OpenHarmony3.0 产品兼容性规范

OpenHarmony3.0 产品兼容性规范

文档版本	01
发布日期	2021-12-31

OpenHarmony 兼容性工作组

目 录

1 概述……1
1.1 OpenHarmony 系统简介 1
1.2 OpenHarmony 兼容性目标 2
1.3 OpenHarmony 产品兼容性规范文档说明 3
2 介绍……4
3 基础系统兼容性……6
3.1 轻量系统（mini system）兼容性 6
3.1.1 硬件 6
3.1.1.1 CPU 6
3.1.1.2 图形显示 6
3.1.1.3 内存和存储 6
3.1.1.4 通信 7
3.1.1.5 摄像头 7
3.1.1.6 音频 7
3.1.1.7 USB 7
3.1.1.8 加解密和安全隔离运行环境 7
3.1.2 软件 7
3.1.2.1 最小集 API 兼容性 7
3.1.2.2 运行时兼容性 12
3.1.2.3 应用兼容性 12
3.1.2.4 应用包格式 12
3.1.2.5 Arc UI 框架 13
3.1.3 分布式 13
3.1.4 性能和功耗 13
3.1.5 安全 13
3.1.6 系统和软件升级 13
3.1.7 开发工具和开发选项 13
3.1.8 DFX（Design for X） 13
3.2 小型系统（small system）兼容性 13

3.2.1 硬件 13
3.2.1.1 CPU 13
3.2.1.2 图形显示 13
3.2.1.3 内存和存储 14
3.2.1.4 通信 14
3.2.1.5 摄像头 14
3.2.1.6 音频 14
3.2.1.7 USB 14
3.2.1.8 加解密和安全隔离运行环境 14
3.2.2 软件 14
3.2.2.1 最小集 API 兼容性 14
3.2.2.2 运行时兼容性 20
3.2.2.3 应用兼容性 20
3.2.2.4 应用包格式 20
3.2.2.5 Arc UI 框架 20
3.2.3 分布式 20
3.2.4 性能和功耗 20
3.2.5 安全 21
3.2.6 系统和软件升级 21
3.2.7 开发工具和开发选项 21
3.2.8 DFX（Design for X） 21
3.3 标准系统（standard system）兼容性 21
3.3.1 硬件 21
3.3.1.1 CPU 21
3.3.1.2 图形显示 21
3.3.1.3 内存和存储 21
3.3.1.4 通信 22
3.3.1.5 摄像头 22
3.3.1.6 音频 22
3.3.1.7 USB 22
3.3.1.8 加解密和安全隔离运行环境 22
3.3.2 软件 22
3.3.2.1 最小集 API 兼容性 22
3.3.2.2 运行时兼容性 28
3.3.2.3 应用兼容性 28
3.3.2.4 应用包格式 29
3.3.2.5 JS UI 框架 29
3.3.3 分布式 29
3.3.4 性能和功耗 29

3.3.5 安全 29
3.3.6 系统和软件升级 29
3.3.7 开发工具和开发选项 29
3.3.8 DFX（Design for X） 30

4 可选系统能力兼容性 31

4.1 图形图像 32
4.2 电池管理服务 32
4.3 电源管理服务 32
4.4 USB 服务 32
4.5 AI 引擎 33
4.6 媒体 33
4.7 音频 34
4.8 相机 35
4.9 图像编解码 35
4.10 电路域电话短信服务 36
4.11 通话管理服务 36
4.12 输入法服务框架 36
4.13 定时&时间时区服务 36
4.14 密钥和凭据 37
4.15 应用权限管理 38
4.16 应用完整性校验 38
4.17 多模输入服务 38
4.18 Ability assistant 工具 38

5 硬件兼容性 39

5.1 内存和存储 39

6 软件兼容性 40

6.1 API 兼容性 40
6.2 HDI 兼容性 41
6.3 运行时兼容性 41
6.4 应用兼容性 41
6.5 应用包格式 42
6.6 Arc UI 框架 42

7 分布式兼容性 43

7.1 分布式硬件 43
7.2 分布式软总线 43
7.3 分布式数据管理 45

7.4 分布式任务调度.....45

8 性能和功耗兼容性.....46

9 系统和软件升级.....47

10 开发工具和开发选项.....48

11 兼容性测试套件.....49

12 修订记录.....50

1 概述

1.1 OpenHarmony 系统简介

1.2 OpenHarmony 兼容性目标

1.3 OpenHarmony 产品兼容性规范文档说明

1.1 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 内存资源的设备上运行，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。

