

云手机应用（ KooPhone ）

SDK 参考

文档版本	01
发布日期	2026-09-14



版权所有 © 华为云计算技术有限公司 2026。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址：贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址：<https://www.huaweicloud.com/>

目 录

1 KooPhone 端侧 SDK 概述.....	1
2 KooPhone 端侧 SDK 功能矩阵.....	2
3 KooPhone Android SDK 开放接口.....	10
3.1 基本功能模块.....	10
3.1.1 串流控制接口.....	10
3.1.1.1 启动云手机串流.....	10
3.1.1.2 停止云手机串流.....	13
3.1.1.3 重新发起串流.....	14
3.1.1.4 切换流类型.....	14
3.1.2 基础配置管理接口.....	15
3.1.2.1 设置日志文件夹路径.....	15
3.1.2.2 注册基础回调接口.....	15
3.1.2.3 取消注册基础回调接口.....	16
3.1.3 高级配置管理接口.....	17
3.1.3.1 配置剪贴板功能.....	17
3.1.4 配置查询方法.....	17
3.1.4.1 获取 SDK 版本号.....	17
3.1.4.2 检查云机是否在串流状态.....	18
3.2 视频功能模块.....	18
3.2.1 画面控制接口.....	18
3.2.1.1 设置云机横竖屏方向.....	18
3.2.1.2 动态设置云机画面比例.....	19
3.2.1.3 调整云手机亮度.....	20
3.2.2 视频配置管理接口.....	20
3.2.2.1 设置质量偏好.....	20
3.2.2.2 设置画质.....	22
3.2.2.2.1 设置画质方法 1.....	22
3.2.2.2.2 设置画质方法 2.....	23
3.2.2.3 启用或禁用画面锐化滤镜.....	24
3.2.2.4 视频配置查询接口.....	25
3.2.2.5 设置云侧视频码率配置.....	25
3.3 音频功能模块.....	26

3.3.1 音频控制接口.....	26
3.3.1.1 设置静音状态.....	26
3.3.1.2 设置云机音量.....	26
3.3.1.3 通知云机音频焦点状态变化.....	27
3.3.1.4 设置播放器当前音频流的音量大小.....	28
3.3.1.5 设置是否绕过变速直通.....	28
3.3.2 音频配置管理接口.....	29
3.3.2.1 启动音频焦点监听.....	29
3.3.2.2 启用/禁用音频保持.....	29
3.4 端云指令模块.....	30
3.4.1 云操作系统管理.....	30
3.4.1.1 获取磁盘使用情况.....	30
3.4.1.2 获取媒体元数据.....	32
3.4.1.3 获取播放状态.....	33
3.4.1.4 切换夜间模式.....	35
3.4.1.5 恢复音乐播放.....	35
3.4.1.6 暂停音乐播放.....	36
3.4.2 云应用管理.....	37
3.4.2.1 获取云机上已安装的应用列表.....	37
3.4.2.2 获取云机上正在运行的应用列表.....	40
3.4.2.3 启动应用.....	43
3.4.2.4 关闭应用.....	44
3.4.2.5 卸载应用.....	46
3.4.2.6 加载应用.....	47
3.4.3 消息/事件/指令发送.....	48
3.4.3.1 发送自定义消息到云端 App.....	48
3.4.3.2 发送按键事件.....	49
3.4.3.3 发送手柄摇杆事件.....	49
3.4.3.4 发送文件.....	50
3.5 回调及监听模块.....	50
3.5.1 基础回调方法.....	51
3.5.1.1 启动流成功回调.....	51
3.5.1.2 第一帧画面回调.....	51
3.5.1.3 会话结束回调.....	52
3.5.1.4 流异常回调.....	52
3.5.1.5 扩展消息回调.....	53
3.5.1.6 弱网络回调.....	53
3.5.1.7 视频配置回调.....	54
3.5.1.8 返回弱网指标回调.....	55
3.5.1.9 网络信息回调.....	55
3.5.2 应用级回调方法（App Level）.....	56
3.5.2.1 应用启动回调.....	56

3.5.2.2 应用关闭回调.....	57
3.5.2.3 获取正在运行的应用回调.....	58
3.5.2.4 应用卸载回调.....	59
3.5.2.5 应用加载回调.....	59
3.5.2.6 获取已安装应用回调.....	60
3.5.2.7 APK 安装状态回调.....	61
3.5.2.8 应用生命周期回调.....	63
3.5.2.9 获取媒体元数据回调.....	64
3.5.2.10 获取播放状态回调.....	65
3.5.2.11 音频监控回调.....	66
3.5.2.12 音频操作监控回调.....	67
3.5.2.13 音乐恢复播放回调.....	69
3.5.2.14 音乐暂停播放回调.....	69
3.5.2.15 音量变化回调.....	70
3.5.2.16 亮度变化回调.....	70
3.5.2.17 动态参数配置的回调.....	71
3.5.3 操作系统级回调方法（System Level）.....	71
3.5.3.1 返回云桌面回调.....	71
3.5.3.2 获取磁盘使用情况回调.....	72
3.5.3.3 云设备音频焦点变化回调.....	73
3.5.3.4 音量控制回调.....	75
3.5.3.5 亮度控制回调.....	76
3.5.4 回调方法注册使用说明.....	78
3.5.4.1 概述.....	78
3.5.4.2 回调注册方法.....	78
3.5.4.2.1 注册基础回调接口.....	78
3.5.4.2.2 注册应用级回调接口.....	79
3.5.4.2.3 注册操作系统级回调接口.....	79
3.5.4.3 回调取消注册方法.....	80
3.5.4.3.1 取消注册基础回调接口.....	80
3.5.4.3.2 取消注册应用级回调接口.....	80
3.5.4.3.3 取消注册操作系统级回调接口.....	81
3.5.4.4 回调注册使用建议及示例.....	81
3.6 常态化体验指标配置模块.....	84
3.6.1 视频动态调整配置参数.....	84
3.6.2 音频动态调整配置参数.....	85
3.6.3 JitterBuffer 配置参数.....	86
3.6.4 补帧精灵配置参数.....	87
3.6.5 视频解码配置.....	88
3.7 附录.....	89
3.7.1 错误码说明.....	89
3.7.1.1 错误码.....	90

3.7.1.2 状态码.....	91
3.7.1.3 网络错误码.....	91
3.7.1.4 错误处理.....	92
3.7.2 调用 Android SDK 开放接口注意事项.....	92
4 KooPhone H5 SDK 开放接口.....	93
4.1 基本功能模块.....	93
4.1.1 串流控制接口.....	93
4.1.1.1 启动串流.....	93
4.1.1.2 终止串流.....	95
4.1.1.3 恢复串流.....	95
4.1.1.4 销毁串流 player 实例.....	96
4.1.2 基础配置管理接口.....	96
4.1.2.1 设置回调.....	96
4.1.2.2 取消回调.....	97
4.1.3 高级配置管理接口.....	97
4.1.3.1 配置定位功能.....	98
4.1.4 配置查询接口.....	98
4.1.4.1 获取 SDK 版本号.....	98
4.1.4.2 检查浏览器是否支持串流.....	99
4.2 视频功能模块.....	99
4.2.1 画面控制接口.....	99
4.2.1.1 设置云机画面分辨率.....	99
4.2.1.2 设置全屏播放.....	100
4.2.1.3 退出全屏播放.....	100
4.2.1.4 设置画中画模式.....	101
4.2.2 视频配置管理接口.....	101
4.2.2.1 设置质量偏好.....	101
4.2.2.2 设置画质.....	102
4.2.3 视频配置查询接口.....	103
4.2.3.1 获取云机当前的 profile 配置列表.....	103
4.2.3.2 获取媒体元素.....	104
4.3 音频功能模块.....	104
4.3.1 音频控制接口.....	104
4.3.1.1 设置静音.....	104
4.3.1.2 设置云机音量.....	105
4.3.1.3 设置串流播放器音量.....	105
4.3.2 音频配置查询接口.....	106
4.3.2.1 获取音量大小.....	106
4.3.2.2 查询云机是否静音.....	106
4.4 端云指令模块.....	107
4.4.1 消息发送接口.....	107
4.4.1.1 发送手柄按键.....	107

4.4.1.2 上传文件.....	109
4.4.1.3 设置禁止传输的键盘按键事件.....	110
4.4.1.4 发送虚拟按键.....	110
4.4.1.5 剪切板同步.....	111
4.4.1.6 发送摇杆轴事件.....	111
4.5 回调及监听模块.....	112
4.5.1 基础回调方法.....	113
4.5.1.1 onClose.....	113
4.5.1.2 onError.....	115
4.5.1.3 onBadNetwork.....	116
4.5.1.4 onStatistics.....	116
4.5.1.5 onProfileChanged.....	117
4.5.1.6 onOrientation.....	118
4.5.1.7 onStateChange.....	119
4.5.2 应用级回调方法.....	121
4.5.2.1 onCloudVolumeChange.....	121
4.5.2.2 onAppMessage.....	122
4.5.2.3 onAutoplayFailed.....	122
4.5.3 监听事件介绍.....	122
4.6 附录.....	124
4.6.1 错误码说明.....	124

1 KooPhone 端侧 SDK 概述

KooPhone端侧SDK是为开发者提供的云手机能力集成工具包，包含Android原生SDK（AAR）和H5 Web SDK（JS）两种形式，使开发者能够快速在移动应用或Web应用中接入云手机的核心功能。通过标准化接口，开发者可轻松实现云手机连接、音视频控制、指令交互等全场景能力。

核心价值

- 多平台支持
 - Android SDK：以AAR包形式提供，适用于原生Android应用集成。
 - H5 SDK：以JavaScript包形式提供，适用于Web应用/混合开发场景。

- 能力模块化

模块类型	功能描述
基本功能模块	包含云手机连接管理、生命周期控制、基础配置。
视频功能模块	包含画质调节、分辨率控制、截屏、视频流解码配置。
音频功能模块	包含音量控制、音频轨道管理、静音切换、采样率配置。
端云指令模块	包含云端设备控制、应用管理、远程命令执行、消息传递。
回调监听模块	包含实时状态通知（连接事件、画质变化、权限请求等）。

- 开箱即用
 - 预封装云手机网络通信协议。
 - 内置重连机制与异常处理。
 - 提供完整的生命周期管理接口。

2 KooPhone 端侧 SDK 功能矩阵

表 2-1 基本功能模块

接口类型	功能	Android函数名称	js函数名称	功能描述
串流控制接口	SDK初始化	init	createPlayer	初始化云手机 SDK
	启动云手机串流	start	initMediaStream	启动云手机串流
	停止云手机串流	stop	closeMediaStream	停止云手机串流
	恢复云手机串流	/	resumeMediaStream	恢复云手机串流
	销毁串流实例	/	destroyMediaStream	销毁串流实例
	重新发起串流	reconnect	/	重新连接云机
	切换流类型	switchStreamType	/	在云机上切换流类型
基础配置管理接口	设置日志文件夹路径	setLogFolder	/	设置日志文件夹路径
	设置事件回调	setPlayerCallback	on	设置事件回调
	取消事件回调	unSetPlayerCallback	off	取消事件回调
高级配置管理接口	配置定位功能	/	setGpsStatus	设置定位配置
	配置剪贴板功能	enableClipboard	/	控制云机上的剪贴板功能
配置查询接口	查询SDK版本号	version	version	获取当前SDK版本号

接口类型	功能	Android函数名称	js函数名称	功能描述
	查询云机是否在串流状态	isPlaying	/	检查云机是否在串流状态
	查询浏览器是否支持	/	isBrowserSupport	检查浏览器是否兼容SDK功能

表 2-2 视频功能模块

接口类型	功能	Android函数名称	js函数名称	功能描述
画面控制接口	设置云机横竖屏方向	setDirection	/	强制设置云机的横竖屏方向
	设置云机画面分辨率	setCloudScreenAspect	setCloudScreenAspect	动态设置云机的画面比例
	调整云手机亮度	changeBrightness	/	向云机发送亮度变化指令
	设置全屏播放	/	setMediaFullScreen	设置串流播放器全屏播放
	退出全屏播放	/	setMediaExitFullScreen	退出串流播放器全屏播放
	设置画中画模式	/	setPicInPicStatus	设置画中画模式，打开或者关闭画中画模式
	设置视频相关的动态配置	setCloudVideoConfig	/	设置码率、分辨率、gop大小等
	设置视频解码模式	setDecodeMode	/	设置解码模式（FAST、BALANCE）
	设置云侧视频码率配置	setVideoBitrateConfig	/	动态拉高码率
视频配置管理接口	设置质量偏好	setQualityPreference	setVideoQuality	设置云手机的质量偏好
	画质调节	setProfile/ setResProfile	setResProfile	设置画质
	启用或禁用画面锐化滤镜	enableSharpenFilter	/	启用或禁用画面锐化滤镜

接口类型	功能	Android函数名称	js函数名称	功能描述
视频配置查询接口	获取云机当前的profile配置列表	getProfileList	getProfileList	获取云机当前的profile配置列表
	获取媒体元素	/	getMediaElement	获取当前串流的媒体元素对象

表 2-3 音频功能模块

接口类型	功能	Android函数名称	js函数名称	功能描述
音频控制接口	设置静音	mute	setMuted	设置云机的静音状态
	设置云机音量	changeVolume	setCloudVolume	设置云机的音量等级
	设置串流播放器音量	/	setVolume	设置串流播放器音量
	通知云机音频焦点状态变化	deviceAudioFocusChange	/	通知云机音频焦点状态变化
	设置播放器当前音频流的音量大小	setAudioTrackVolume	/	设置云机的播放器当前音频流的音量大小
	JitterBuffer配置参数	setJitterBuffer	/	设置音视频 jitter上下限相关的动态配置
	设置音频相关的动态配置	setCloudAudioConfig	/	设置音频参数如码率上下限等
	设置是否绕过变速直通	setBypassSoundStretch	/	设置是否绕过变速直通，提升音质
音频配置管理接口	启动音频焦点监听	startAudioFocusChangeListener	/	启动音频焦点监听
	启用/禁用音频保持	enableAudioKeeping	/	启用或禁用音频保持功能
音频配置查询接口	获取音量大小	/	getVolume	获取音量大小

接口类型	功能	Android函数名称	js函数名称	功能描述
	查询云机是否静音	/	isMuted	查询云机串流播放器是否静音

表 2-4 端云指令模块

接口类型	功能	Android函数名称	js函数名称	功能描述
云操作系统管理接口	获取云机磁盘使用情况	getDiskUsage	/	用于查询云机的磁盘使用情况
	获取媒体元数据	getMetaDataMedia	/	查询云机的媒体元数据信息
	获取播放状态	getPlayBackState	/	查询云机的播放状态
	切换夜间模式	nightModeChange	/	在云机上启用或禁用夜间模式
	恢复音乐播放	resumeMusic	/	在云机上恢复音乐播放
	暂停音乐播放	pauseMusic	/	在云机上暂停音乐播放
应用管理接口	获取已安装应用列表	getInstalledApps	/	用于查询云机上已安装的应用程序
	获取正在运行应用列表	getRunningApps	/	获取云机上正在运行的应用列表
	启动云机应用	startApp	/	在云机上启动指定的应用程序
	关闭云机应用	closeApp	/	在云机上关闭指定的应用程序
	卸载云机应用	uninstallApp	/	在云机上卸载指定的应用程序

接口类型	功能	Android函数名称	js函数名称	功能描述
	加载云机应用	loadApp	/	在云机上加载指定的应用程序
消息发送接口	发送消息到云手机	sendExtMessageToCloud	/	用于以手机广播形式，向云机系统/云端APP发送自定义消息
	发送按键事件	sendKeyEvent	sendKeyEvent	向云机发送按键事件
	发送手柄摇杆事件	sendJoyStickEvent	sendAxesEvent	用于在云机上模拟手柄摇杆操作
	发送文件	sendFile	sendFile	用于将本地文件发送到云机
	发送虚拟按键	/	sendVirtualButtonEvent	发送虚拟按键（针对JS场景下，无物理注入的部分按键，如home、back等）
	剪切板同步	/	paste	将文本发送至云机剪切板

表 2-5 回调及监听功能模块

接口类型	功能	Android函数名称	js函数名称	功能描述
基础回调接口	启动流成功回调	onSuccess	/	云手机连接成功后触发
	第一帧画面回调	onFirstFrame	/	云机画面首次渲染完成时触发
	会话结束回调	/	onClose	云手机会话结束时触发
	流异常回调	onFailure	onError	云手机连接或运行异常时触发

接口类型	功能	Android函数名称	js函数名称	功能描述
	扩展消息回调	onExtMessageReceived	/	收到扩展消息时触发
	弱网络回调	onWeakNetwork	onBadNetwork	检测到网络质量变差时触发
	串流视频信息回调	/	onStatistics	默认每秒触发1次，反馈当前串流分辨率，帧率，码率等信息
	串流状态回调	/	onStateChange	建立串流、串流过程中连接的状态变化回调。比如请求鉴权，鉴权成功，建联成功等
	云机画质变化回调	/	onProfileChanged	串流过程中，由于主动切换视频质量或网络状态变化自动切换视频质量会触发此回调
	云机屏幕方向改变回调	/	onOrientation	云机屏幕方向发生变化时触发
	云机视频画质配置回调	onProfileList	/	主动调用getProfileList()时触发，回调视频画质配置列表
	网络统计指标回调	onCloudStatistics	/	显示相关网络数据
应用级回调方法	应用启动回调	onStartApp	/	应用启动操作完成时触发
	应用关闭回调	onCloseApp	/	应用关闭操作完成时触发
	获取正在运行的应用回调	onGetRunningApps	/	获取正在运行的应用信息完成时触发
	应用卸载回调	onUninstallApp	/	应用卸载操作完成时触发

