000001150258916)

## 分布式硬件

【G-DISTRIBUTE-0100】禁止修改设备UDID生成规则。

## 分布式软总线

**网络协议**

面向的设备如果有网络的需求：

【G-HARDWARE-0200】必须包括IPV4协议栈的支持,必须提供Linux BSD Socket API或者LWIP BSD Socket API之一。

【G-HARDWARE-0201】必须包括DHCPv4/Client的支持。

**WLAN**

如果设备支持WLAN：

【G-HARDWARE-0202】必须支持IEEE 802.11b/g/n。

【G-HARDWARE-0203】必须支持STA和AP模式。

【G-HARDWARE-0204】必须通过GetHardwareProfile()中WLAN字段提供是否支持wlan硬件的标识。

如果设备支持IEEE 802.11定义的WLAN 省电模式：

【G-HARDWARE-0205】必须支持在应用使用WLAN服务API中的性能或者低延迟模式中关闭省电模式。

【G-HARDWARE-0206】必须支持多播DNS（mDNS），并且在任何操作时间不应过滤mDNS数据包。

【G-HARDWARE-0207】必须支持DHCPv4 Client/Server。

【G-HARDWARE-SR-0500】作为AP时，建议支持≥2个STA接入。

【G-HARDWARE-0208】必须支持WFA WPA/WPA2personal，建议支持WPS2.0。

【G-HARDWARE-SR-0600】建议支持≥1个SDIO2.0 Slave接口。

**蓝牙**

如果设备支持蓝牙

【G-HARDWARE-0210】必须支持BLE（Bluetooth Low Energy）。

【G-HARDWARE-SR-0700】推荐支持PTA（Packet Traffic Arbitration）。

【G-HARDWARE-0211】必须通过GetHardwareProfile()中bluetooth字段提供是否支持蓝牙硬件的标识。

**分布式软总线**

【G-DISTRIBUTE-0200】禁止修改OpenHarmony分布式软总线基于CoAP的设备发现管理协议。

【G-DISTRIBUTE-0201】禁止修改OpenHarmony分布式软总线基于蓝牙BLE的设备发现管理协议。

【G-DISTRIBUTE-0202】禁止修改OpenHarmony分布式软总线设备组网的管理协议，包括设备信息交换协议及心跳规则和机制。

【G-DISTRIBUTE-0203】禁止修改或终止设备保活策略和心跳规则，如TCP的keepalive周期以及BLE广播间隔和扫描占空比。

【G-DISTRIBUTE-0204】禁止修改组网设备类型定义，需遵照统一的设备类型设定约束。

【G-DISTRIBUTE-0205】禁止修改OpenHarmony分布式软总线设备间传输通道的管理协议。



上述规则中的管理协议指软总线发布的报文格式的定义，字段定义以及字段的含义、交互的顺序。协议在保证兼容性的基础上可受控扩展，但是禁止删除和修改已有定义。

【G-DISTRIBUTE-0206】禁止直接修改传输的默认协议（TCP/UDP），新增或者变更默认传输协议必须通过协商机制来实现。

【G-DISTRIBUTE-0207】禁止修改设备UUID、NetworkId生成规则。

【G-DISTRIBUTE-0208】设备必须支持蓝牙、WiFi或以太网等软总线依赖的通信能力中的一种或者多种。

【G-DISTRIBUTE-0209】软总线使用的CoAP协议监听端口（5684）禁止修改，以免产生兼容性问题。

【G-DISTRIBUTE-0210】软总线默认使用的BLE广播的UUID（0xFDEE）禁止修改，以免产生兼容性问题，如果发生变更就无法和使用原0xFDEE地址为UUID的蓝牙设备实现互通。

【G-DISTRIBUTE-0211】使用消息传输接口，消息大小不超过4K字节，超过时需要业务对消息进行分包处理，或者改为使用字节传输接口。字节传输可支持最大4M字节（注意：不同产品因硬件限制，此上限可能有变化）。