为保证在不同硬件上易集成，OpenHarmony 定义了三种基础系统类型，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。三种基础系统类型的参考定义：

- 轻量系统（mini system）

  面向 MCU 类处理器的设备，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）

  面向具备 MMU 的应用处理器的设备，支持的设备最小内存为 4MiB，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。

说明

如上几种基础系统中所支持的最小内存，字节的次方单位采用 IEC 60027-2 标准中定义的二进制前缀。

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。

1.2 OpenHarmony 兼容性目标

鉴于 OpenHarmony 组件可以按需拼装，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
  a. 单设备功能可用：
     i. 单设备功能/特性具有延续性；
     ii. 合法渠道安装的应用功能可用。
  b. 设备组合功能可用：
     i. 基于使用场景，可以通过组合相关设备完成场景功能；
     ii. 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；
     iii. 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；
     iv. 设备加入/退出已有场景组合时，业务体验可以预期；
     v. 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
  a. 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。
  b. 设备开发者：
     i. 用于指导集成 OpenHarmony 系统的统一的软硬件设计规范；
     ii. 明确的设备认证/发布要求与指导；
     iii. 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 开源项目的源代码。开发者可以免费获得并移植到设备。
- OpenHarmony 产品兼容性规范文档（PCS），定义设备兼容性的标准。开发者必须遵循此文档定义的规则进行设备开发。

- OpenHarmony 兼容性测试套件（XTS），提供验证设备兼容性的执行机制。开发者可以借助此套件对兼容性进行评估和验证。

1.3 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，以下简称产品兼容性规范文档）定义了与 OpenHarmony 系统兼容的产品所必须满足的要求，满足本产品兼容性规范文档所定义的要求的产品，才可能被认定为与 OpenHarmony 系统兼容。

本产品兼容性规范文档从 API 兼容性、分布式接口兼容性方面对生态设备的设计进行定义，以促进 OpenHarmony 应用在所有 OpenHarmony 生态设备上兼容性运行和分布式互通。本产品兼容性规范文档还从软硬件基础规格等方面对生态设备的设计进行定义，以保证用户在 OpenHarmony 生态设备上的同一个使用场景的体验一致性。

需要说明的是，本产品兼容性规范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，OpenHarmony 开源代码本身包含了首选的 OpenHarmony 开源实现，设备开发时建议优先选择该首选的 OpenHarmony 开源实现，如果要修改开源实现获得 OpenHarmony 兼容的实现，请下载相关的开源代码并详细阅读本产品兼容性规范中相应的兼容性规则并遵循其规则。

2 介绍

用词约定

本文中出现的“必须（MUST）”、“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）”、“可选（OPTIONAL）”和“禁止（MUST NOT）”用词遵循 RFC2119 中定义的 IETF 标准。

文档结构

第 3 节基础系统兼容性定义特定系统类型的兼容性规范，第 4 节可选系统能力兼容性定义可选系统能力的兼容性规范，第 5 节 硬件兼容性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）。
- 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，设备选择集成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，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规范。
- SOFTWARE：适用于软件兼容性的规范。
- UPDATE：适用于设备与 APP 升级兼容性的规范。
- DISTRIBUTED：适用于分布式兼容性的规范。

- PERFORMANCE：适用于性能兼容性的规范。
- POWER：适用于功耗兼容性的规范。
- SECURITY：适用于安全兼容性的规范。
- MEDIA：适用于多媒体兼容性的规范。
- TOOLS：关于开发工具与选项兼容性的规范。
- CERTIFICATION：关于兼容性测试的规范。
- DFX：Design for X，关于可靠性、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规范。

强制规范定义

对每个“必须（MUST）”规范定义的格式：

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】)。

3 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

3.2 小型系统（small system）兼容性

3.3 标准系统（standard system）兼容性

3.1 轻量系统（mini system）兼容性

3.1.1 硬件

3.1.1.1 CPU

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

3.1.1.2 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。

3.1.1.3 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】设备整体内存空间≥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。

3.1.1.4 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。

3.1.1.5 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。

3.1.1.6 音频

如果设备支持音频输入或输出，必须遵循 4.7 音频定义的硬件兼容性规范。

3.1.1.7 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。

3.1.1.8 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。

3.1.2 软件

3.1.2.1 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：

表 3-1 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 或 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

必选子系统	需要兼容的 API
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	/

说明

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:

表 3-2 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

API 接口	字符范围	名称，说明和要求	举例
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

API 接口	字符范围	名称，说明和要求	举例
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	1. riscv-liteos 2. arma7_hard_neon-vfpv4-liteos 3. arma7_soft-liteos 4. 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，仅有应用生态的系统使用	3
GetFirstApiVersion()	整型	设备首版本的系统软件 API version。 一般是整数，仅有应用生态的系统使用	3

API 接口	字符范围	名称，说明和要求	举例
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

API 接口	字符范围	名称，说明和要求	举例
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

说明

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。

3.1.2.2 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。

3.1.2.3 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。

3.1.2.4 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。

3.1.2.5 Arc UI 框架

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

3.1.3 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。

3.1.4 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。

3.1.5 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。

3.1.6 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。

3.1.7 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。

3.1.8 DFX（Design for X）

【M-DFX-0100】流水日志特指设备运行期间会持续生成的具备流水性质的日志文件，打印流水日志必须使用 hilog 接口，并且按照接口要求定义模块标识，确保日志可追溯。

3.2 小型系统（small system）兼容性

3.2.1 硬件

3.2.1.1 CPU

【S-HARDWARE-0100】具备 MMU。

3.2.1.2 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。

3.2.1.3 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】设备整体内存空间≥4MiB。

【S-HARDWARE-SR-0201】设备整体可读写非易失存储空间≥16MiB。

3.2.1.4 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。

3.2.1.5 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。

3.2.1.6 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。

3.2.1.7 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。

3.2.1.8 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。

3.2.2 软件

3.2.2.1 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：

表 3-3 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

必选子系统	需要兼容的 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	/
用户程序框架	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 foundation/appexecfwk/appexecfwk_lite/interfaces/kits/bundle_lite/*.h

说明

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:

表 3-4 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

API 接口	字符范围	名称，说明和要求	举例
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

API 接口	字符范围	名称，说明和要求	举例
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	1. riscv-liteos 2. arma7_hard_neon-vfpv4-linux 3. arma7_soft-linux 4. arma7_softfp_neon-vfpv4-linux 5. arma7_hard_neon-vfpv4-liteos 6. arma7_soft-liteos 7. 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，仅有应用生态的系统使用	3

API 接口	字符范围	名称，说明和要求	举例
GetFirstApiVersion()	整型	设备首版本的系统软件 API version。 一般是整数，仅有应用生态的系统使用	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

API 接口	字符范围	名称，说明和要求	举例
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

API 接口	字符范围	名称，说明和要求	举例
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

说明

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。

3.2.2.2 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。

3.2.2.3 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。

3.2.2.4 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。

3.2.2.5 Arc 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 Arc UI 框架定义的兼容性规范。

3.2.3 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。

3.2.4 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。

3.2.5 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。

3.2.6 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。

3.2.7 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。

3.2.8 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。

3.3 标准系统（standard system）兼容性

3.3.1 硬件

3.3.1.1 CPU

【STD-HARDWARE-SR-0100】建议支持图形处理 GPU。

3.3.1.2 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。

3.3.1.3 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】建议设备整体内存空间≥128MiB。

【STD-HARDWARE-SR-0201】建议设备整体可读写非易失存储空间≥2GiB。

3.3.1.4 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。

3.3.1.5 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。

3.3.1.6 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。

3.3.1.7 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。

3.3.1.8 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。

3.3.2 软件

3.3.2.1 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：

表 3-5 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

必选子系统	需要兼容的 API
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.DISTRIBUTEDdaccount
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.DISTRIBUTEDdschedule
用户程序框架	如下 JS 模块定义的 API： @ohos.ability.wantconstant @ohos.ability.featureability @ohos.bundle @ohos.app.abilitymanager
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

说明

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：

表 3-6 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

API 接口	字符范围	名称，说明和要求	举例
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
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

API 接口	字符范围	名称，说明和要求	举例
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	1. arma7_hard_neon-vfpv4-linux 2. arma7_soft-linux 3. 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，仅有应用生态的系统使用	3
getFirstApiVersion()	整型	设备首版本的系统软件 API version。 一般是整数，仅有应用生态的系统使用	3

API 接口	字符范围	名称，说明和要求	举例
getIncremental Version()	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

API 接口	字符范围	名称，说明和要求	举例
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

说明

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。

3.3.2.2 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。

3.3.2.3 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 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。