接口类型	功能	Android函数名称	js函数名称	功能描述
	应用加载回调	onLoadApp	/	应用加载操作完成时触发
	获取已安装应用回调	onGetInstalledApps	/	获取已安装应用信息完成时触发
	APK安装状态回调	onAPKInstallState	/	APK安装状态更新时触发
	应用生命周期回调	onAppLifecycle	/	应用生命周期状态更新时触发
	获取媒体元数据回调	onGetMediaMetadata	/	获取媒体元数据完成时触发
	获取播放状态回调	onGetPlaybackState	/	获取播放状态完成时触发
	音频监控回调	onMonitorAudio	/	音频监控数据更新时触发
	音频数据变化回调	onMonitorAudioAction	/	音频操作监控数据更新时触发
	音乐恢复播放回调	onResumeMusic	/	音乐恢复播放操作完成时触发
	音乐暂停播放回调	onPauseMusic	/	音乐暂停播放操作完成时触发
	音量变化回调	onChangeVolume	onCloudVolumeChange	音量变化操作完成时触发
	亮度变化回调	onChangeBrightness	/	亮度变化操作完成时触发
	应用广播消息回调	/	onAppMessage	云侧应用下发的广播消息回调
	自动播放失败回调	/	onAutoplayFailed	串流视频自动播放失败回调，通常由于浏览器不支持默认播放导致
	动态参数配置的回调	onDynamicConfigNotify	/	音视频动态参数设置完成时触发

接口类型	功能	Android函数名称	js函数名称	功能描述
操作系统级回调方法	返回云桌面回调	onReturnLaun cher	/	用户操作返回云桌面时触发
	获取磁盘使用情况回调	onGetDiskUsa ge	/	获取磁盘使用情况完成时触发
	云设备音频焦点变化回调	onCloudDevic eAudioFocusC hange	/	云设备音频焦点状态变化时触发
	音量控制回调	onVolumeCon trol	/	音量控制操作完成时触发
	亮度控制回调	onBrightnessC ontrol	/	亮度控制操作完成时触发

3 KooPhone Android SDK 开放接口

3.1 基本功能模块

3.1.1 串流控制接口

3.1.1.1 启动云手机串流

方法名称

boolean start(PlayerFragment fragment, final Bundle bundle)

功能描述

启动云手机串流。该方法用于启动云手机的串流功能，将云机画面和音频传输到本地设备。

参数说明

字段名	类型	说明
fragment	PlayerFragment	播放器Fragment，用于显示云机画面，可以为null
bundle	Bundle	启动参数，包含云机连接信息，如IP地址、端口、用户ID等

以下参数可通过Bundle对象在启动云手机串流时进行自定义配置：

- 基础连接参数（必填）

参数键	类型	说明
LAUNCH_KEY_ADDRESS	String	云机IP地址，必填。由后台返回的服务端地址
LAUNCH_KEY_PORT	int	云机端口号，必填。由后台返回的服务端口，默认值：0
LAUNCH_KEY_TOKEN	String	认证令牌，随机生成，必填。用于云机身份校验，校验失败返回错误码100002
LAUNCH_KEY_USERID	String	用户标识，必填。多云机场景下的用户唯一ID

• 音视频配置参数（选填）

参数键	类型	说明
LAUNCH_KEY_FREE_ASPECT	boolean	自由比例开关，默认值：true true：允许自定义分辨率
LAUNCH_KEY_WIDTH	int	画面宽度（像素），自由比例开启时生效，默认值：1080
LAUNCH_KEY_HEIGHT	int	画面高度（像素），自由比例开启时生效，默认值：1920
LAUNCH_KEY_ASAM_INV	int	音频采样间隔（毫秒），可选值：10或20，默认值：10
LAUNCH_KEY_ASAM_RATE	int	音频采样率，可选值：8000/44100/48000，默认值：48000
LAUNCH_KEY_ACH_COUNT	int	音频声道数，默认值：2 1：单声道 2：双声道
LAUNCH_KEY_ABUF_SIZE	int	音频缓冲区大小，默认值：0 0：自动调整 1：最小缓冲

• 高级配置

参数键	类型	说明
LAUNCH_KEY_VIDEO_PERF	int	性能日志开关，开启后收集视频性能数据，默认值：2

参数键	类型	说明
LAUNCH_KEY_EVENT_EXECUTOR	int	执行器选择，默认值：0 1: Ignition执行器（IGNITION_EXECUTOR），任务会通过Ignition策略调度执行，适用于端侧低算力场景 0: EventLoop执行器（LOOP_EXECUTOR），任务通过传统的EventLoop策略执行，适用于大部分普通场景
LAUNCH_KEY_HIDE_STREAM_AT_START_UP	boolean	启动时串流画面选项，用于控制串流启动时是否显示画面的配置项，通过设置该值为true，可以实现串流启动时隐藏画面，便于开发者进行页面初始化或动画加载等操作，默认值：false
LAUNCH_KEY_PROTOCOL_MODE	int	网络传输协议配置，默认值：0 0: mright协议 1: mlight协议

注意事项

- 标有**必填**的参数必须提供有效值，否则可能导致连接失败。
- 加密参数在公有云部署环境下强制生效。
- 分辨率参数仅在LAUNCH_KEY_FREE_ASPECT=true时生效。
- 通过createBundle(userId, Map)可扩展自定义参数。

Bundle初始化参数配置示例：

```
public Bundle createBundle(String address, int aprot, int atype, String token, String userId, int width,
int height) {
    Bundle bundle = new Bundle();

    // 基础连接参数
    bundle.putString(LAUNCH_KEY_ADDRESS, address); // 云机ip, 后台返回
    bundle.putInt(LAUNCH_KEY_PORT, aprot); // 云机端口, 后台返回
    bundle.putInt(LAUNCH_KEY_TYPE, atype); // 后台返回
    bundle.putString(LAUNCH_KEY_TOKEN, token); // 校验token, 后台返回, 云机校验失败会返回
100002异常码
    bundle.putString(LAUNCH_KEY_USERID, userId); // 多云机共用userid, 可用用户登录
id, 也可自定义

    // 个性化视频配置
    bundle.putBoolean(LAUNCH_KEY_FREE_ASPECT, true); // 开启自由比例
    bundle.putInt(LAUNCH_KEY_WIDTH, width); // 任意分辨率宽
    bundle.putInt(LAUNCH_KEY_HEIGHT, height); // 任意分辨率高

    // 个性化音频配置
    bundle.putInt(LAUNCH_KEY_ASAM_INV, 10); // 音频采样间隔: 10或20
    bundle.putInt(LAUNCH_KEY_ASAM_RATE, 48000); // 音频采样率: 8000、44100、48000
    bundle.putInt(LAUNCH_KEY_ACH_COUNT, 2); // 音频声道数: 1-单声道, 2-双声道
    bundle.putInt(LAUNCH_KEY_ABUF_SIZE, 0); // 端侧音频缓冲大小: 0-自动大小, 1-最小

    // 高级配置
```

```
bundle.putInt(LAUNCH_KEY_VIDEO_PERF, 2);           // 开启性能日志  
  
return bundle;  
}
```

返回值

boolean：成功返回true，失败返回false。

调用示例

```
// 启动自定义串流Activity  
private void startCloudPhoneStream() {  
    // 准备串流参数（这些参数通常从服务器获取）  
    String address = "192.168.1.100"; // 云机IP  
    int aport = 10000;                // 云机端口  
    int atype = 1;                    // 类型  
    String token = "your_token_here"; // 认证token  
    String userId = "user_123";       // 用户ID  
    int width = 1080;                 // 分辨率宽  
    int height = 1920;                // 分辨率高  
  
    // 创建Bundle参数  
    Bundle bundle = createBundle(address, aport, atype, token, userId, width, height);  
  
    // 启动Activity  
    Intent intent = new Intent(this, CustomCloudPhoneActivity.class);  
    intent.putExtras(bundle);  
    startActivity(intent);  
}
```

注意事项：

- 必须先调用init方法，否则SDK未初始化将导致启动失败。
- bundle必须包含必要的连接参数，否则启动失败。
- 建议使用try-catch包裹该方法调用，以捕获可能的异常。

3.1.1.2 停止云手机串流

方法名称

void stop()

功能描述

停止云手机串流，退出云机。该方法用于结束当前的云机串流会话，释放相关资源。

参数说明

无

返回值

无

调用示例

```
CloudPhoneClient.stop();
```

3.1.1.3 重新发起串流

方法名称

`boolean reconnect()`

功能描述

重新连接云机。该方法用于在串流中断后重新连接云机，恢复串流功能。

参数说明

无

返回值

无

调用示例

```
CloudPhoneClient.reconnect();
```

3.1.1.4 切换流类型

方法名称

`void switchStreamType(CloudPhoneConst.CLOUD_APP_SWITCH_STREAM_TYPE type)`

功能描述

切换流类型。该方法用于在云机上切换流类型。

参数说明

字段名	类型	说明
type	CLOUD_APP_SWITCH_STREAM_TYPE	流类型，枚举值： - DATA：仅数据传输通道 - VIDEO：仅拉取视频流，不拉取音频流，包含DATA - AUDIO：仅拉取音频流，不拉取视频流，包含DATA - BOTH：启动后音视频正常拉流，默认值，包含DATA

返回值

无

调用示例

```
CloudPhoneClient.switchStreamType(CloudPhoneConst.CLOUD_APP_SWITCH_STREAM_TYPE.BOTH);
```

3.1.2 基础配置管理接口

3.1.2.1 设置日志文件夹路径

方法名称

```
void setLogFolder(String path)
```

功能描述

设置日志文件夹路径。该方法用于指定SDK日志的存储路径，便于日志管理和问题排查。

参数说明

字段名	类型	说明
path	String	日志文件夹路径

返回值

无

调用示例

```
CloudPhoneClient.setLogFolder("/sdcard/cloudphone_logs");
```

3.1.2.2 注册基础回调接口

方法名称

```
void setPlayerCallback(Callback callback)
```

功能描述

注册基础回调接口。该方法用于注册实现了基础回调方法的接口实例。更多回调接口使用详见[回调及监听模块](#)。

参数说明

字段名	类型	说明
callback	Callback	实现了基础回调方法的接口实例

返回值

无

调用示例

```
CloudPhoneClient.setPlayerCallback(new CloudPhoneClient.Callback() {
    @Override
    public void onSuccess() {
        Log.d("CloudPhone", "连接成功");
    }

    @Override
    public void onTerminated() {
        Log.d("CloudPhone", "会话结束");
    }

    // 其他基础回调方法实现...
});
```

3.1.2.3 取消注册基础回调接口

方法名称

void UnsetPlayerCallback(Callback callback)

功能描述

取消注册基础回调接口。该方法用于取消注册已注册的基础回调接口实例。更多回调接口使用详见[回调及监听模块](#)。

参数说明

字段名	类型	说明
callback	Callback	已注册的基础回调接口实例

返回值

无

调用示例

```
CloudPhoneClient.UnsetPlayerCallback(playerCallback);
```

3.1.3 高级配置管理接口

3.1.3.1 配置剪贴板功能

方法名称

void enableClipBoard(boolean enableClipBoard)

功能描述

启用或禁用剪贴板功能。该方法用于控制云机上的剪贴板功能。

参数说明

字段名	类型	说明
enableClipBoard	boolean	是否配置剪贴板功能 • true：启用 • false：禁用

返回值

无

调用示例

无

3.1.4 配置查询方法

3.1.4.1 获取 SDK 版本号

方法名称

String version()

功能描述

获取当前SDK版本号。该方法用于获取SDK的版本信息，便于调试和版本管理。

参数说明

无

返回值

String：版本信息。

调用示例

```
String version = CloudPhoneClient.version();  
Log.i("SDK版本", version);
```

3.1.4.2 检查云机是否在串流状态

方法名称

boolean isPlaying()

功能描述

检查云机是否在串流状态。该方法用于判断当前云机是否正在串流，便于开发者进行状态管理。

参数说明

无

返回值

boolean：是否正在串流。

调用示例

```
String isPlaying = CloudPhoneClient.isPlaying();
```

3.2 视频功能模块

3.2.1 画面控制接口

3.2.1.1 设置云机横竖屏方向

方法名称

void setDirection(Boolean direction)

功能描述

强制设置云机的横竖屏方向。该方法用于在特定场景下锁定云机的屏幕方向，例如游戏或视频播放时。

参数说明

字段名	类型	说明
direction	Boolean	是否设置云机横竖屏方向 • true：横屏 • false：竖屏

返回值

无

调用示例

```
// 强制横屏
CloudPhoneClient.setDirection(true);
// 强制竖屏
CloudPhoneClient.setDirection(false);
```

3.2.1.2 动态设置云机画面比例

方法名称

void setCloudScreenAspect(int screenWidth, int screenHeight)

功能描述

动态设置云机的画面比例（自定义尺寸）。该方法允许开发者根据实际需求设置云机的分辨率，以适配特定场景。

参数说明

字段名	类型	说明
screenWidth	int	真机屏幕宽
screenHeight	int	真机屏幕高

返回值

无

调用示例

```
CloudPhoneClient.setCloudScreenAspect(1920,1080);
```

3.2.1.3 调整云手机亮度

方法名称

`void changeBrightness(Double brightness)`

功能描述

向云机发送亮度变化指令。该方法用于调整云机屏幕的亮度，以适应不同环境下的显示需求。

参数说明

字段名	类型	说明
brightness	Double	亮度值，范围0.0 ~ 1.0

返回值

无

调用示例

```
CloudPhoneClient.changeBrightness(0.5); // 亮度50%
```

3.2.2 视频配置管理接口

3.2.2.1 设置质量偏好

方法名称

`void setQualityPreference(int level)`

功能描述

设置云手机的质量偏好。该方法用于在画质和性能之间进行权衡，适用于不同网络环境下的自适应调整。

参数说明

字段名	类型	说明
level	int	质量偏好等级 0：均衡模式 解码器：fast 1：画质优先模式 解码器：balance 2：流畅优先模式 解码器：fast 3：低流量模式 解码器：balance 4：低流量模式 解码器：fast 5：均衡模式 解码器：fast

返回值

无

调用示例

无

📖 说明

设置质量偏好setQualityPreference（level：[0,5]区间值）接口

使用规则：

音乐类应用推荐使用 level 4 低流量模式&FAST解码，降低播放音乐的前台视频流量消耗。

视频类应用推荐使用 level 0 均衡模式&BALANCE解码，画面缓冲实现平滑播放确保极致流畅。

其他类应用推荐使用 level 5 均衡模式&FAST解码。

说明：

setQualityPreference和setDecodeMode有覆盖关系。若使用了setQualityPreference，不需要再使用setDecodeMode。

0-5的枚举值描述：

```
/**
 * 画质偏好模式：画质优先模式解码器为balance
 */
public static final int QUALITY_PRIORITY_MODE = 1;
/**
 * 画质偏好模式：均衡模式解码器为balance
 */
public static final int BALANCED_MODE = 0;
/**
 * 画质偏好模式：流畅优先模式解码器为fast
 */
public static final int SMOOTHNESS_PRIORITY_MODE = 2;
/**
 * 画质偏好模式：低流量模式解码器为balance
 */
public static final int LOW_FLOWRATE_MODE = 3;
/**
 * 画质偏好模式：低流量模式解码器为fast
 */
public static final int LOW_FLOWRATE_MODE_FAST_DECODE = 4;
/**
 * 画质偏好模式：均衡模式解码器为fast
 */
public static final int BALANCED_MODE_FAST_DECODE = 5;
```

3.2.2.2 设置画质

3.2.2.2.1 设置画质方法 1

方法名称

boolean setProfile(int level)

功能描述

设置云手机的画质。

参数说明

字段名	类型	说明
level	int	画质档位 0：自动档 1：流畅档 2：均衡档 3：高清档

返回值

boolean，设置成功返回true，设置失败返回false

调用示例

```
CloudPhoneClient.setProfile(3);
```

3.2.2.2.2 设置画质方法 2

方法名称

void setResProfile(Profile profile)

功能描述

设置云手机的画质。Profile 内参数的优先级：level > width > height.