【G-DISTRIBUTE-0213】出于安全和兼容性考虑，不应修改安全相关逻辑（permission.json检查，设备认证，会话认证），不应变更软总线依赖的安全（HiChain）和协议部件（CoAP，JSON）。

【G-DISTRIBUTE-0214】禁止修改RPC中定义的数据结构和接口，并提供对应的完整实现。

【G-DISTRIBUTE-0215】设备的NetworkId仅在设备在线时可用，设备下线后其NetworkId也随之失效。不要将NetworkId作为设备的长期的标识；不要企图修改NetworkId的生命周期，比如在对端设备下线后，继续对外提供本地缓存的对端设备NetworkId，否则可能引发不可预知的问题。

【G-DISTRIBUTE-SR-0200】传输层的安全不代表业务安全的达成，建议机密数据的传输应在使用软总线接口前就完成安全加密，采用业务到业务间的端到端安全机制。

## 分布式数据管理

【G-DISTRIBUTE-0300】禁止修改分布式数据服务进行设备间数据通步的协议帧、包结构以及加解密算法。

【G-DISTRIBUTE-0301】禁止删除和修改service\_meta数据库存贮的KvStoreMetaData的已有字段以及Key的生成方法。

【G-DISTRIBUTE-0302】禁止改变分布式同步的数据的key和value的规格。



进行分布式同步的数据，必须满足每个key不超过1KByte，每个value不超过4MByte。

【G-DISTRIBUTE-0303】禁止修改系统定义的用于存储FeatureAbility、ParticleAbility的map表的格式。

## 分布式任务调度

【G-DISTRIBUTE-0400】系统服务必须继承SystemAbility框架实现。

【G-DISTRIBUTE-0401】禁止修改系统定义的用于存储SystemAbility的map表的格式。

【G-DISTRIBUTE-0402】禁止修改SystemAbility框架定义的数据结构和接口，并提供完整实现。

# 性能和功耗兼容性

**IO性能**

如果设备支持Wi-Fi 2.4G:

【G-PERFORMANCE-SR-0100】强烈建议支持2.4G的网络IO性能≥10Mbps。

**低功耗模式**

低功耗模式可以有效的减少芯片的功耗，需要芯片提供多种低功耗的控制来动态降低芯片的功耗：

【G-POWER-SR-0100】强烈建议提供时钟门控功能，在模块空闲的时候，可以关闭相应的时钟。

【G-POWER-SR-0101】强烈建议提供模块的工作频率动态调整功能。

# 系统和软件升级

【G-UPDATE-0100】设备必须支持提供对整个系统软件升级的机制。升级机制可以包括：在线升级，USB线缆升级或者移动存储卡升级等。

【G-UPDATE-0101】设备升级过程必须支持不擦除用户数据。

【G-UPDATE-0102】设备升级前必须对升级包进行基于公私钥对的签名校验。

【G-UPDATE-SR-0100】签名校验方式强烈建议使用SHA-256进行哈希计算，通过RSA3072进行验证。

【G-UPDATE-0103】如果设备具备连接因特网能力，必须支持在线下载升级包并完成设备的升级。

【G-UPDATE-0104】设备升级过程中必须支持掉电保护功能。

【G-UPDATE-SR-0101】建议对全部系统分区进行A/B备份或者只对关键分区进行A/B备份。

# 开发工具和开发选项

**开发工具**

【G-TOOLS-0100】必须支持OpenHarmony调试命令控制台。

【G-TOOLS-0101】必须支持OpenHarmony调试命令集合。

【G-TOOLS-0102】必须支持OpenHarmony调试所需权限。

【G-TOOLS-0103】必须支持OpenHarmony调试节点的完整性。

【G-TOOLS-0104】必须满足OpenHarmony日志、trace等格式输入输出要求。

【G-TOOLS-0105】必须满足OpenHarmony hiprofiler相关系统数据获取格式。

【G-TOOLS-0106】必须满足OpenHarmony升级方式不允许改变。

【G-TOOLS-0107】必须满足OpenHarmony工具链API与命令的兼容不发生改变。

**开发选项**

【G-TOOLS-0108】必须支持进入OpenHarmony调试模式。

【G-TOOLS-0109】必须支持OpenHarmony调试配置。

# 应用兼容性分发规则

OpenHarmony系统支持应用一次开发多设备部署，为此，OpenHarmony将系统部件描述为系统能力（SystemCapability），并通过在API中标注“@Syscap”来实现API与系统能力的关联，应用开发时可以根据其调用的API对应的“@Syscap”标注来要求该应用运行时依赖的系统能力。应用市场向设备分发应用时，需要先判断该设备具备应用必需的系统能力后才能向该设备分发，否则不应向该设备使用者展示此应用或者通过免安装流程使得此应用在该设备上运行。

兼容性标识的定义

PCID： Product Compatibility ID，产品兼容性标识，南向设备用以标记设备支持的基础系统类型和选配的系统能力。

RPCID：Required Product Compatibility ID，要求产品兼容性标识，北向应用用以标记应用静态依赖的系统能力。

兼容性标识的构成

| 字段定义 | 字段说明 |
| --- | --- |
| Version | **数据范围**：bit0~bit15。  **数据说明**：  1、PCID中取设备开发采用的OpenHarmony系统系统的API Version，bit15固定0。  2、RPCID中取应用开发使用的OpenHarmony SDK中的API Version，bit15固定1。 |
| BCG（Base Components Group） | **数据范围**：bit16~bit31。  **数据说明**：  1、bit含义  bit16: mini system  bit17: small system  bit18: standard system  bit19: large system  bit20~bit31: reserved  2、PCID中，只能某一个bit设置为1，表示此系统类型。默认全0，表示系统类型未定义。  3、RPCID中，可以多个bit设置为1，表示该APP支持运行在多个系统上。默认全1，表示支持运行在所有系统类型的设备上。 |
| MID（Manufacturer ID） | **数据范围**：bit32~bit63。  **数据说明**：  1、此字段用以标识一个唯一的厂家信息，如果不存在扩展系统能力，则此字段写0。  2、此字段在设备提交认证时，如果涉及扩展的系统能力，由兼容性认证管理平台统一分配一个非0值。 |
| OCG（Optional Components Group） | **数据范围**：采用TLV格式封装。  T：SystemCapability ID，共32bits。其中bit31固定为0，表示此SystemCapability为系统定义的系统能力。  L：Length，16bits。  V：Value，≥16bits。  **数据说明**：  1、此部分由系统定义和维护。  2、支持的SystemCapability才写入。 |
| PCG（Product Components Group） | **数据范围**：采用TLV格式封装。  T：Ext SystemCapability ID，共32bits。其中bit31固定为1，表示此SystemCapability为OEM扩展的系统能力。  L：Length，16bits。  V：Value，≥16bits。  **数据说明**：  1、此部分由设备开发者定义和维护，并在兼容性认证时提交给认证管理平台。  2、支持的SystemCapability才写入。 |

# 认证测试套件

【G-CERTIFICATION-0100】必须通过OpenHarmony要求的所有认证测试。

【G-CERTIFICATION-0101】必须使用配套OpenHarmony发布的最新认证测试套件进行认证测试。

# 修订记录

[13.1 版本更新记录](#_ZH-CN_TOPIC_0000001196138765)

[13.2 此版本兼容性条款修订记录](#_ZH-CN_TOPIC_0000001196218647)

## 版本更新记录

为兼容更多不同硬件能力的设备运行OpenHarmony系统，此版本对兼容性定义所做更改主要如下：

1. 以硬件能力范围为基础定义了4种基础系统配置，并为每种基础系统定义了兼容性规范；
2. 为每种基于基础系统可选的系统能力定义了兼容性规范；
3. 增加了对RAM资源≥128MiB设备的支持；
4. 根据如上变化调整了文档的目录结构。

## 此版本兼容性条款修订记录

本次为OpenHarmony 产品兼容性规范 3.0版本首次发布。