说明

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。

3.3.2.4 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。

3.3.2.5 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 Arc UI 框架定义的兼容性规范。

3.3.3 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。

3.3.4 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。

3.3.5 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。

3.3.6 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。

3.3.7 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。

3.3.8 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。

4 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。

4.1 图形图像

4.2 电池管理服务

4.3 电源管理服务

4.4 USB 服务

4.5 AI 引擎

4.6 媒体

4.7 音频

4.8 相机

4.9 图像编解码

4.10 电路域电话短信服务

4.11 通话管理服务

4.12 输入法服务框架

4.13 定时&时间时区服务

4.14 密钥和凭据

4.15 应用权限管理

4.16 应用完整性校验

4.17 多模输入服务

4.18 Ability assistant 工具

4.1 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。

4.2 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。

4.3 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。

4.4 USB 服务

USB device mode

如果设备实现支持 USB device 模式：

【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。

4.5 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。

4.6 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。

4.7 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。

4.8 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。

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。

4.9 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，支持静态和动态 GIF，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-0100】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。

4.10 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。

说明

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。

4.11 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。

说明

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。

4.12 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】必须预安装英文输入法应用。

4.13 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。

说明

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。

4.14 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-SR-1500】强烈推荐支持硬件 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。

4.15 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)。

说明

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（分布式业务发起方）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。

4.16 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。

4.17 多模输入服务

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

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

4.18 Ability assistant 工具

【C-ALL-SOFTWARE-1900】必须遵守 aa 工具命令行语义。

5 硬件兼容性

5.1 内存和存储

5.1 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。

6 软件兼容性

6.1 API 兼容性

6.2 HDI 兼容性

6.3 运行时兼容性

6.4 应用兼容性

6.5 应用包格式

6.6 Arc UI 框架

6.1 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。

6.2 HDI 兼容性

【G-SOFTWARE-0200】必须按照“驱动开发指南”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，不允许未实现的 HDI 返回异常。

6.3 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。

说明

ES6 以下几个特性明确不支持：

1. with 语句
2. eval
3. Function

【G-SOFTWARE-0302】JS 运行时环境必须运行在严格模式。

6.4 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。

6.5 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】禁止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，禁止在没有用户确认的情况下静默卸载应用。

6.6 Arc 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。

7 分布式兼容性

7.1 分布式硬件

7.2 分布式软总线

7.3 分布式数据管理

7.4 分布式任务调度

7.1 分布式硬件

【G-DISTRIBUTED-0100】禁止修改设备 UDID 生成规则。

7.2 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】必须支持 IPV4 协议栈,必须提供 Linux BSD Socket API 或者 LWIP BSD Socket API 之一。

【G-HARDWARE-0201】必须支持 DHCPv4/Client。

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-SR-0700】推荐支持 BLE（Bluetooth Low Energy）。

【G-HARDWARE-SR-0701】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-DISTRIBUTED-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-DISTRIBUTED-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-DISTRIBUTED-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-DISTRIBUTED-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-DISTRIBUTED-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-DISTRIBUTED-0205】禁止修改 OpenHarmony 分布式软总线设备间传输通道的管理协议。

说明

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-DISTRIBUTED-0206】禁止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-DISTRIBUTED-0207】禁止修改设备 UUID、NetworkId 生成规则。

【G-DISTRIBUTED-0208】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-DISTRIBUTED-0209】禁止修改软总线默认使用的 BLE 广播的 UUID（0xFDEE），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-DISTRIBUTED-0210】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-DISTRIBUTED-0211】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-DISTRIBUTED-0212】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-DISTRIBUTED-0213】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。不应将 NetworkId 作为设备的长期的标识；不应修改 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-DISTRIBUTED-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。

7.3 分布式数据管理

【G-DISTRIBUTED-0300】禁止修改分布式数据服务中用于设备间数据同步的协议帧、包结构以及加解密算法。

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

说明

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

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

7.4 分布式任务调度

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

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

【G-DISTRIBUTED-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。

8 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。

9 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。

10 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 工具链 API 与命令的兼容不发生改变。

开发选项

【G-TOOLS-0107】必须支持进入 OpenHarmony 调试模式。

【G-TOOLS-0108】必须支持 OpenHarmony 调试配置。

11 兼容性测试套件

【G-CERTIFICATION-0100】必须通过 OpenHarmony 要求的兼容性测试。

【G-CERTIFICATION-0101】必须使用配套 OpenHarmony 发布的最新兼容性测试套件进行测试。

说明

兼容性测试以 OpenHarmony 官网兼容性测试专区发布的最新测试要求和测试套件为准。

12 修订记录

2021/12/31：OpenHarmony 3.0 产品兼容性规范首次发布。