1. 如果设置了level，按level匹配处理云机画质档位；
2. 如果没有设置level（值为null），且设置了width，按width距离计算就近的云机画质档位；
3. 如果没有设置level（值为null），width（值为null），且设置了height，按height距离计算就近的云机画质档位。

 **说明**

width和height与分辨率的对应关系：例如1080p画质档位，分辨率为1080x1920，此时width对应于1920，height对应于1080。

另外，参数设置匹配时，还需考虑端侧硬件支持的档位限制。

参数说明

Profile 对象结构：

字段名	类型	说明
level	Integer	画质档位 0：自动档 1：流畅档 2：均衡档 3：高清档
width	Integer	分辨率宽度
height	Integer	分辨率高度

返回值

boolean，设置成功返回true，设置失败返回false。

调用示例

```
Profile profile = new Profile();
profile.setHeight(1080); // 设置1080p分辨率
CloudPhoneClient.setResProfile(profile);
```

3.2.2.3 启用或禁用画面锐化滤镜

方法名称

boolean enableSharpenFilter(boolean enable)

功能描述

启用或者禁用画面锐化滤镜（默认为禁用）。该方法用于在需要时对画面进行锐化处理，适用于串流文字阅读类APP等场景。

参数说明

字段名	类型	说明
enable	boolean	启用或者禁用画面锐化滤镜 - true：启用画面锐化滤镜 - false：禁用画面锐化滤镜

返回值

boolean，设置成功返回true，设置失败返回false

调用示例

```
CloudPhoneClient.enableSharpenFilter(true);
```

3.2.2.4 视频配置查询接口

方法名称

```
void getProfileList()
```

功能描述

获取视频质量配置列表（结果通过onProfileList接口回调）。

参数说明

无

返回值

无

调用示例

```
CloudPhoneClient.getProfileList();
```

3.2.2.5 设置云侧视频码率配置

方法名称

```
boolean setVideoBitrateConfig(boolean enable, float ratio)
```

功能描述

该方法用于触屏后动态拉高码率，提升滑动模糊恢复速度；指定时间间隔后自动回落，节省流量成本。

参数说明

字段名	类型	说明
enable	boolean	是否开启视频码率调整功能 true: 启用 false: 禁用
ratio	float	码率上调比例（百分比，例如 1.1 表示上调10%，上调范围1.0-3.0）

返回值

int, 执行成功返回0，执行失败返回1

调用示例

```
CloudPhoneClient.setVideoBitrateConfig(true,1.1f);
```

3.3 音频功能模块

3.3.1 音频控制接口

3.3.1.1 设置静音状态

方法名称

void mute(boolean mute)

功能描述

设置云机的静音状态。该方法用于在需要时静音云机的音频输出，例如在会议或视频播放时。

参数说明

字段名	类型	说明
mute	boolean	云机的静音状态 • true：静音 • false：取消静音

返回值

无

调用示例

```
// 静音
CloudPhoneClient.mute(true);
// 取消静音
CloudPhoneClient.mute(false);
```

3.3.1.2 设置云机音量

方法名称

void changeVolume(MediaVolume volume)

功能描述

向云机发送音量变化指令。该方法用于远程调整云机的音量，适用于需要远程控制的情景。

参数说明

字段名	类型	说明
volume	MediaVolume	音量对象，包含音量值

返回值

无

调用示例

```
MediaVolume mediaVolume = new MediaVolume(0.7f); // 音量70%
CloudPhoneClient.changeVolume(mediaVolume);
```

3.3.1.3 通知云机音频焦点状态变化

方法名称

void deviceAudioFocusChange(String state)

功能描述

通知云机音频焦点状态变化。该方法用于在设备音频焦点变化时通知云机，以调整音频播放行为。

参数说明

参数名	类型	说明
deviceAudioFocusChangeState	String	传入云侧操控助手音频焦点状态变化类型，可选值： - AUDIOFOCUS_GAIN：获取音频焦点 - AUDIOFOCUS_LOSS：永久失去音频焦点 - AUDIOFOCUS_TRANSIENT：暂时失去音频焦点 - UDIOFOCUS_TRANSIENT_CAN_DUCK：暂时失去音频焦点但可以继续播放（音量降低） 当云侧操控助手获得云侧音频时，云侧第三方音乐会失去焦点，暂停播放；当云侧操控助手失去云侧音频时，操控助手会主动拉起第三方音乐并让其获得音频焦点，此时播放音乐。

返回值

无

调用示例

```
CloudPhoneClient.deviceAudioFocusChange("FOCUSED");
```

3.3.1.4 设置播放器当前音频流的音量大小

方法名称

```
void setAudioTrackVolume(float volume)
```

功能描述

该方法用于调整端侧播放器音量。

参数说明

参数名	类型	说明
volume	float	传入音量大小，可选值： 0.0：静音 1.0：标准原始音量 0.0 ~ 1.0：线性音量调节区间

返回值

无

调用示例

```
CloudPhoneClientInternal.setAudioTrackVolume(volume);
```

3.3.1.5 设置是否绕过变速直通

方法名称

```
void setBypassSoundStretch(boolean bypass)
```

功能描述

设置是否绕过音频播放变速直通（默认为true）。该方法用于提升音质场景，遇到卡顿严重时恢复正常变速补偿。

参数说明

字段名	类型	说明
bypass	boolean	是否绕过变速原样直通（音质无损） • true：绕过变速原样透传 • false：恢复正常变速补偿

返回值

无

调用示例

```
CloudPhoneClient.setBypassSoundStretch(false);
```

3.3.2 音频配置管理接口

3.3.2.1 启动音频焦点监听

方法名称

void startAudioFocusChangeListener()

功能描述

启动音频焦点监听。该方法用于监听设备音频焦点的变化，以便及时调整云机的音频播放行为。

参数说明

无

返回值

无

调用示例

```
CloudPhoneClient.startAudioFocusChangeListener();
```

3.3.2.2 启用/禁用音频保持

方法名称

void enableAudioKeeping(boolean isKeep)

功能描述

启用或禁用音频保持功能。该方法用于在云机断开连接时保持音频播放，适用于需要连续播放的场景。

参数说明

字段名	类型	说明
isKeep	boolean	启用或禁用音频保持 - true：启用音频保持 - false：禁用音频保持

返回值

无

调用示例

```
CloudPhoneClient.enableAudioKeeping(true);
```

3.4 端云指令模块

3.4.1 云操作系统管理

3.4.1.1 获取磁盘使用情况

方法名称

void getDiskUsage()

功能描述

获取云机磁盘使用情况。该方法用于查询云机的磁盘使用情况。

参数说明

无

返回值

无

回调参数

- AppOperateResponse结构

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
data	DiskUsageInfo	磁盘使用数据对象

- DiskUsageInfo结构

字段	类型	计算关系	说明
totalSize	long	-	磁盘物理总容量，单位：字节
usedSize	long	usedSize=totalSize-freeSize	已占用空间（含系统保留），单位：字节
freeSize	long	-	用户可用空间，单位：字节

调用示例

```
// 1. 实现带单位转换的回调处理器
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetDiskUsage(AppOperateResponse<DiskUsageInfo> response) {
        if(!"0".equals(response.getError_code())){
            System.err.println(" 查询失败: " + response.getError_msg());
            return;
        }
        DiskUsageInfo usage = response.getData();
        System.out.printf("磁盘使用情况: \n" +
            " 总容量: %.2f GB\n" +
            " 已使用: %.2f GB (%.1f%%)\n" +
            " 可用空间: %.2f GB\n",
            bytesToGB(usage.getTotalSize()),
            bytesToGB(usage.getUsedSize()),
            (usage.getUsedSize() * 100.0 / usage.getTotalSize()),
            bytesToGB(usage.getFreeSize()));
    }
    private double bytesToGB(long bytes) {
        return bytes / (1024.0 * 1024 * 1024);
    }
};
// 2. 发起查询
CloudPhoneClient.getDiskUsage();
```

响应示例

```
// 成功响应
{
  "error_code": "0",
  "error_msg": "success",
  "data": {
    "totalSize": 53687091200,
    "usedSize": 21474836480,
    "freeSize": 32212254720
  }
}
```

```
// 异常响应
{
  "error_code": "500",
  "error_msg": "Storage service unavailable"
}
```

3.4.1.2 获取媒体元数据

方法名称

void getMediaMetaData()

功能描述

获取云机媒体元数据。该方法用于查询云机的媒体元数据信息。

参数说明

无

返回值

无

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
result_data	JsonObject	媒体信息

MediaMetaData对象，包含以下字段：

字段名	类型	说明
packageName	String	提供媒体的应用包名
isPlaying	Boolean	媒体是否正在播放 • true：播放中 • false：暂停或停止
metadataKeyTitle	String	歌曲名称
metadataKeyArtist	String	歌手名称
metadataKeyDuration	Integer	歌曲时长，单位：毫秒
metadataKeyAlbum	String	专辑名称
metadataKeyAlbumArtist	String	专辑歌手

字段名	类型	说明
metadataKeyAlbumCover	String	专辑封面，通常为图片URL或Base64编码字符串

调用示例

```
// 调用接口获取媒体元数据
CloudPhoneClient.getInstance().getMediaMetadata();

// 设置回调监听
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetMediaMetadata(AppOperateResponse<MediaMetaData> response) {
        if (response.isSuccess() && response.getData() != null) {
            MediaMetaData metaData = response.getData();
            // 处理获取到的媒体元数据
            String title = metaData.getMetadataKeyTitle();
            String artist = metaData.getMetadataKeyArtist();
            // ...其他字段处理
        } else {
            // 处理获取失败情况
            String errorMsg = response.getErrorMsg();
        }
    }
});

// 其他回调方法...
});
```

响应示例

```
response.getData()
{
    "packageName": "com.example.musicplayer",
    "isPlaying": true,
    "metadataKeyTitle": "Hello",
    "metadataKeyArtist": "Adele",
    "metadataKeyDuration": 215000,
    "metadataKeyAlbum": "25",
    "metadataKeyAlbumArtist": "Adele",
    "metadataKeyAlbumCover": "https://example.com/covers/adele_25.jpg"
}
```

3.4.1.3 获取播放状态

方法名称

void getPlayBackState()

功能描述

获取云机播放状态。该方法用于查询云机的播放状态。

参数说明

无

返回值

无

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
result_data	JsonObject	媒体状态信息

MediaPlayState对象，包含以下字段：

字段名	类型	说明
packageName	String	提供媒体播放服务的应用包名
isPlaying	Boolean	媒体是否正在播放 - true：播放中 - false：暂停或停止
title	String	当前播放的媒体标题，通常为歌曲名或视频标题
currentPosition	Integer	当前播放位置，从媒体开始处计算，单位：毫秒

调用示例

```
// 调用接口获取媒体播放状态
CloudPhoneClient.getInstance().getPlayBackState();

// 设置回调监听
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetPlayBackState(AppOperateResponse<MediaPlayState> response) {
        if (response.isSuccess() && response.getData() != null) {
            MediaPlayState playState = response.getData();
            // 处理获取到的播放状态信息
            boolean isPlaying = playState.getIsPlaying();
            int currentPos = playState.getCurrentPosition();
            String currentTitle = playState.getTitle();
            // ...其他字段处理
        } else {
            // 处理获取失败情况
            String errorMsg = response.getErrorMsg();
        }
    }
});

// 其他回调方法...
});
```

响应示例

成功获取媒体播放状态时，参数中的response.getData()内容如下：

```
response.getData()  
返回示例：  
{  
  "packageName": "com.example.musicplayer",  
  "isPlaying": true,  
  "title": "Someone Like You",  
  "currentPosition": 45000  
}
```

3.4.1.4 切换夜间模式

方法名称

void nightModeChange(NightModeStateEnum nightModeChangeState)

功能描述

通知云机切换夜间模式。该方法用于在云机上启用或禁用夜间模式。

参数说明

字段名	类型	说明
nightModeChangeState	NightModeStateEnum	夜间模式状态

返回值

无

调用示例

```
CloudPhoneClient.nightModeChange(NightModeStateEnum.ENABLED);
```

3.4.1.5 恢复音乐播放

方法名称

void resumeMusic()

功能描述

恢复云机上的音乐播放。该方法用于在云机上恢复音乐播放。

参数说明

无

返回值

无

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
result_data	boolean	恢复音乐播放是否成功 • true：成功 • false：失败

调用示例

```
CloudPhoneClient.resumeMusic();
```

3.4.1.6 暂停音乐播放

方法名称

```
void pauseMusic()
```

功能描述

暂停云机上的音乐播放。该方法用于在云机上暂停音乐播放。

参数说明

无

返回值

无

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
result_data	JsonObject	暂停音乐播放是否成功 • true：成功 • false：失败

调用示例

```
CloudPhoneClient.pauseMusic();
```

3.4.2 云应用管理

3.4.2.1 获取云机上已安装的应用列表

方法名称

void getInstalledApps(GetAppsReq getAppsReq)

功能描述

获取云机上已安装的应用列表。该方法用于查询云机上已安装的应用程序。

参数说明

字段名	类型	说明
getAppsReq	GetAppsReq	获取应用请求对象

getAppsReq结构

参数名	类型	说明
appType	AppTypeEnum	应用类型筛选，必填。枚举值： - SYSTEM_APP：系统应用 - THIRD_APP：第三方应用 - BOTH：全部
quality	Integer	图标质量参数，选填。取值范围0-100，默认100，建议设为70~85以平衡清晰度与大小
pageNum	Integer	分页页码，必填。从1开始，需配合pageSize使用
pageSize	Integer	分页大小，必填。最大支持20条/页
needIcon	Boolean	是否返回图标，必填。true返回图标数据/URL，false不返回图标相关字段

返回值

无

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
data	GetAppsRsp	磁盘使用数据对象

回调接口通过

CloudPhoneClient.setAppCallback(CloudPhoneClient.Callback.AppCallback callback)注册，回调参数为AppOperateResponse<GetAppsRsp>，其中GetAppsRsp结构如下：

参数名	类型	说明
appList	List	应用列表（分页数据）
totalCount	Integer	符合筛选条件的应用总数
pageNum	Integer	当前分页页码（与请求参数一致）
pageSize	Integer	当前分页大小（与请求参数一致）

CloudAppInfo应用详情结构如下：

参数名	类型	说明
appName	String	应用名称，如“微信”、“设置”
packageName	String	应用包名（唯一标识），示例：com.tencent.mm
icon	String	WebP格式图标原始数据Base64编码，仅当needIcon=true时可能返回，null表示无数据
iconUrl	String	WebP格式图标URL，端侧优先使用此字段，无值时fallback到icon字节数组
isReady	Boolean	应用是否准备就绪（可启动状态） • true：应用可正常使用 • false：未安装完成或异常状态

参数名	类型	说明
versionName	String	应用外部版本号，示例：1.1.1
versionCode	long	应用内部版本号，示例：111
size	long	应用大小，示例：1048

调用示例

```
// 1. 创建请求对象并设置参数
GetAppsReq req = new GetAppsReq();
req.setAppType(AppTypeEnum.THIRD_APP); // 筛选第三方应用
req.setQuality(80); // 图标质量设为80（平衡清晰度与大小）
req.setPageNum(1); // 查询第1页
req.setPageSize(10); // 每页10条数据
req.setNeedIcon(true); // 需要返回图标

// 2. 注册回调接口（建议在初始化时注册一次）
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetInstalledApps(AppOperateResponse<GetAppsRsp> response) {
        if (response.isSuccess()) {
            GetAppsRsp rsp = response.getData();
            List<CloudAppInfo> apps = rsp.getAppList();
            // 处理应用列表数据（如展示名称、包名、图标等）
            for (CloudAppInfo app : apps) {
                String appName = app.getAppName();
                String packageName = app.getPackageName();
                String iconUrl = app.getIconUrl(); // 优先使用URL
                byte[] iconBytes = app.getIcon(); // URL为空时使用字节数组
                boolean isReady = app.getIsReady();
                // ... 业务逻辑处理
            }
        } else {
            // 处理接口调用失败（如网络异常、参数错误等）
            String errorMsg = response.getErrorMsg();
            int errorCode = response.getErrorCode();
        }
    }
});

// 3. 调用接口
CloudPhoneClient.getInstalledApps(req);
```

响应示例

```
{
  "success": true,
  "data": {
    "appList": [
      {
        "appName": "微信",
        "packageName": "com.tencent.mm",
        "icon": null, // 当iconUrl存在时，icon可能为null
        "iconUrl": "https://example.com/icons/wechat.webp",
        "versionName": "xxx",
        "versionCode": 123,
        "size": 1234453,
        "isReady": true
      },
      {
        "appName": "支付宝",
        "packageName": "com.eg.android.AlipayGphone",
        "icon": null,

```

```
"iconUrl": "https://example.com/icons/alipay.webp",
"versionName": "xxx",
"versionCode": 123,
"size": 1234453,
"isReady": true
}
// ... 更多应用（最多10条，与pageSize一致）
],
"totalCount": 25, // 符合条件的第三方应用总数
"pageNum": 1,
"pageSize": 10
},
"errorCode": 0,
"errorMsg": "success"
}
```

3.4.2.2 获取云机上正在运行的应用列表

方法名称

void getRunningApps(GetAppsReq getAppsReq)

功能描述

获取云机上正在运行的应用列表。该方法用于查询当前在云机上运行的应用程序。

参数说明

字段名	类型	说明
getAppsReq	GetAppsReq	获取应用请求对象

getAppsReq结构

字段名	类型	必填	说明
appType	AppTypeEnum	是	应用类型枚举： - SYSTEM_APP：系统应用 - THIRD_APP：第三方应用 - BOTH：全部
quality	Integer	否	图标质量参数，取值范围0-100，默认值为100，建议设为70~85平衡清晰度和大小
pageNum	Integer	是	分页页码，从1开始
pageSize	Integer	是	分页大小，最大20

字段名	类型	必填	说明
needIcon	Boolean	否	是否返回应用图标，true返回icon和iconUrl字段，默认值：false

返回值

无

回调参数

- AppOperateResponse结构

字段	类型	说明
error_code	String	错误代码，见 附录
error_msg	String	错误信息，见 附录
data	GetAppsRsp	应用信息列表，可能为空列表

- GetAppsRsp结构

字段名	类型	说明
appList	List<AppInfo>	应用列表（详见下方AppInfo结构）
totalCount	Integer	应用总数（用于分页）
pageNum	Integer	当前页码
pageSize	Integer	当前分页大小

- AppInfo结构

字段名	类型	说明
appName	String	应用名称
packageName	String	应用包名
icon	String	应用图标原始字节（WebP格式），若needIcon=false或未获取到则为null
iconUrl	String	优先使用图标下载地址（WebP格式），无值则用icon

字段名	类型	说明
isReady	Boolean	应用是否处于可运行状态
snapshot	String	运行应用截图原始字节（WebP格式），部分场景（例如不同意应用协议）可能为空，需要自行补充图片

调用示例

```
// 1. 注册回调
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetInstalledApps(AppOperateResponse<GetAppsRsp> response) {
        if (response.isSuccess()) {s
            GetAppsRsp data = response.getData();
            for (AppInfo app : data.getAppList()) {
                Log.d("APP", "Name: " + app.getAppName() + ", Icon URL: " + app.getIconUrl());
            }
        } else {
            Log.e("APP", "Error: " + response.getErrorMessage());
        }
    }
});

// 2. 构建请求
GetAppsReq req = new GetAppsReq();
req.setAppType(GetAppsReq.AppTypeEnum.BOTH); // 获取所有应用
req.setQuality(80); // 图标质量设为80
req.setPageNum(1); // 第一页
req.setPageSize(10); // 每页10条
req.setNeedIcon(true); // 返回图标

// 3. 调用接口
getRunningApps(req);
```

响应示例

```
{
  "appList": [
    {
      "appName": "Chrome",
      "packageName": "com.android.chrome",
      "iconUrl": "https://example.com/icon_chrome.webp",
      "isReady": true,
      "snapshot": "[WebP字节数据]"
    },
    {
      "appName": "Settings",
      "packageName": "com.android.settings",
      "icon": "[WebP字节数据]",
      "isReady": true,
      "snapshot": "[WebP字节数据]"
    }
  ],
  "totalCount": 42,
  "pageNum": 1,
  "pageSize": 10
}
```

3.4.2.3 启动应用

方法名称

String startApp(String packageName)

功能描述

启动指定包名的应用。该方法用于在云机上启动指定的应用程序。

参数说明

字段名	类型	说明
packageName	String	应用包名

返回值

String：本次startApp指令的请求id。

回调参数

AppOperateResponse结构：

字段	类型	说明
error_code	String	错误码，0表示成功
error_msg	String	错误详情
response_id	String	本次startApp指令的返回id，与调用startApp方法时的请求id一致，用于将请求与返回进行关联
data	String	包名com.tencent.mm，失败返回为空

调用示例

```
// 1. 实现回调接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onStartApp(AppOperateResponse<Void> response) {
        if ("0".equals(response.getError_code())) {
            System.out.println("启动成功");
        } else {
            System.err.println("启动失败: " + response.getError_msg());
        }
    }
};
// 2. 注册回调
CloudPhoneClient.setAppCallback(callback);
// 3. 构造启动参数（通过类名启动）
```

```
AppParams params1 = new AppParams();
params1.setClazz("com.tencent.mm.ui.LauncherUI");
params1.setIntents(new HashMap<String, Object>() {{
    put("scene", "chat");
}});
String reqID = CloudPhoneClient.startApp(params1);
// 或通过Scheme启动
AppParams params2 = new AppParams();
params2.setPackageName("qq.live");
params2.setScheme("weixin://scanqrcode");
String reqID2 = CloudPhoneClient.startApp(params2);
```

响应示例

```
// 成功响应
{
  "error_code": "0",
  "error_msg": "success",
  "response_id": "1778049424514_1"
}
// 失败响应（应用未安装）
{
  "error_code": "404",
  "error_msg": "Target app not installed"
}
```

3.4.2.4 关闭应用

方法名称

String closeApp(String packageName)

功能描述

关闭指定包名的应用。该方法用于在云机上关闭指定的应用程序。

参数说明

参数名	类型	必填	说明
packageName	String	是	目标应用的完整包名，示例： com.tencent.mm

返回值

String：本次closeApp指令的请求id、生成规则时间戳加递增序列。示例：
1768987825628_2。

回调参数

AppOperateResponse结构：

字段	类型	说明
error_code	String	操作结果代码 - 成功示例：0 - 失败示例：见附录
error_msg	String	详细结果说明 - 成功示例：success - 失败示例：见附录
response_id	String	本次closeApp指令的返回id，与调用closeApp方法时的请求id一致，用于将请求与返回进行关联 - 成功示例：1768987825628_2 - 失败示例：失败返回为空
data	String	包名 - 成功示例：com.tencent.mm - 失败示例：失败返回为空

调用示例

```
// 1. 实现回调接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onCloseApp(AppOperateResponse<Void> response) {
        if ("0".equals(response.getError_code())) {
            System.out.println(" 应用关闭成功");
        } else {
            System.err.println(" 关闭失败: " + response.getError_msg());
        }
    }
};
// 2. 注册回调（若未全局注册）
CloudPhoneClient.setAppCallback(callback);
// 3. 关闭目标应用（如微信）
CloudPhoneClient.closeApp("com.tencent.mm");
```

响应示例

```
// 成功响应
{
  "error_code": "0",
  "error_msg": "success"
}
// 失败响应（应用未运行）
{
  "error_code": "404",
  "error_msg": "App not running or already closed"
}
```

3.4.2.5 卸载应用

方法名称

void uninstallApp(String packageName)

功能描述

卸载指定包名的应用。该方法用于在云机上卸载指定的应用程序。

参数说明

参数名	类型	必填	说明
packageName	String	是	应用包名，需符合Android包名规范，示例： com.example.app

返回值

无

回调参数

AppOperateResponse结构：

字段	类型	说明
error_code	String	操作结果代码，0表示成功，其他见 错误码说明
error_msg	String	可读性状态描述，成功返回success
data	String	包名，示例： com.tencent.mm

调用示例

```
// 1. 实现回调处理器
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onUninstallApp(AppOperateResponse<Void> response) {
        switch (response.getError_code()) {
            case "0":
                System.out.println("卸载完成");
                break;
            case "404":
                System.err.println("应用不存在");
                break;
            default:
                System.err.println("错误["+response.getError_code()+"]: "+response.getError_msg());
        }
    }
}
```

```
}  
};  
// 2. 注册回调（建议在初始化阶段完成）  
CloudPhoneClient.setAppCallback(callback);  
// 3. 执行卸载（如卸载测试应用）  
CloudPhoneClient.uninstallApp("com.example.testapp");
```

响应示例

```
// 成功响应（回调可能无法完整送达）  
{  
  "error_code": "0",  
  "error_msg": "Reboot scheduled"  
}  
  
// 失败响应（立即返回）  
{  
  "error_code": "503",  
  "error_msg": "System service unavailable"  
}
```

3.4.2.6 加载应用

方法名称

```
void loadApp(String packageName)
```

功能描述

加载指定包名的应用。该方法用于在云机上加载指定的应用程序。

参数说明

参数名	类型	必填	说明
packageName	String	是	应用包名，需匹配已安装应用的包名，示例： com.tencent.mm

返回值

无

回调参数

字段	类型	说明
error_code	String	加载过程状态码，0表示预加载成功
error_msg	String	详细状态描述，失败时包含错误详情

字段	类型	说明
data	String	包名，示例： com.tencent.mm

调用示例

```
// 1. 实现带重试机制的回调处理器
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onLoadApp(AppOperateResponse<Void> response) {
        System.err.printf(" 加载结果[%s]: %s\n",
            response.getError_code(),
            response.getError_msg());
    }
};
// 2. 注册回调（建议使用单例模式）
CloudPhoneClient.setAppCallback(callback);
// 3. 预加载目标应用（如微信）
CloudPhoneClient.loadApp("com.tencent.mm");
```

响应示例

```
// 成功响应
{
    "error_code": "0",
    "error_msg": "Resource preloaded"
}
// 应用未安装
{
    "error_code": "404",
    "error_msg": "Package not found"
}
```

3.4.3 消息/事件/指令发送

3.4.3.1 发送自定义消息到云端 App

方法名称

void sendExtMessageToCloud(String message)

功能描述

向所有云机广播消息。该方法用于向云端App发送自定义消息，消息内容以JSON格式传递。

参数说明

字段名	类型	说明
message	String	消息体，格式为JSON字符串

```
{
  "funcName": "每个funcName对应一个功能",
  "params": {
    //消息体自定义，端和云侧协商好
  }
}
```

返回值

无

调用示例

```
String message = "{\"type\":\"custom\",\"data\":{\"hello\"}";
CloudPhoneClient.sendExtMessageToCloud(message);
```

3.4.3.2 发送按键事件

方法名称

void sendKeyEvent(int keyCode, int action)

功能描述

向云机发送按键事件。该方法用于向云机发送指定动作的按键事件。

参数说明

字段名	类型	说明
keyCode	int	按键值
action	int	按键动作，可以使用系统定义的KeyEvent常量

返回值

无

调用示例

```
// 发送 Home 键按下事件
CloudPhoneClient.sendKeyEvent(KeyEvent.KEYCODE_HOME, KeyEvent.ACTION_DOWN);
```

3.4.3.3 发送手柄摇杆事件

方法名称

void sendJoyStickEvent(MotionEvent event)

功能描述

向云机发送手柄摇杆事件。该方法用于在云机上模拟手柄摇杆操作。

参数说明

字段名	类型	说明
event	MotionEvent	摇杆事件对象

返回值

无

调用示例

```
MotionEvent event = MotionEvent.obtain(...);
CloudPhoneClient.sendJoypadEvent(event);
```

3.4.3.4 发送文件

方法名称

```
void sendFile(boolean isAsync, String srcLocalFile, String dstRemoteFile, String
jobId, String customMsg)
```

功能描述

发送文件到云机。该方法用于将本地文件发送到云机。

参数说明

字段名	类型	说明
isAsync	boolean	是否异步发送
srcLocalFile	String	本地文件路径
dstRemoteFile	String	远端文件路径
jobId	String	任务ID
customMsg	String	自定义消息

返回值

无

调用示例

无

3.5 回调及监听模块

3.5.1 基础回调方法

3.5.1.1 启动流成功回调

方法名称

void onSuccess()

功能描述

启动流成功回调。该方法在云手机连接成功后触发。

参数说明

无

返回值

无

调用示例

见[回调注册使用建议及示例](#)

3.5.1.2 第一帧画面回调

方法名称

void onFirstFrame()

功能描述

第一帧画面回调。该方法在云机画面首次渲染完成时触发。

参数说明

无

返回值

无

调用示例

```
@Override
public void onFirstFrame() {
    Log.d("CloudPhone", "云机画面首次渲染完成");
    // 可以在此处执行首次画面渲染后的操作
}
```

3.5.1.3 会话结束回调

方法名称

void onTerminated()

功能描述

会话结束回调。该方法在云手机会话结束时触发，可认为是最后一次回调。

参数说明

无

返回值

无

调用示例

见[回调注册使用建议及示例](#)

3.5.1.4 流异常回调

方法名称

void onFailure(int code, String message)

功能描述

流异常回调。该方法在云手机连接或运行过程中出现异常时触发。

参数说明

字段名	类型	说明
code	int	错误码
message	String	错误消息

返回值

无

调用示例

见[回调注册使用建议及示例](#)

3.5.1.5 扩展消息回调

方法名称

void onExtMessageReceived(Activity act, String message)

功能描述

扩展消息回调。该方法在收到扩展消息时触发。

参数说明

字段名	类型	说明
act	Activity	上下文，自定义UI时空
message	String	消息体

返回值

无

调用示例

无

3.5.1.6 弱网络回调

方法名称

void onWeakNetwork()

功能描述

弱网络回调。该方法在检测到网络质量变差时触发。

参数说明

无

返回值

无

调用示例

无

3.5.1.7 视频配置回调

方法名称

void onProfileList(ProfileList profileList)

功能描述

视频配置回调。该方法在主动调用getProfileList()时触发，回调当前的视频画质配置和支持的视频画质配置列表。

参数说明

字段名	类型	说明
profileList	ProfileList	视频配置列表

ProfileList结构

参数名	类型	说明
profileLevel	int	当前的视频配置级别
profiles	List<ProfileInfo>	支持的视频配置列表

ProfileInfo结构

参数名	类型	说明
level	int	视频配置级别（0：自动，1：流畅，2：标清，3：高清）
width	int	画面宽度
height	int	画面高度
fps	int	帧率
bitrate	int[]	码率范围，如 [2000, 4000]，表示 bitrate 根据网络情况在 2000 ~ 4000 范围内波动

返回值

无

调用示例

```
@Override
public void onProfileList(ProfileList profileList) {
    Log.i(TAG, profileList.toString());
}

@Override
public void onProfileList(ProfileList profileList) {
    Log.i(TAG, profileList.toString());
}
```

3.5.1.8 返回弱网指标回调

方法名称

void onCloudStatistics(KPCloudStat kPClientStat)

功能描述

回调接口显示相关网络数据

```
public static native void nativeClassInit();
public int sendFrameRate;    // 发送帧率
public int encFrameRate;    // 编码帧率
public int renderFrameRate; // 显示帧率
public int rtt;              // 往返时延
public int upBitrate;        // 上行流量速度
public int downBitrate;      // 下行流量速度
public int fecRate;          // 纠错率
public int arqRate;          // 重传率
public int estimateBW;       // 估算带宽
public int lostRate;         // 估算丢包
```

重写toString显示参数信息

```
public String toString() {
    return "KPCloudStat{" +
        "sendFrameRate=" + sendFrameRate +
        ", encFrameRate=" + encFrameRate +
        ", renderFrameRate=" + renderFrameRate +
        ", rtt=" + rtt +
        ", upBitrate=" + upBitrate +
        ", downBitrate=" + downBitrate +
        ", fecRate=" + fecRate +
        ", arqRate=" + arqRate +
        ", estimateBW=" + estimateBW +
        ", lostRate=" + lostRate +
        '}';
}
```

3.5.1.9 网络信息回调

方法名称

void onClientStatistics(KPClientStat kPClientStat)

功能描述

回调接口显示相关网络数据

```
public static native void nativeClassInit();
public int recvFrameRate;    // 接受帧率
```

```
public int IFrameRate;      // 关键帧率
public double aveFrameSize; // 帧平均大小
public int rtt;             // 往返时延
public int lostRate;        // 估算丢包
public int upBitrate;       // 上行流量速度
public int downBitrate;     // 下行流量速度
public int displayFps;      // 显示帧
public int decodeDelay;     // 解码延迟
public int displayDelay;    // 显示延迟
public int clientPackLatency; // 客户端打包延迟
public int clientUnPackLatency; // 客户端解包延迟
public int videoDecodeLatency; // 视频解码延迟
public int videoTransLatency; // 视频传输延迟
```

重写toString显示参数信息

```
public String toString() {
    return "KPClientStat{" +
        "recvFrameRate=" + recvFrameRate +
        ", IFrameRate=" + IFrameRate +
        ", aveFrameSize=" + aveFrameSize +
        ", rtt=" + rtt +
        ", lostRate=" + lostRate +
        ", upBitrate=" + upBitrate +
        ", downBitrate=" + downBitrate +
        ", displayFps=" + getDisplayFps() +
        ", decodeDelay=" + getDecodeDelay() +
        ", displayDelay=" + getDisplayDelay() +
        ", clientPackLatency=" + clientPackLatency +
        ", clientUnPackLatency=" + clientUnPackLatency +
        ", videoDecodeLatency=" + videoDecodeLatency +
        ", videoTransLatency=" + videoTransLatency +
        '}';
}
```

3.5.2 应用级回调方法（App Level）

3.5.2.1 应用启动回调

方法名称

void onStartApp(AppOperateResponse<String> appOperateResponse)

功能描述

应用启动回调。该方法在应用启动操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 1. 实现回调接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onStartApp(AppOperateResponse<Void> response) {
        if ("0".equals(response.getError_code())) {
            System.out.println(" 启动成功");
        } else {
            System.err.println(" 启动失败: " + response.getError_msg());
        }
    }
};
// 2. 注册回调
CloudPhoneClient.setAppCallback(callback);
// 3. 构造启动参数（通过类名启动）
AppParams params1 = new AppParams();
params1.setClazz("com.tencent.mm.ui.LauncherUI");
params1.setIntents(new HashMap<String, Object>() {{
    put("scene", "chat");
}});
String reqID = CloudPhoneClient.startApp(params1);
// 或通过Scheme启动
AppParams params2 = new AppParams();
params2.setPackageName("qq.live");
params2.setScheme("weixin://scanqrcode");
String reqID2 = CloudPhoneClient.startApp(params2);
```

3.5.2.2 应用关闭回调

方法名称

void onCloseApp(AppOperateResponse<String> appOperateResponse)

功能描述

应用启动回调。该方法在应用关闭操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 1. 实现回调接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onCloseApp(AppOperateResponse<Void> response) {
        if ("0".equals(response.getError_code())) {
            System.out.println(" 应用关闭成功");
        } else {
            System.err.println(" 关闭失败: " + response.getError_msg());
        }
    }
};
```

```
};  
// 2. 注册回调（若未全局注册）  
CloudPhoneClient.setAppCallback(callback);  
// 3. 关闭目标应用（如微信）  
CloudPhoneClient.closeApp("com.tencent.mm");
```

3.5.2.3 获取正在运行的应用回调

方法名称

```
void onGetRunningApps(AppOperateResponse<GetAppsRsp>  
appOperateResponse)
```

功能描述

获取正在运行的应用回调。该方法在获取正在运行的应用信息完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 1. 注册回调  
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {  
    @Override  
    public void onGetInstalledApps(AppOperateResponse<GetAppsRsp> response) {  
        if (response.isSuccess()) {s  
            GetAppsRsp data = response.getData();  
            for (AppInfo app : data.getAppList()) {  
                Log.d("APP", "Name: " + app.getAppName() + ", Icon URL: " + app.getIconUrl());  
            }  
        } else {  
            Log.e("APP", "Error: " + response.getErrorMessage());  
        }  
    }  
});  
  
// 2. 构建请求  
GetAppsReq req = new GetAppsReq();  
req.setAppType(GetAppsReq.AppTypeEnum.BOTH); // 获取所有应用  
req.setQuality(80); // 图标质量设为80  
req.setPageNum(1); // 第一页  
req.setPageSize(10); // 每页10条  
req.setNeedIcon(true); // 返回图标  
  
// 3. 调用接口  
getRunningApps(req);
```

3.5.2.4 应用卸载回调

方法名称

void onUninstallApp(AppOperateResponse<String> appOperateResponse)

功能描述

应用卸载回调。该方法在应用卸载操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 1. 实现回调处理器
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onUninstallApp(AppOperateResponse<Void> response) {
        switch (response.getError_code()) {
            case "0":
                System.out.println("卸载完成");
                break;
            case "404":
                System.err.println("应用不存在");
                break;
            default:
                System.err.println("错误["+response.getError_code()+"]: "+response.getError_msg());
        }
    }
};
// 2. 注册回调（建议在初始化阶段完成）
CloudPhoneClient.setAppCallback(callback);
// 3. 执行卸载（如卸载测试应用）
CloudPhoneClient.uninstallApp("com.example.testapp");
```

3.5.2.5 应用加载回调

方法名称

void onLoadApp(AppOperateResponse<String> appOperateResponse)

功能描述

应用加载回调。该方法在应用加载操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 1. 实现带重试机制的回调处理器
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
@Override
public void onLoadApp(AppOperateResponse<Void> response) {
System.err.printf(" 加载结果[%s]: %s%n",
response.getError_code(),
response.getError_msg());
}
}
};
// 2. 注册回调（建议使用单例模式）
CloudPhoneClient.setAppCallback(callback);
// 3. 预加载目标应用（如微信）
CloudPhoneClient.loadApp("com.tencent.mm");
```

3.5.2.6 获取已安装应用回调

方法名称

```
void onGetInstalledApps(AppOperateResponse<GetAppsRsp>
appOperateResponse)
```

功能描述

获取已安装应用回调。该方法在获取已安装应用信息完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
/ 1. 创建请求对象并设置参数
GetAppsReq req = new GetAppsReq();
req.setAppType(AppTypeEnum.THIRD_APP); // 筛选第三方应用
req.setQuality(80); // 图标质量设为80（平衡清晰度与大小）
```

```
req.setPageNum(1); // 查询第1页
req.setPageSize(10); // 每页10条数据
req.setNeedIcon(true); // 需要返回图标

// 2. 注册回调接口（建议在初始化时注册一次）
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetInstalledApps(AppOperateResponse<GetAppsRsp> response) {
        if (response.isSuccess()) {
            GetAppsRsp rsp = response.getData();
            List<CloudAppInfo> apps = rsp.getAppList();
            // 处理应用列表数据（如展示名称、包名、图标等）
            for (CloudAppInfo app : apps) {
                String appName = app.getAppName();
                String packageName = app.getPackageName();
                String iconUrl = app.getIconUrl(); // 优先使用URL
                byte[] iconBytes = app.getIcon(); // URL为空时使用字节数组
                boolean isReady = app.getIsReady();
                // ... 业务逻辑处理
            }
        } else {
            // 处理接口调用失败（如网络异常、参数错误等）
            String errorMsg = response.getErrorMsg();
            int errorCode = response.getErrorCode();
        }
    }
});

// 3. 调用接口
CloudPhoneClient.getInstalledApps(req);
```

3.5.2.7 APK 安装状态回调

方法名称

void onAPKInstallState(AppOperateResponse<AppInfo> appOperateResponse)

功能描述

APK安装状态回调。该方法在APK安装状态更新时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
data	GetAppsRsp	磁盘使用数据对象

AppInfo结构

参数名	类型	说明
appName	String	应用名称，如“微信”、“设置”
packageName	String	应用包名（唯一标识），示例：com.tencent.mm
icon	String	WebP格式图标原始数据Base64编码，仅当needIcon=true时可能返回，null表示无数据
iconUrl	String	WebP格式图标URL，端侧优先使用此字段，无值时fallback到icon字节数组
isReady	Boolean	应用是否准备就绪（可启动状态） • true：应用可正常使用 • false：未安装完成或异常状态

返回值

无

调用示例

```
// 2. 注册回调接口（建议在初始化时注册一次）
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetInstalledApps(AppOperateResponse<AppInfo> response) {
        if (response.isSuccess()) {
            AppInfo rsp = response.getData();

            } else {
                // 处理接口调用失败（如网络异常、参数错误等）
                String errorMsg = response.getErrorMsg();
                int errorCode = response.getErrorCode();
            }
        }
    });
```

响应示例

```
{
  "data": {
    "appName": "支付宝",
    "packageName": "com.eg.android.AlipayGphone",
    "icon": null,
    "iconUrl": "https://example.com/icons/alipay.webp",
    "isReady": true
  }
  "errorCode": 0,
```

```
"errorMsg": "success"  
}
```

3.5.2.8 应用生命周期回调

方法名称

void onAppLifeCycle(AppOperateResponse<AppLifeCycle> appOperateResponse)

功能描述

应用生命周期回调。该方法在应用生命周期状态更新时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
data	AppLifeCycle	磁盘使用数据对象

AppLifeCycle

字段	类型	说明
packageName	String	应用包名
lifeCycleStatus	String	生命周期状态 0: 打开 1: 关闭（应用退出只需要处理状态为1的数据）
restoreAppProgress	Integer	应用还原进度，0-100
restoreCode	String	应用还原结果
intercepted	boolean	应用还原的触发类型，true 热点应用自动还原，false 用户点击触发

返回值

无

调用示例

```
// 2. 注册回调接口（建议在初始化时注册一次）
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetInstalledApps(AppOperateResponse<AppLifeCycle> response) {
        if (response.isSuccess()) {
            AppLifeCycle rsp = response.getData();

        } else {
            // 处理接口调用失败（如网络异常、参数错误等）
            String errorMsg = response.getErrorMsg();
            int errorCode = response.getErrorCode();
        }
    }
});
```

3.5.2.9 获取媒体元数据回调

方法名称

```
void onGetMediaMetadata(AppOperateResponse<MediaMetaData>
appOperateResponse)
```

功能描述

获取媒体元数据回调。该方法在获取媒体元数据完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 调用接口获取媒体元数据
CloudPhoneClient.getInstance().getMediaMetadata();

// 设置回调监听
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onGetMediaMetadata(AppOperateResponse<MediaMetaData> response) {
        if (response.isSuccess() && response.getData() != null) {
            MediaMetaData metaData = response.getData();
            // 处理获取到的媒体元数据
            String title = metaData.getMetadataKeyTitle();
            String artist = metaData.getMetadataKeyArtist();
            // ...其他字段处理
        }
    }
});
```

```
} else {  
    // 处理获取失败情况  
    String errorMsg = response.getErrMsg();  
}  
}  
  
// 其他回调方法...  
});
```

3.5.2.10 获取播放状态回调

方法名称

```
void onGetPlayBackState(AppOperateResponse<MediaPlayState>  
appOperateResponse)
```

功能描述

获取播放状态回调。该方法在获取播放状态完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 调用接口获取媒体播放状态  
CloudPhoneClient.getInstance().getPlayBackState();  
  
// 设置回调监听  
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {  
    @Override  
    public void onGetPlayBackState(AppOperateResponse<MediaPlayState> response) {  
        if (response.isSuccess() && response.getData() != null) {  
            MediaPlayState playState = response.getData();  
            // 处理获取到的播放状态信息  
            boolean isPlaying = playState.getIsPlaying();  
            int currentPos = playState.getCurrentPosition();  
            String currentTitle = playState.getTitle();  
            // ...其他字段处理  
        } else {  
            // 处理获取失败情况  
            String errorMsg = response.getErrMsg();  
        }  
    }  
});  
  
// 其他回调方法...  
});
```

3.5.2.11 音频监控回调

方法名称

```
void onMonitorAudio(AppOperateResponse<MediaMonitorData>
appOperateResponse)
```

功能描述

音频监控回调。该方法在音频监控数据更新时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
result_data	JsonObject	媒体信息

MediaMonitorData对象，包含以下字段：

字段名	类型	说明
packageName	String	提供音频播放服务的应用包名
isPlaying	Boolean	音频是否正在播放 - true：播放中 - false：暂停
metadataKeyTitle	String	歌曲名称
metadataKeyArtist	String	歌手名称
metadataKeyDuration	Integer	歌曲时长，单位：毫秒
metadataKeyAlbum	String	专辑名称
metadataKeyAlbumArtist	String	专辑歌手
metadataKeyAlbumCover	String	专辑封面，通常为图片URL或Base64编码字符串

返回值

无

调用示例

```
// 设置音频监听回调
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onMonitorAudio(AppOperateResponse<MediaMonitorData> response) {
        if (response.isSuccess() && response.getData() != null) {
            MediaMonitorData monitorData = response.getData();
            // 处理监听得到的音频信息
            String title = monitorData.getMetadataKeyTitle();
            boolean isPlaying = monitorData.getIsPlaying();
            String albumCover = monitorData.getMetadataKeyAlbumCover();
            // ...其他字段处理
        } else {
            // 处理回调失败情况
            String errorMsg = response.getErrorMsg();
        }
    }
});

// 其他回调方法...
});
```

响应示例

当音频状态或元数据发生变化时，参数中的response.getData()内容如下：

```
response.getData()
返回示例：
{
  "packageName": "com.example.musicplayer",
  "isPlaying": true,
  "metadataKeyTitle": "Perfect",
  "metadataKeyArtist": "Ed Sheeran",
  "metadataKeyDuration": 263000,
  "metadataKeyAlbum": "÷ (Divide)",
  "metadataKeyAlbumArtist": "Ed Sheeran",
  "metadataKeyAlbumCover": "https://example.com/covers/ed_sheeran_divide.jpg"
}
```

3.5.2.12 音频操作监控回调

方法名称

void onMonitorAudioAction(AppOperateResponse<AudioActionData> appOperateResponse)

功能描述

音频操作监控回调。该方法在音频操作监控数据更新时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

字段	类型	说明
error_code	String	状态码，0表示成功
error_msg	String	错误详情
result_data	JsonObject	媒体控制结果

AudioActionData对象，包含以下字段：

字段名	类型	说明
playAction	String	媒体控制动作，可选值： • MEDIA_ACTION_SKIP_PREVIOUS，媒体控制切换上一首 • MEDIA_ACTION_SKIP_NEXT，媒体控制切换下一首
actionResult	Boolean	控制结果 • true：控制成功 • false：控制失败

返回值

无

调用示例

```
// 设置媒体控制结果监听回调
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onMonitorAudioAction(AppOperateResponse<AudioActionData> response) {
        if (response.isSuccess() && response.getData() != null) {
            AudioActionData data = response.getData();
            // 处理监听得到的媒体控制结果
            String playAction = data.getPlayAction();
            boolean actionResult = data.getActionResult();
        } else {
            // 处理回调失败情况
            String errorMsg = response.getErrorMsg();
        }
    }
});

// 其他回调方法...
});
```

响应示例

当应用控制媒体切换下一首成功时，参数中的response.getData()内容如下：

```
response.getData()
返回示例：
{
  "playAction": "MEDIA_ACTION_SKIP_NEXT",
  "actionResult": "true",
}
```

3.5.2.13 音乐恢复播放回调

方法名称

```
void onResumeMusic(AppOperateResponse<MusicOperateResult>
appOperateResponse)
```

功能描述

音乐恢复播放回调。该方法在音乐恢复播放操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

无

3.5.2.14 音乐暂停播放回调

方法名称

```
void onPauseMusic(AppOperateResponse<MusicOperateResult>
appOperateResponse)
```

功能描述

音乐暂停播放回调。该方法在音乐暂停播放操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

无

3.5.2.15 音量变化回调

方法名称

void onChangeVolume(AppOperateResponse<MediaVolume>
appOperateResponse)

功能描述

音量变化回调。该方法在音量变化操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

无

3.5.2.16 亮度变化回调

方法名称

void onChangeBrightness(AppOperateResponse<MediaBrightness>
appOperateResponse)

功能描述

亮度变化回调。该方法在亮度变化操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

无

3.5.2.17 动态参数配置的回调

方法名称

void onDynamicConfigNotify(JSONObject result)

功能描述

动态参数配置的回调日志

响应示例

```
// 成功响应
{"type":"XXXConfig","result":0}
// 失败响应（应用未运行）
{"type":"XXXConfig","result":1}
```

3.5.3 操作系统级回调方法（System Level）

3.5.3.1 返回云桌面回调

方法名称

void onReturnLauncher(AppOperateResponse<LauncherResultData> appOperateResponse)

功能描述

用于处理端侧返回桌面（Launcher）后的回调结果，并根据返回原因决定：

- 是否断流
- 是否返回应用墙（MainActivity）
- 是否保持当前云应用界面

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

外层字段

字段名	类型	说明
error_code	String	接口执行结果 0：成功 - 其它：失败
data	LauncherResultData	返回桌面的控制结果数据

LauncherResultData

字段名	类型	说明
isDisconnectStream	Integer	是否断流 0：不断流 1：断流
reasonCode	Integer	返回桌面原因码
reasonDescription	String	原因描述

返回值

无

调用示例

```
@Override
public void onReturnLauncher(AppOperateResponse<LauncherResultData> appOperateResponse) {
    if (appOperateResponse.isSuccess()) {
        Log.d("CloudPhone", "成功返回云桌面");
    } else {
        Log.e("CloudPhone", "返回云桌面失败: " + appOperateResponse.getMessage());
    }
}
```

3.5.3.2 获取磁盘使用情况回调

方法名称

void onGetDiskUsage(AppOperateResponse<DiskUsagelInfo> appOperateResponse)

功能描述

获取磁盘使用情况回调。该方法在获取磁盘使用情况完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

返回值

无

调用示例

```
// 1. 实现带单位转换的回调处理器
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
@Override
public void onGetDiskUsage(AppOperateResponse<DiskUsageInfo> response) {
if(!"0".equals(response.getError_code())){
System.err.println(" 查询失败: " + response.getError_msg());
return;
}
DiskUsageInfo usage = response.getData();
System.out.printf("磁盘使用情况: \n" +
" 总容量: %.2f GB\n" +
" 已使用: %.2f GB (%.1f%%)\n" +
" 可用空间: %.2f GB\n",
bytesToGB(usage.getTotalSize()),
bytesToGB(usage.getUsedSize()),
(usage.getUsedSize() * 100.0 / usage.getTotalSize()),
bytesToGB(usage.getFreeSize()));
}
private double bytesToGB(long bytes) {
return bytes / (1024.0 * 1024 * 1024);
}
};
// 2. 发起查询
CloudPhoneClient.getDiskUsage();
```

3.5.3.3 云设备音频焦点变化回调

方法名称

```
void
onCloudDeviceAudioFocusChange(AppOperateResponse<AudioFocusStateResponse> appOperateResponse)
```

功能描述

云设备音频焦点变化回调。该方法在云设备音频焦点状态变化时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

AudioFocusStateResponse包含以下字段：

字段名	类型	说明
focusState	String	当前音频焦点状态，可能的值： • AUDIOFOCUS_GAIN：获取音频焦点 • AUDIOFOCUS_LOSS：永久失去音频焦点 • AUDIOFOCUS_LOSS_TRANSIENT：暂时失去音频焦点 • AUDIOFOCUS_LOSS_TRANSIENT_CAN_DUCK：暂时失去音频焦点但可以继续播放（音量降低）
possibleSnatcher	String	可能抢占音频焦点的应用包名，可能为null

返回值

无

调用示例

```
CloudPhoneClient.setSysCallback(new CloudPhoneClient.Callback.SysCallback() {
    @Override
    public void onCloudDeviceAudioFocusChange(CloudDeviceAudioFocusChangeResponse response) {
        if (Response.getError_code().equals("0")) {
            String focusState = response.getFocusState();
            String possibleSnatcher = response.getPossibleSnatcher();
            // 处理音频焦点变化
            switch (focusState) {

            }

            // 处理可能的抢占者信息
            if (possibleSnatcher != null) {
                Log.d("AudioFocus", "可能抢占焦点的应用: " + possibleSnatcher);
            }
        }
    }
});
```

响应示例

```
{
  "focusState": "AUDIOFOCUS_LOSS_TRANSIENT",
  "possibleSnatcher": "com.example.videoapp"
}
```

3.5.3.4 音量控制回调

方法名称

void onVolumeControl(AppOperateResponse<VolumeDTO> appOperateResponse)

功能描述

音量控制回调。该方法在音量控制操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

VolumeDTO包含以下字段：

字段名	类型	说明
isAdjustVolume	Boolean	是否是用户主动调整音量操作 - true：用户主动调整 - false：系统自动调整
streamType	Integer	音量流类型，对应Android系统的AudioManager.STREAM_*常量值
volumeIndex	Integer	当前音量索引值，范围：0 ~ maxVolumeIndex
volumeFlag	Integer	音量调整标志位，对应Android系统的AudioManager.FLAG_*常量值
maxVolumeIndex	Integer	最大音量索引值
isMute	Boolean	是否静音状态 - true：静音 - false：非静音

返回值

无

调用示例

```
CloudPhoneClient.setSysCallback(new CloudPhoneClient.Callback.SysCallback() {
    @Override
    public void onVolumeControl(VolumeControlResponse response) {
        if (response.getError_code().equals("0")) {
            boolean isAdjustVolume = response.getIsAdjustVolume();
            int streamType = response.getStreamType();
            int volumeIndex = response.getVolumeIndex();
            int maxVolume = response.getMaxVolumeIndex();
            boolean isMute = response.getIsMute();

            // 处理音量变化
            if (streamType == AudioManager.STREAM_MUSIC) {
                // 音乐音量变化
                float volumePercent = (float) volumeIndex / maxVolume;

                if (isAdjustVolume) {
                    // 用户主动调整音量
                    Log.d("VolumeControl", "用户调整音乐音量: " + volumePercent * 100 + "%");
                    updateVolumeUI(volumePercent, isMute);
                } else {
                    // 系统自动调整音量
                    Log.d("VolumeControl", "系统调整音乐音量: " + volumePercent * 100 + "%");
                }
            }

            // 处理静音状态变化
            if (isMute) {
                Log.d("VolumeControl", "音量已静音");
                showMuteIndicator();
            } else {
                Log.d("VolumeControl", "音量已恢复");
                hideMuteIndicator();
            }
        }
    }
});

// 其他回调方法...
```

响应示例

```
{
  "isAdjustVolume": true,
  "streamType": 3,
  "volumeIndex": 15,
  "volumeFlag": 8,
  "maxVolumeIndex": 30,
  "isMute": false
}
```

3.5.3.5 亮度控制回调

方法名称

```
void onBrightnessControl(AppOperateResponse<BrightnessDTO>
appOperateResponse)
```

功能描述

亮度控制回调。该方法在亮度控制操作完成时触发。

参数说明

字段名	类型	说明
appOperateResponse	AppOperateResponse	操作响应对象

回调参数

BrightnessDTO对象，包含以下字段：

字段名	类型	说明
brightnessIndex	String	亮度索引值，范围：0~255，对应系统亮度设置的原始值
screenBrightness	Double	屏幕亮度百分比，范围：0.0~1.0。0.0表示最暗，1.0表示最亮

返回值

无

调用示例

```
CloudPhoneClient.setSysCallback(new CloudPhoneClient.Callback.SysCallback() {
    @Override
    public void onBrightnessControl(BrightnessControlResponse response) {
        if (response.getError_code().equals("0")) {
            String brightnessIndex = response.getBrightnessIndex();
            double screenBrightness = response.getScreenBrightness();

            // 处理亮度变化
            Log.d("BrightnessControl", "亮度索引: " + brightnessIndex +
                ", 亮度百分比: " + String.format("%.2f", screenBrightness * 100) + "%");

            // 根据亮度调整UI
            if (screenBrightness < 0.3) {
                // 低亮度环境，切换到暗色模式
                switchToDarkMode();
            } else if (screenBrightness > 0.7) {
                // 高亮度环境，切换到亮色模式
                switchToLightMode();
            }

            // 更新亮度指示器
            updateBrightnessIndicator(screenBrightness);
        }
    }
});
```

响应示例

```
{
  "brightnessIndex": "180",
```

```
"screenBrightness": 0.7058823529411765
}
```

3.5.4 回调方法注册使用说明

3.5.4.1 概述

Callback是云手机SDK的核心回调接口，所有事件通知都通过该接口的相应方法回调。

开发者需要实现该接口来处理各种云手机事件。在CloudPhoneClient中，开发者可以通过回调注册和回调取消注册这两种方法注册和取消注册不同级别的回调接口，以便灵活地管理事件监听。

3.5.4.2 回调注册方法

3.5.4.2.1 注册基础回调接口

方法名称

void setPlayerCallback(Callback callback)

功能描述

注册基础回调接口。该方法用于注册实现了基础回调方法的接口实例。

参数说明

字段名	类型	说明
callback	Callback	实现了基础回调方法的接口实例

返回值

无

调用示例

```
CloudPhoneClient.setPlayerCallback(new CloudPhoneClient.Callback() {
    @Override
    public void onSuccess() {
        Log.d("CloudPhone", "连接成功");
    }

    @Override
    public void onTerminated() {
        Log.d("CloudPhone", "会话结束");
    }

    // 其他基础回调方法实现...
});
```

3.5.4.2.2 注册应用级回调接口

方法名称

void setAppCallback(Callback.AppCallback callback)

功能描述

注册应用级回调接口。该方法用于注册实现了应用级回调方法的接口实例。

参数说明

字段名	类型	说明
callback	Callback.AppCallback	实现了应用级回调方法的接口实例

返回值

无

调用示例

```
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onStartApp(AppOperateResponse<String> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            Log.d("CloudPhone", "应用启动成功: " + appOperateResponse.getData());
        } else {
            Log.e("CloudPhone", "应用启动失败: " + appOperateResponse.getMessage());
        }
    }
});

// 其他应用级回调方法实现...
});
```

3.5.4.2.3 注册操作系统级回调接口

方法名称

void setSysCallback(Callback.SysCallBack callback)

功能描述

注册操作系统级回调接口。该方法用于注册实现了操作系统级回调方法的接口实例。

参数说明

字段名	类型	说明
callback	Callback.SysCallBack	实现了操作系统级回调方法的接口实例

返回值

无

调用示例

```
CloudPhoneClient.setSysCallback(new CloudPhoneClient.Callback.SysCallBack() {
    @Override
    public void onVolumeControl(AppOperateResponse<VolumeDTO> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            VolumeDTO volumeDTO = appOperateResponse.getData();
            Log.d("CloudPhone", "音量控制结果: " + volumeDTO.getVolume());
        } else {
            Log.e("CloudPhone", "音量控制失败: " + appOperateResponse.getMessage());
        }
    }
});

// 其他系统级回调方法实现...
});
```

3.5.4.3 回调取消注册方法

3.5.4.3.1 取消注册基础回调接口

方法名称

void UnsetPlayerCallback(Callback callback)

功能描述

取消注册基础回调接口。该方法用于取消注册已注册的基础回调接口实例。

参数说明

字段名	类型	说明
callback	Callback	已注册的基础回调接口实例

返回值

无

调用示例

```
CloudPhoneClient.UnsetPlayerCallback(playerCallback);
```

3.5.4.3.2 取消注册应用级回调接口

方法名称

void UnsetAppCallback(Callback.AppCallback callback)

功能描述

取消注册应用级回调接口。该方法用于取消注册已注册的应用级回调接口实例。

参数说明

字段名	类型	说明
callback	Callback.AppCallback	已注册的应用级回调接口实例

返回值

无

调用示例

```
CloudPhoneClient.UnsetAppCallback(appCallback);
```

3.5.4.3.3 取消注册操作系统级回调接口

方法名称

void UnsetSysCallback(Callback.SysCallBack callback)

功能描述

取消注册操作系统级回调接口。该方法用于取消注册已注册的操作系统级回调接口实例。

参数说明

字段名	类型	说明
callback	Callback.SysCallBack	已注册的操作系统级回调接口实例

返回值

无

调用示例

```
CloudPhoneClient.UnsetSysCallback(sysCallback);
```

3.5.4.4 回调注册使用建议及示例

回调注册使用建议

- 按需注册：开发者应根据实际需求注册对应的回调接口，避免不必要的资源占用。

- 避免重复注册：确保同一个回调接口不会被多次注册，防止事件重复触发。
- 及时取消注册：在不再需要监听事件时，及时取消注册回调接口，避免内存泄漏。
- 线程安全：回调注册和取消注册操作应在主线程中执行，确保线程安全。
- 模块化管理：建议将不同级别的回调接口分别管理，提高代码可维护性。

回调注册示例

注意事项：必须显示实现所有abstract回调方法，即使为空。

- 注册全局回调：

```
CloudPhoneClient.setPlayerCallback(new CloudPhoneClient.Callback() {
    @Override
    public void onSuccess() {
        Log.d("CloudPhone", "连接成功");
    }

    @Override
    public void onTerminated() {
        Log.d("CloudPhone", "会话结束");
    }

    @Override
    public void onFailure(int code, String message) {
        Log.e("CloudPhone", "错误码: " + code + ", 错误信息: " + message);
    }

    // 实现 AppCallback 中的方法
    @Override
    public void onStartApp(AppOperateResponse<String> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            Log.d("CloudPhone", "应用启动成功: " + appOperateResponse.getData());
        } else {
            Log.e("CloudPhone", "应用启动失败: " + appOperateResponse.getMessage());
        }
    }

    @Override
    public void onGetRunningApps(AppOperateResponse<GetAppsRsp> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            GetAppsRsp appsRsp = appOperateResponse.getData();
            Log.d("CloudPhone", "正在运行的应用: " + appsRsp.getApps().toString());
        } else {
            Log.e("CloudPhone", "获取运行应用失败: " + appOperateResponse.getMessage());
        }
    }

    // 其他方法可以不实现，前提是它们不是 abstract 的
});
```

- 分别注册需要的回调：

```
// 注册应用级回调
CloudPhoneClient.setAppCallback(new CloudPhoneClient.Callback.AppCallback() {
    @Override
    public void onStartApp(AppOperateResponse<String> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            Log.d("CloudPhone", "应用启动成功: " + appOperateResponse.getData());
        } else {
            Log.e("CloudPhone", "应用启动失败: " + appOperateResponse.getMessage());
        }
    }
});
```

```
// 其他应用级回调方法实现...
});

// 注册系统级回调
CloudPhoneClient.setSysCallback(new CloudPhoneClient.Callback.SysCallBack() {
    @Override
    public void onVolumeControl(AppOperateResponse<VolumeDTO> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            VolumeDTO volumeDTO = appOperateResponse.getData();
            Log.d("CloudPhone", "音量控制结果: " + volumeDTO.getVolume());
        } else {
            Log.e("CloudPhone", "音量控制失败: " + appOperateResponse.getMessage());
        }
    }
});

// 其他系统级回调方法实现...
});
```

- Activity类直接implements (实现) 对应回调接口类, 并在初始化Activity时注册回调CloudPhoneClient.setPlayerCallback(this):

```
public class MyCloudPhoneActivity extends Activity implements CloudPhoneClient.Callback {
    private PlayerFragment mPlayerFragment;
    private void initPlayer() {
        mPlayerFragment = (PlayerFragment)
getFragmentManager().findFragmentById(com.nbc.acsdk.android.R.id.fragment_player);
        CloudPhoneClient.bindFragment(mPlayerFragment);
        getFragmentManager().beginTransaction().show(mPlayerFragment).commit();

        CloudPhoneClient.setPlayerCallback(this);
        CloudPhoneClient.start(mPlayerFragment, getIntent().getExtras());
    }

    // 只实现关心的回调方法
    @Override
    public void onSuccess() {
        Log.d("CloudPhone", "连接成功");
    }

    @Override
    public void onFailure(int code, String message) {
        Log.e("CloudPhone", "错误码: " + code + ", 错误信息: " + message);
    }

    // 实现 AppCallback 中的方法
    @Override
    public void onStartApp(AppOperateResponse<String> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            Log.d("CloudPhone", "应用启动成功: " + appOperateResponse.getData());
        } else {
            Log.e("CloudPhone", "应用启动失败: " + appOperateResponse.getMessage());
        }
    }

    @Override
    public void onGetRunningApps(AppOperateResponse<GetAppsRsp> appOperateResponse) {
        if (appOperateResponse.isSuccess()) {
            GetAppsRsp appsRsp = appOperateResponse.getData();
            Log.d("CloudPhone", "正在运行的应用: " + appsRsp.getApps().toString());
        } else {
            Log.e("CloudPhone", "获取运行应用失败: " + appOperateResponse.getMessage());
        }
    }
}
```

```
}  
  
// 其他方法可以不实现，不会被调用  
}
```

3.6 常态化体验指标配置模块

3.6.1 视频动态调整配置参数

方法名称

int setCloudVideoConfig(String jsonStr)

功能描述

设置视频相关的动态配置，基本校验，返回成功与否

请求参数

参数名	类型	必填	描述	示例值
jsonStr	String	是	视频配置参数的 jsonObject转成的 String	{"videoBitrate":1000,"fps":30, "width":800,"height":300,"rc Mode":1,"gop":2000,"videoUp per":5000,"videoLower":4000}

JSON参数说明：

编码码率（单位：Mbps）：key: bitrate, value；注意：修改后需要同步设置码率上下限 1~100

编码帧率（单位：fps）：key: fps, value: 5~60；

编码分辨率 宽：key: width, value；仅校验比例正确

编码分辨率 高：key: height, value；仅校验比例正确

码率控制模式（ABR/CBR/CAPPED_CRF）：key: rcMode, value 0~3, 0 ABR 1 CRF 2 CBR 3 CAPPED_CRF；

编码GOP大小（30~5000）：key: gop, value 30~5000；

码率上限（单位：Kbps）：key: videoUpper, value；校验上限>下限 0~100000

码率下限（单位：Kbps）：key: videoLower, value；校验上限>下限 0~100000

调用样例

```
// 1. 接口  
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {  
    public static int setCloudVideoConfig(String jsonConfig) {  
        return CloudPhoneClientInternal.setCloudVideoConfig(jsonConfig);  
    }  
};
```

```
// 2. 调用
JSONObject videoConfig = new JSONObject();
...
CloudPhoneClient.setCloudVideoConfig(videoConfig.toString());
```

错误日志

报错参数	描述
bitrate	Failed to set cloud video config, invalid bitrate
fps	Failed to set cloud video config, invalid fps
width, height	Failed to set cloud video config, width and height should be appear in pairs
rcMode	Failed to set cloud video config, invalid rcMode
gop	Failed to set cloud video config, invalid gop
videoUpper, videoLower	Failed to set cloud video config, video upper should larger than lower
videoUpper, videoLower	Failed to set cloud video config, invalid video upper or lower
jsonObject内容错误	Failed to set cloud video config, invalid json format

3.6.2 音频动态调整配置参数

方法名称

```
int setCloudAudioConfig(String jsonStr)
```

功能描述

设置音频相关的动态配置，基本校验，返回成功与否

请求参数

参数名	类型	必填	描述	示例值
jsonStr	String	是	音频配置参数的 jsonObject转成的 String	{"audioBitrate":256000,"audioUpper":256,"audioLower":64}

JSON参数说明：

编码码率（单位：bps，13200~512000）：key：bitrate，value 13200~512000；

码率上限（单位：Kbps）：key：audioUpper，value 0~512；

码率下限（单位：Kbps）：key: audioLower, value 0~512;

调用样例

```
// 1. 接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    public static int setCloudAudioConfig(String jsonConfig) {
        return CloudPhoneClientInternal.setCloudAudioConfig(jsonConfig);
    }
};
// 2. 调用
JSONObject audioConfig = new JSONObject();
...
CloudPhoneClient.setCloudAudioConfig(audioConfig.toString());
```

错误日志

报错参数	描述
bitrate	Failed to set cloud audio config, invalid bitrate
audioUpper, audioLower	Failed to set cloud audio config, audio upper should larger than lower
audioUpper, audioLower	Failed to set cloud audio config, invalid audio upper or lower
jsonObject内容错误	Failed to set cloud audio config, invalid json format

3.6.3 JitterBuffer 配置参数

方法名称

int setJitterBuffer(String jsonStr)

功能描述

设置音视频jitter上下限相关的动态配置，基本校验，返回成功与否

请求参数

参数名	类型	必填	描述	示例值
jsonStr	String	是	jitter配置参数的 jsonObject转成的 String	{"videoJitterUpper":4000, "videoJitterLower": 2000 "audioJitterUpper": 3000, "audioJitterLower": 1000 }

JSON参数说明：
jitter上限（单位：ms）：key: videoJitterUpper, value ； 设置接收端jitter buffer 0~5000

jitter下限（单位：ms）：key: videoJitterLower, value；设置接收端jitter buffer 0~5000

jitter上限（单位：ms）：key: audioJitterUpper, value；设置接收端jitter buffer 0~5000

jitter下限（单位：ms）：key: audioJitterLower, value；设置接收端jitter buffer 0~5000

调用样例

```
// 1. 接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    public static int setJitterBuffer(String jsonConfig) {
        return CloudPhoneClientInternal.setJitterBuffer(jsonConfig);
    }
};
// 2. 调用
JSONObject jitterBuffer = new JSONObject();
...
CloudPhoneClient.setJitterBuffer(JitterConfig.toString());
```

错误日志

报错参数	描述
videoJitterUpper, videoJitterLower, audioJitterUpper, audioJitterLower	Failed to set jitter buffer, upper and lower should be appear in pairs
videoJitterUpper, videoJitterLower	Failed to set video jitter buffer, invalid upper or lower
audioUpper, audioLower	Failed to set audio jitter buffer, invalid upper or lower
videoJitterUpper, videoJitterLower, audioJitterUpper, audioJitterLower	Failed to set jitter buffer, lower larger than upper
videoJitterUpper, videoJitterLower, audioJitterUpper, audioJitterLower	Failed to set jitter buffer, adapter return error
jsonObject内容错误	Failed to set jitter buffer, invalid json format

3.6.4 补帧精灵配置参数

方法名称

int setCloudRenderRepeat(String jsonStr)

功能描述

设置补帧精灵的参数，基本校验，返回成功与否

请求参数

参数名	类型	必填	描述	示例值
jsonStr	String	是	补帧配置参数的 jsonObject转成的 String	{"enable": true, "durationMs": 60, "frameRate": 10}

JSON参数说明：

补帧开关（0~1）：key: enable, value true/false；实际更新补帧精灵

补帧时长：key: durationMs, value 单位ms；实际更新补帧精灵

补帧帧率：key: frameRate, value 5~60，5的倍数；实际更新补帧精灵

调用样例

```
// 1. 接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    public static int setCloudRenderRepeat(String jsonConfig) {
        return CloudPhoneClientInternal.setCloudRenderRepeat(jsonConfig);
    }
};
// 2. 调用
JSONObject renderRepeat= new JSONObject();
...
CloudPhoneClient.setCloudRenderRepeat(renderRepeat.toString());
```

错误日志

报错参数	描述
durationMs	Failed to set cloud render config, invalid durationMs
frameRate	Failed to set cloud render config, invalid frameRate
jsonObject内容错误	Failed to set cloud render config, invalid json format

3.6.5 视频解码配置

方法名称

int setDecodeMode(int mode)

功能描述

设置视频解码模式，基本校验，返回成功与否

请求参数

参数名	类型	必填	描述	示例值
mode	int	是	设置视频解码模式	{"decodeMode": 0}

JSON参数说明：

解码模式（FAST、BALANCE）：key: decodeMode, value 0~1,0 FAST 1 BALANCE；

调用样例

```
// 1. 接口
CloudPhoneClient.Callback.AppCallback callback = new CloudPhoneClient.Callback.AppCallback() {
    public static int setDecodeMode(int mode) {
        return CloudPhoneClientInternal.setDecodeMode(mode);
    }
};
// 2. 调用
JSONObject decodeMode= new JSONObject();
...
CloudPhoneClient.setDecodeMode(decodeMode.toString());
```

错误日志

报错参数	描述
mode	Failed to set decode mode, invalid mode

说明

设置视频解码模式接口 setDecodeMode(mode: 0 FAST, 1 BALANCE)

使用规则：

非视频类APP使用FAST解码模式。

视频类APP推荐使用BALANCE解码模式。该模式通过缓冲实现平滑播放，确保极致流畅。

限制：

硬件显存严重不足时，不能使用 BALANCE 解码模式。

使用BALANCE 解码模式会增加端到端时延。

3.7 附录

3.7.1 错误码说明

3.7.1.1 错误码

错误码	描述	可能触发接口
1001	无效参数	所有接口
1002	云机未连接	所有接口
1003	云机繁忙	所有接口
1004	权限不足	所有接口
1005	网络通信失败	所有接口
1006	云机内部错误	所有接口
1007	请求超时	所有接口
1009	无效上下文	所有接口
2001	应用不存在/未安装	startApp, uninstallApp, loadApp
2002	应用已运行	startApp
2003	应用未运行	closeApp
2004	无启动Activity	startApp
2005	系统应用限制操作	closeApp, uninstallApp
2006	应用操作被系统限制	startApp, closeApp
2007	应用包损坏	startApp
2008	操作被用户取消	uninstallApp
2009	应用状态获取失败	getRunningApps
2010	应用截图生成失败	getRunningApps
2011	应用正在还原中	startApp, loadApp
2012	应用启动失败	startApp
2014	应用还原失败	loadApp
2015	应用卸载失败	uninstallApp
3001	系统权限不足	uninstallApp
3002	有进程阻止操作	closeApp
3003	自定义ROM限制操作	uninstallApp
3004	系统资源不足	startApp

3.7.1.2 状态码

错误码	错误说明	处理建议
CLOUD_PHONE_RET_OK = 0	操作成功	正常流程，继续执行
CLOUD_PHONE_RET_INTERNAL_ERROR = 100000	内部错误	检查日志，联系技术支持
CLOUD_PHONE_RET_INVALID_PARAMS = 100001	无效参数	检查参数格式和值是否正确
CLOUD_PHONE_RET_INVALID_TOKEN = 100002	无效的token	重新获取token或检查token是否过期
CLOUD_PHONE_RET_INVALID_VERSION = 100003	版本不匹配	检查SDK版本是否兼容
CLOUD_PHONE_RET_GAME_NOT_FOUND = 100004	游戏未找到	检查游戏包名是否正确，游戏是否已安装
CLOUD_PHONE_RET_GAME_LAUNCH_FAIL = 100005	游戏启动失败	检查游戏兼容性，尝试重新启动
CLOUD_PHONE_RET_EXPIRED = 100006	会话已过期	重新初始化连接
CLOUD_PHONE_RET_QUIT_QUEUE = 100007	退出队列	用户主动退出排队，可重新申请
CLOUD_PHONE_RET_USER_EXIT = 100008	用户主动退出	正常退出流程，清理资源
CLOUD_PHONE_RET_DUPLICATE_USER = 100009	重复用户	检查用户ID是否唯一，避免重复登录
CLOUD_PHONE_RET_STOP_STREAMING = 100010	停止串流	正常停止流程，检查是否为主调用stop()

3.7.1.3 网络错误码

错误码	错误说明	处理建议
CLOUD_PHONE_RET_NETWORK_NOT_AVAILABLE = 110000	网络不可用	检查网络连接，提示用户检查网络设置
CLOUD_PHONE_RET_CONNECT_ERROR = 110001	连接错误	检查服务器地址和网络，尝试重新连接
CLOUD_PHONE_RET_REQUEST_ERROR = 110002	请求错误	检查请求参数，重试请求
CLOUD_PHONE_RET_DROPLINE = 110003	断线	网络不稳定，尝试重连

错误码	错误说明	处理建议
CLOUD_PHONE_RET_HEARTBEAT_TIMEOUT = 110004	心跳超时	网络延迟过高，检查网络状况
CLOUD_PHONE_RET_STREAM_SERVER_ERROR = 110006	串流服务器错误	服务器故障，稍后重试

3.7.1.4 错误处理

- 错误回调处理：在onFailure回调中根据错误码进行相应处理。
- 网络错误重试：对于网络相关错误（110000-110006），建议实现自动重连机制。
- 用户提示：对于用户相关的错误（100008-100011），提供友好的用户提示。
- 日志记录：记录错误码和错误信息，便于问题排查。
- 异常恢复：对于可恢复的错误（如网络断线），实现自动恢复机制。

错误处理示例：

```
@Override
public void onFailure(int code, String message) {
    switch (code) {
        case CloudPhoneConst.CLOUD_PHONE_RET_NETWORK_NOT_AVAILABLE:
            // 网络不可用，提示用户检查网络
            showNetworkErrorDialog();
            break;
        case CloudPhoneConst.CLOUD_PHONE_RET_KICKED:
            // 被踢出房间，提示用户
            showKickedDialog(message);
            break;
        case CloudPhoneConst.CLOUD_PHONE_RET_EXPIRED:
            // 会话过期，重新初始化
            reinitializeConnection();
            break;
        default:
            // 其他错误，记录日志
            Log.e("CloudPhone", "Error code: " + code + ", message: " + message);
            showErrorDialog("连接失败，请稍后重试");
    }
}
```

3.7.2 调用 Android SDK 开放接口注意事项

- 所有静态方法都是线程安全的，可以在多线程环境下调用。
- 某些方法需要在start调用之前设置才生效（如setVideoDecoder）。
- 回调方法在主线程中执行，避免耗时操作。
- 文件操作相关方法需要确保文件路径存在且有权限。
- 网络相关操作需要确保网络连接正常。

4 KooPhone H5 SDK 开放接口

4.1 基本功能模块

4.1.1 串流控制接口

4.1.1.1 启动串流

方法名称

```
player.initMediaStream(params)
```

功能描述

启动Streaming，调用此函数之后，后续通过回调函数通知启动结果。

params 参数说明

字段名	类型	是否必选	说明
boxid	string	是	需要启动Streaming的目标设备ID
token	string	是	启动Streaming的鉴权token
uuid	string	是	用户唯一标识符
memberid	string	是	成员ID（用于token安全校验）
muted	boolean	否	本次Streaming是否静音（继承播放器默认值）

字段名	类型	是否必选	说明
profile	int	否	视频质量，默认值：2 • -1：自动 • 0：流畅 • 1：普通 • 2：高清
location	string	否	客户端地理位置信息
signaling_url	string	是	信令服务器地址
accelerometer_rotation	int	否	设备旋转设置 • 1：旋转屏幕内容 • 0：保持纵向 • -1：保持云机当前设置
app_params	string	否	启动云应用的附加参数。JSON字符串结构： • clazz=包名+类名 • scheme=启动协议 • intents=参数对象
prefer_ipv6	boolean	否	是否优先使用IPv6网络传输流媒体，默认值：false
ipv6_timeout	int	否	IPv6连接超时时间（毫秒），默认值：5000

返回值

无

调用示例

```
let player = client.createPlayer(options);
player.initMediaStream({
  boxid: 'xxx',
  token: 'xxx',
  uuid: 'xxx',
```

```
signaling_url: 'https://xx.xx.xx.xx:xx'  
});
```

4.1.1.2 终止串流

方法名称

player.closeMediaStream()

功能描述

主动关闭Streaming串流。调用后云机将直接退出串流，不会进入keeping状态等待重连。

参数说明

无

返回值

无

调用示例

```
let player = client.createPlayer(options);  
// 主动关闭串流  
player.closeMediaStream();
```

4.1.1.3 恢复串流

方法名称

player.resumeMediaStream()

功能描述

重新激活窗口并恢复Streaming传输。需在statechange回调状态为countdown（关闭倒计时）或keeping（重连等待）时调用。

参数说明

无

返回值

boolean：仅表示流程是否触发成功（非最终结果），实际结果需通过statechange回调监听（状态变为playing表示成功）。

调用示例

```
let player = client.createPlayer(options);  
// 当进入倒计时/重连状态时激活应用  
const isTriggered = player.resumeMediaStream();  
if (!isTriggered) {
```

```
console.error('继续运行应用失败');  
}
```

4.1.1.4 销毁串流 player 实例

方法名称

player.destroyMediaStream()

功能描述

销毁CloudPhonePlayer实例，需在window.onbeforeunload()中调用。

参数说明

无

返回值

无

调用示例

```
let player = client.createPlayer(options);  
//...  
//主动销毁  
player.destroy();  
//页面退出时销毁  
window.onbeforeunload = function () {  
    if (player) {  
        player.destroyMediaStream();  
    }  
};
```

4.1.2 基础配置管理接口

4.1.2.1 设置回调

方法名称

player.on(event, listener)

功能描述

设置回调事件的处理函数。回调函数必须在调用initMediaStream()方法前设置。

参数说明

字段名	类型	是否必选	说明
event	string	是	回调事件名称
listener	function	是	回调事件的处理函数

返回值

无

调用示例

```
// 设置 statechange 事件的回调
player.on("onStateChange", function (state) {
  if (state === "playing") {
    console.log("player正在播放中");
  }
});
```

4.1.2.2 取消回调

方法名称

player.off(event, listener)

功能描述

移除回调事件的处理函数。

参数说明

字段名	类型	是否必选	说明
event	string	是	回调事件名称
listener	function	是	调用on时设置的处理函数

返回值

无

调用示例

```
function statechangeListener(event) {
  console.log("onStateChange:" + event);
}
player.on("onStateChange", statechangeListener);
// 移除 onStateChange 事件的回调
player.off("onStateChange", statechangeListener);
```

4.1.3 高级配置管理接口

4.1.3.1 配置定位功能

方法名称

`player.setGpsStatus(status)`

功能描述

打开或者关闭GPS功能，默认status = disable表示关闭。

options 参数说明

字段名	类型	是否必选	说明
status	string	是	是否配置定位功能，默认值： disable - enable：启动GPS - disable：停用GPS

返回值

无

调用示例

```
// 启用GPS
player.setGpsStatus('enable');
```

4.1.4 配置查询接口

4.1.4.1 获取 SDK 版本号

方法名称

`version()`

功能描述

获取当前JSSDK版本号。该方法用于获取SDK的版本信息，便于调试和版本管理。

参数说明

无

返回值

成功返回当前SDK版本。

调用示例

```
let client = new KPClient();
let version = client.version();
console.log("current version: " + version);
```

4.1.4.2 检查浏览器是否支持串流

方法名称

player.isBrowserSupport()

功能描述

检测浏览器是否支持Streaming功能。

参数说明

无

返回值

支持返回true，不支持返回false。

调用示例

```
if (player.isBrowserSupport()) {
    console.log("浏览器支持 Streaming");
} else {
    console.error("浏览器不支持 Streaming");
}
```

4.2 视频功能模块

4.2.1 画面控制接口

4.2.1.1 设置云机画面分辨率

方法名称

player.setCloudScreenAspect(screenHeight, screenWidth)

功能描述

设置云机屏幕分辨率比例。

参数说明

字段名	类型	是否必选	说明
screenHeight	int	是	云机屏幕高度

字段名	类型	是否必选	说明
screenWidth	int	是	云机屏幕宽度

返回值

无

调用示例

```
// 设置1080P分辨率，16:9比例
player.setCloudScreenAspect(1080, 1920);
```

4.2.1.2 设置全屏播放

方法名称

player.setMediaFullScreen()

功能描述

串流media全屏播放。

参数说明

无

返回值

无

调用示例

```
// 单击按钮进入全屏
fullScreenBt.addEventListener('click', () => {
  player.setMediaFullScreen();
});
```

4.2.1.3 退出全屏播放

方法名称

player.setMediaExitFullScreen()

功能描述

串流media退出全屏播放。

参数说明

无

返回值

无

调用示例

```
// 单击按钮退出全屏
exitFullScreenBtn.addEventListener('click', () => {
  player.setMediaExitFullScreen();
});
```

4.2.1.4 设置画中画模式

方法名称

player.setPicInPicStatus(status)

功能描述

设置画中画模式，进入画中画模式或退出画中画模式。

参数说明

字段名	类型	是否必选	说明
status	string	是	是否设置画中画模式，默认值： disable - enable：进入画中画模式 - disable：退出画中画模式

返回值

无

调用示例

```
// 单击按钮进入画中画
pipBtn.addEventListener('click', () => {
  player.setPicInPicStatus('enable');
});
```

4.2.2 视频配置管理接口

4.2.2.1 设置质量偏好

方法名称

player.setVideoQuality(level)

功能描述

设置调整视频/画面质量。

参数说明

字段名	类型	是否必选	说明
level	int	是	调整视频/画面质量 • -1：自动 • 0：流畅 • 1：普通 • 2：高清

返回值

无

调用示例

```
// 调整为流畅模式
player.setVideoQuality(0);
```

4.2.2.2 设置画质

方法名称

player.setResProfile(profile)

功能描述

设置云手机的画质。profile内参数的优先级：level > width > height.

1. 如果设置了level，按level匹配处理云机画质档位；
2. 如果没有设置level（值为null），且设置了width，按width距离计算就近的云机画质档位；
3. 如果没有设置level（值为null），width（值为null），且设置了height，按height距离计算就近的云机画质档位。

说明

width和height与分辨率的对应关系：例如1080p画质档位，分辨率为1080x1920，此时width对应于1920，height对应于1080。

另外，参数设置匹配时，还需考虑端侧硬件支持的档位限制。

参数说明

字段名	类型	说明
profile	object	profile对象描述： level: int，画质的级别 -1：自动 0：流畅 1：普通 2：高清 width: int，画面宽度 height: int，画面高度

返回值

无

调用示例

```
// 调整为流畅模式
let profile = { level:2, width:1280, height:720 };
player.setResProfile(profile);
```

4.2.3 视频配置查询接口

4.2.3.1 获取云机当前的 profile 配置列表

方法名称

player.getProfileList()

功能描述

获取视频质量配置列表（结果通过onProfileList事件回调）。

参数说明

无

返回值

无

调用示例

```
player.addEventListener('onProfileList', (profiles) => {
  console.log('可用画质选项:', profiles);
});
player.getProfileList();
```

4.2.3.2 获取媒体元素

方法名称

player.getMediaElement()

功能描述

获取底层video媒体元素。

参数说明

无

返回值

HTMLVideoElement：视频元素对象。

调用示例

```
const videoEl = player.getMediaElement();
```

4.3 音频功能模块

4.3.1 音频控制接口

4.3.1.1 设置静音

方法名称

player.setMuted(muted)

功能描述

设置串流player静音。

参数说明

字段名	类型	是否必选	说明
muted	boolean	是	设置静音 - true：静音 - false：取消静音

返回值

无

调用示例

```
// 开启静音
player.setMuted(true);
```

4.3.1.2 设置云机音量

方法名称

player.setCloudVolume(volume)

功能描述

设置云机的音量，注意这个音量将作用到云机设备，不是串流player的声音。

参数说明

字段名	类型	是否必选	说明
volume	int	是	云机的音量，音量值0~15

返回值

无

调用示例

```
// 设置云机音量70%
player.setCloudVolume(11);
```

4.3.1.3 设置串流播放器音量

方法名称

player.setVolume(volume)

功能描述

设置串流Player音量。

参数说明

字段名	类型	是否必选	说明
volume	float	是	串流播放器音量，范围0~1，1为最大音量

返回值

无

调用示例

```
// 设置音量为50%  
player.setVolume(0.5);
```

4.3.2 音频配置查询接口

4.3.2.1 获取音量大小

方法名称

player.getVolume()

功能描述

获取当前音量值。

参数说明

无

返回值

float：当前音量值。

调用示例

```
const currentVolume = player.getVolume();  
console.log(`当前音量: ${currentVolume}`);
```

4.3.2.2 查询云机是否静音

方法名称

player.isMuted()

功能描述

查询云机串流播放器是否静音。

参数说明

无

返回值

静音状态返回true，非静音状态返回false。

调用示例

```
const isMute = player.isMuted();
if (isMute) {
  console.log('Player已静音');
}
```

4.4 端云指令模块

4.4.1 消息发送接口

4.4.1.1 发送手柄按键

方法名称

sendKeyEvent(keyCode, action, keyType)

功能描述

发送手柄按键事件到云机。

参数说明

字段名	类型	是否必选	说明
keyCode	Number	是	具体的按键编码 - back: 0 - home: 1 - appSwitch: 2 - volumeUp: 3 - volumeDown: 4 - volumeMute: 5 - X: 307 - Y: 308 - A: 304 - B: 305
action	Number	是	按键动作 0: 按下 1: 抬起
keyType	Number	否	键类型，默认值：207 207: 键盘类型 214: 手柄按键类型

返回值

无

调用示例

```
onButtonDown(key) {  
  let keycode = 0;  
  switch(key){  
    case "UP":  
      keycode = 19;  
      break;  
    case "DOWN":  
      keycode = 20;  
      break;  
    case "LEFT":  
      keycode = 21;  
      break;  
    case "RIGHT":  
      keycode = 22;  
      break;  
    case "X":  
      keycode = 307;  
      break;  
    case "Y":  
      keycode = 308;  
      break;  
    case "A":  
      keycode = 304;  
      break;  
    case "B":  
      keycode = 305;  
      break;  
    case "L1":  
      keycode = 310;  
      break;  
    case "L2":  
      keycode = 312;  
      break;  
    case "R1":  
      keycode = 311;  
      break;  
    case "R2":  
      keycode = 313;  
      break;  
    case "SELECT":  
      keycode = 314;  
      break;  
    case "START":  
      keycode = 315;  
      break;  
    case "L3":  
      keycode = 317;  
      break;  
    case "R3":  
      keycode = 318;  
      break;  
  }  
  console.log((new Date()).toLocaleString()+" onButtonDown "+ key+" keycode"+keycode);  
  //发送虚拟手柄某个按键被按下的事件到云机  
  player.sendKeyEvent(keycode,0,214);  
}
```

4.4.1.2 上传文件

方法名称

player.sendFile(file, dirPath, fileType, localFilePath)

功能描述

文件上传功能。

参数说明

字段名	类型	是否必选	说明
file	File/string	是	- File类型：表示上传的目标文件 - string类型：表示上传的图像的BASE64编码，此时file字符串必须以data:image开头
uploadPath	string	是	上传文件的相对目标路径（云机路径：建议放在/sdcard/Download下）
fileType	string	是	文件类型，支持image和file两种类型
localFilePath	string	是	本地文件路径，只有file格式为string时，此参数需要有实际值

返回值

无

调用示例

```
<div>
  <input type="file" id="inputFile">
  <button onclick="sendFile()">上传文件</button>
  <div id="transfer_progress"></div>
  <div id="transfer_dir"></div>
  <div id="file_tree">
  </div>
</div>
<br>

<script type="text/javascript">
function sendFile(){
  console.log("开始上传文件");
  let files = document.querySelector("#inputFile").files;
  if (files && files.length > 0) {
    let f = files[0];
    let uploadPath = '/sdcard/Download/test';
    // 文件类型支持image和file，默认为file类型
    let fileType = 'file';
    // 当使用base64传输时，localFilePath为本地文件名，当使用二进制传输时，localFilePath使用空字符串即可
  }
}
```

```
        let localFilePath = "";
        player.sendFile(f, uploadPath, fileType, localFilePath)
    } else {
        alert("请选择文件");
    }
    openModal();
}

function openModal() {
    modal.style.display = 'block';
    document.body.style.overflow = 'hidden'; // 禁止底层页面滚动
}
</script>
```

4.4.1.3 设置禁止传输的键盘按键事件

方法名称

player.setDisableKeys(keysArray)

功能描述

设置禁止传输的键盘按键事件。

参数说明

字段名	类型	是否必选	说明
keysArray	list	是	keysArray为键盘按键事件(event.code)的数组，如['Escape','Enter']

返回值

无

调用示例

```
//设置禁止发送esc和enter按键的事件
player.setDisableKeys(['Escape', 'Enter']);
```

4.4.1.4 发送虚拟按键

方法名称

player.sendVirtualButtonEvent(key)

功能描述

发送虚拟按键到云机。

参数说明

字段名	类型	是否必选	说明
key	Number	是	具体的按键编码 - back: 0 - home: 1 - appSwitch: 2

返回值

发送成功返回true，发送失败返回false。

调用示例

```
// 发送返回键
player.sendVirtualButtonEvent(0);
```

4.4.1.5 剪切板同步

方法名称

player.paste(text)

功能描述

将文本发送至云机剪切板。

参数说明

字段名	类型	是否必选	说明
text	string	是	剪贴板内容

返回值

无

调用示例

```
// 剪切板同步
player.paste('this is clipboard text');
```

4.4.1.6 发送摇杆轴事件

方法名称

sendAxesEvent(action,x,y,z,rz,lt,rt,hx,hy)

功能描述

发送摇杆轴事件到云机。

参数说明

字段名	类型	是否必选	说明
action	Number	是	摇杆轴事件 0: 按下 1: 抬起 2: 移动中
x	Number	是	-1~1，左摇杆横坐标
y	Number	是	-1~1，左摇杆纵坐标
z	Number	是	-1~1，右摇杆横坐标
rz	Number	是	-1~1，右摇杆纵坐标
lt	Number	是	lt值，默认值：0
rt	Number	是	rt值，默认值：0
hx	Number	是	hx值，默认值：0
hy	Number	是	hy值，默认值：0

返回值

无

调用示例

```
//用户操纵左摇杆
handleLeftMove(payload) {
  const { data } = payload;
  //发送左摇杆的横坐标和纵坐标信息到云机
  player.sendAxesEvent(2,payload.data.vector.x,-payload.data.vector.y,0,0,0,0,0);
}

//用户抬起左摇杆
handleLeftEnd(payload) {
  //发送左摇杆坐标归零的信息到云机（使用趋近于零的1e-3）
  player.sendAxesEvent(2,1e-3,1e-3,0,0,0,0,0);
}
```

4.5 回调及监听模块

4.5.1 基础回调方法

4.5.1.1 onClose

事件说明

player播放结束或失败时触发。

回调参数说明

字段名	类型	是否必选	说明
event	Object	是	包含错误信息 - event.code: int, 消息码 - event.message: string, 消息文字描述 - event.extra: string, 附加信息, 可用于屏幕共享退出的附加信息

消息码对照表

code	message	备注
0	成功	-
1000	Streaming异常断开	未知原因导致Streaming断开
1010	SI服务器连接断开	H5客户端和SI服务器连接断开
1011	SI服务器连接超时	H5客户端在超时时间内没有连上SI服务器
1012	SI服务器未接收到云机响应	在信令交互过程中, H5客户端15秒内没接收到返回信息
1013	云机已退出SI服务器	在Streaming过程中或信令交互过程中, 云机主动退出了SI服务器
1014	SI服务器无响应	客户端发送了start指令后, 信令服务器5秒内未返回start的结果
1020	IC服务器连接断开	H5客户端和IC服务器连接断开
1021	IC服务器连接超时或出错	H5客户端在超时时间内没有连上IC服务器

code	message	备注
1100	浏览器不支持	浏览器不支持串流技术
1101	请求参数不正确	请求参数错误
1102	TOKEN错误	传入的TOKEN已失效
1103	APPID错误	暂未使用
1104	APPKEY错误	暂未使用
1200	云机没有连接上SI服务器	H5客户端连上SI服务器后，查询10次（每秒1次）都未查询到云机信息
1201	云机被占用	云机正在被其他用户使用
1202	云机正在维护	云机正在维护中
1203	连接云机超时	在超时时间内，信令交互未完成或Peer未成功连接，云机主动终止了本次会话
1300	IC协商出错	网络穿透失败，公网环境未使用
1301	SD协商出错	浏览器不支持某些协议格式（如h264格式）
1302	PE连接失败	Peer未成功连接
1400	试玩时间已到期	当设置了过期时间，试玩时间到期时触发
1401	用户主动关闭	调用close()时触发（云机端会立刻释放资源回到空闲状态）
1402	在另一个页面打开了新的连接	在Streaming过程中，同一个用户（uuid）在另一个页面打开了同一应用，本页面的Streaming被强制断开
1403	长时间未操作，已退出应用	页面闲置时间达到设置的闲置超时时间（createCloudPhonePlayer传入的noOperationThreshold参数）
1404	释放Player	调用destroy()时触发（云机端会进入keeping time，等待重连。keeping time内未重连成功，云机端会释放资源回到空闲状态）
1405	禁止开发控制台调试	打开了开发控制台时强制终止Streaming，释放资源
1406	等待Media数据流超时	Peer连接成功，但5秒内未接收到A/V数据流
1407	主控用户已退出同屏互动功能	主控用户退出时，从控端也会被强制退出
1408	页面退出	window的beforeunload或pagehide事件被触发

code	message	备注
1410	云机与SI服务器连接断开，退出Streaming	在Streaming过程中，云机主动退出了SI服务器
1411	断线重连保持时间已结束，重连失败	重连过程中，未在keeping time内成功建立连接，云机释放资源回到空闲状态
1412	用户获取到另一台新云机，当前云机被回收	同一用户请求云机时设置了“切换新设备”参数，获取新云机成功时，当前云机被回收
1413	检测到带宽太低，无法进行应用	带宽检测的结果不足以支持Streaming
1414	云机与服务后台连接断开	云机和服务后台的连接发生了掉线
1415	Session ID不匹配	H5客户端的sessionid和云机保持的sessionid不一致，云机主动终止了本次会话
1416	当前无主控用户，观众无法连接	在同屏互动模式下，客户端作为观众打开应用，但当前没有主控用户连接，云机禁止观众连接
1417	云机重连超时	当连接断线并在10秒内无法成功重连时触发
1418	同屏共享已结束	主播端停止同屏，断开了观众端的连接
3000	未知错误	-

调用示例

```
player.on("onClose", function (event) {
    console.log("player已退出，code:" + event.code + "，message:" + event.message);
});
```

4.5.1.2 onError

事件说明

SDK内错误事件回调。

回调参数说明

字段名	类型	是否必选	说明
event	Object	是	包含错误信息 - event.type: string，错误类型，如： - openCameraFailed：打开摄像头失败 - openMicrophoneFailed：打开麦克风失败 - event.error: string，错误具体信息

调用示例

```
player.on("onError", function (event) {  
    console.log("sdk内发送错误，错误类型:", event.type);  
    console.log("错误具体信息:", event.error);  
});
```

4.5.1.3 onBadNetwork

事件说明

网络不佳时触发。

回调参数说明

无

调用示例

```
player.on("onBadNetwork", function () {  
    console.log("====bad network=====");  
});
```

4.5.1.4 onStatistics

事件说明

Streaming质量状态统计，周期性1秒回调一次。

回调参数说明

字段名	类型	是否必选	说明
event	Object	是	包含统计信息 - event.rtt: int, 数据往返时延（单位：ms） - event.fps: int, 帧率（单位：帧/秒） - event.bitrate: int, 一秒内接收到的video数据大小（单位：kb/s） - event.packetsLostRate: int, 一秒内的丢包率 - event.width: int, video的当前宽度 - event.height: int, video的当前高度

调用示例

```
player.on("onStatistics", function (stat) {
  console.log("rtt:" + stat.rtt);
  console.log("fps:" + stat.fps);
  console.log("bitrate:" + stat.bitrate);
  console.log("width:" + stat.width);
  console.log("height:" + stat.height);
});
```

4.5.1.5 onProfileChanged

事件说明

profile变化时的回调通知。

回调参数说明

字段名	类型	是否必选	说明
event	Object	是	包含profile变化信息 • event.fps: int, 帧率 • event.width: int, 宽度 • event.height: int, 高度 • event.minBitrate: int, bitrate范围的最小值（单位：kbps） • event.maxBitrate: int, bitrate范围的最大值（单位：kbps） • event.level: int, 当前的profile level - -1: 自动 - 0: 流畅 - 1: 普通 - 2: 高清

调用示例

```
player.on("onProfileChanged", function (event) {  
    // 打印当前的 profile  
    console.log(event);  
});
```

4.5.1.6 onOrientation

事件说明

云机的横竖屏状态回调。

回调参数说明

字段名	类型	是否必选	说明
event	Object	是	包含横竖屏状态信息， event.orientation： string - landscape：横屏 - portrait：竖屏

调用示例

```
player.on("onOrientation", function (event) {  
    console.log("当前云机横竖屏状态: " + event.orientation);  
});
```

4.5.1.7 onStateChange

事件说明

Streaming状态变化事件，用于监听播放器状态的改变。

回调参数说明

字段名	类型	是否必选	说明
event	Object	是	包含状态信息， - event.state: string，状态值，可能值包括： - new：创建新连接，初始状态 - reconnecting：Streaming正在重连，断线重连时触发 - signal_connecting：SI服务器正在连接中 - signal_connected：SI服务器连接成功 - authorized：SI服务器授权通过 - peer_connecting：正在连接云机 - peer_connected：云机连接成功 - playing：播放中，Streaming启动成功或恢复播放时触发 - paused：播放暂停，画面转到后台不可见时或用户调用pause()时触发 - countdown：用户长时间无操作，进入关闭倒计时。此时，event.extra的值是倒计时长（单位：秒） - keeping：Streaming已断开，云机保持应用中，等待用户继续运行应用 - closed：连接关闭，可不处理该消息，而是在close回调处理 - event.extra: int，附加参数，可缺省。当state为countdown时，为倒计时长（单位：秒）

调用示例

```
player.on("onStateChange", function (event) {
  switch(event.state) {
    case 'playing':
      console.log("播放中");
      // 隐藏加载动画
      break;
    case 'paused':
      console.log("播放暂停");
      break;
    case 'reconnecting':
      console.log("正在重连");
      // 显示重连动画
      break;
    case 'countdown':
```

```
        console.log("长时间未操作，进入倒计时");
        // 显示倒计时对话框，倒计时长: event.extra (单位: s)
        break;
    case 'keeping':
        console.log("长时间未操作连接已断开，是否继续运行应用");
        // 显示是否继续运行应用对话框
        break;
    case 'closed':
        console.log("player已退出");
        // 可不处理此消息，而是在 close 回调事件处理
        break;
    }
});
```

4.5.2 应用级回调方法

4.5.2.1 onCloudVolumeChange

事件说明

云机音量/静音状态改变事件回调，串流成功或云机声音设置改变时回调。

回调参数说明

字段名	类型	说明
event	Object	包含音量/静音状态信息： - event.music.maxVolume: int，媒体最大音量 - event.music.volume: int，媒体音量 - event.music.muted: boolean，媒体是否静音 - event.system.maxVolume: int，系统最大音量 - event.system.volume: int，系统音量 - event.system.muted: boolean，系统是否静音

调用示例

```
player.on("onCloudVolumeChange", function (event) {
    console.log("当前云机静音状态:", event.music.muted);
    console.log("当前云机音量:", event.music.volume);
});
```

4.5.2.2 onAppMessage

事件说明

App通知消息，当App发送com.cloudphone.cloud.APP_MESSAGE广播时，将触发该回调。

回调参数说明

字段名	类型	说明
event	Object	包含消息内容： - event.from: string，发送消息的应用包名 - event.message: string，消息内容，建议使用base64编码后的JSON字符串

调用示例

```
player.on("onAppMessage", function (event) {  
    console.log("包名:" + event.from);  
    console.log("消息:" + event.message);  
});
```

4.5.2.3 onAutoplayFailed

事件说明

自动播放失败回调。

回调参数说明

无

调用示例

```
player.on("onAutoplayFailed", function () {  
    console.log("自动播放失败");  
});
```

4.5.3 监听事件介绍

JSSDK默认监听浏览器事件，以实现对用户键盘、鼠标、触摸等行为进行反馈。

鼠标事件

JSSDK目前默认监听如下鼠标事件：

事件名	触发时机
mousedown	按下鼠标按键时（左键/右键/中键）
mouseup	松开鼠标按键时（与mousedown配对）
mousemove	按下鼠标左键并松开（完整单击），若按右键则触发contextmenu而非click
mouseout	鼠标离开元素区域
wheel	滚动鼠标滚轮
contextmenu	按下鼠标右键

键盘事件

JSSDK目前默认监听如下键盘事件：

事件名	触发时机
keydown	按下键盘按键时（持续按会重复触发）
keyup	松开键盘按键时

焦点事件

JSSDK目前默认监听如下焦点事件：

事件名	触发时机
focus	获取元素焦点时
blur	失去元素焦点时

触摸事件

JSSDK目前默认监听如下触摸事件：

事件名	触发时机
touchstart	手指触摸屏幕时
touchend	手指离开屏幕时
touchmove	手指在屏幕上滑动时
touchcancel	触摸被中断时（如来电、弹窗）

4.6 附录

4.6.1 错误码说明

错误码	错误说明	处理建议
1203	连接云机超时	请检查网络后重试
1300-1301	浏览器不支持	请更换Chrome浏览器后重试
1400	试玩时间已到期	请确认云手机订单信息
1401	用户退出云机（用户主动退出）	正常退出逻辑，无需处理
1402	当前云机已在其它设备登录，已自动退出云机	请确认是否有其他人获取到了云机信息导致抢占
1403	长时间未操作，已退出应用	正常退出逻辑，如果希望增加无操作等待时间，在createPlayer时配置noOperationThreshold为更大值
1406	等待Media数据流超时	请检查网络后重试
1412	用户退出云机（用户获取到另一台云机，此云机自动退出）	正常退出逻辑，无需处理
1415	云机信息错误	请检查串流时提供的实例ID是否正确，Token是否在有效期
3000	发生未知错误	请联系技术支持