

Android PaaS SDK 接入文档说明

版本变更记录

| | |
|--------|---|
| V1.0.0 | SDK初版 |
| V1.1.0 | <ul style="list-style-type: none">1、去掉重连，VcStreamListener新增onAbnormalDisconnected()回调2、startConnect 参数调整，去掉 cardWidth, cardHeight, cardDensity参数3、去除3.12 中setCardSize接口4、去除3.9 中setReconnecStreamCounts(int count)接口5、添加3.16 设置设备DeviceId接口6、开启预览图时,添加下发其他接口请求操作支持7、VcStreamListener新增视频流信息streamInfo()回调8、setVideoSize参数调整，去除dpi参数9、OperatorListener和PreviewOperatorListener增加requestId返回10、相关接口增加超时timeout参数 |
| V1.1.1 | <ul style="list-style-type: none">1、去除4.2.16,由SDK自动生成2、调整2.2中SDK初始化方式3、更新音视频SDK,优化三星等手机拉流花屏问题 |
| V2.0.0 | <ul style="list-style-type: none">1、恢复3.9 中设置重连次数setReconnecStreamCounts(int count)接口, SDK层监测到流断连后,会自动重连,不需要上层处理重连2、调整3.10 接口名称,以及封装键盘事件类型,传封装的类型事件即可3、新增3.15 首帧到达回调4、新增3.16 设置音频开关5、新增3.17 暂停推流6、新增3.18 恢复推流 |

| | |
|---------|--|
| | <ul style="list-style-type: none"> 7、新增3.19 设置编码方式 8、新增3.20 设置键盘类型 9、新增3.21 设置视频流画质 10、3.4拉流监听新增重连失败、重连成功等回调 11、新增4.2.17 启用应用 12、新增4.2.18 禁用应用 13、新增4.2.19 应用数据清理 14、新增4.2.20 实例屏幕截图 15、新增4.2.21 设备新机 16、新增5.2.1 文件上传接口 17、新增5.2.2 文件下载接口 18、新增5.2.3 实例截图文件下载 19、更新2.7 混淆配置 |
| V2.0.1 | <ul style="list-style-type: none"> 1、更新音视频SDK,优化暂停流时View空白问题 2、调整3.2 流渲染View和添加配置音视频SDK日志功能 3、新增3.22 设置视频流帧率接口 4、更新5.1 FileOperatorListener监听中新增文件操作进度值回调 5、调整3.21 设置视频流画质接口 6、新增3.23 自定义视频流画质接口 7、调整3.7 开始拉流接口，新增参数 8、废除3.11 设置最大码率接口 9、新增3.24 拉流资源释放接口 |
| V2.0.2 | <ul style="list-style-type: none"> 1、更新3.7 开始拉流接口,新增画质等参数 2、新增6.22消息订阅功能（屏幕亮度、剪切板透传） |
| V2.0.4a | <ul style="list-style-type: none"> 1、新增6.2.5 添加摇一摇接口 2、新增6.2.6 添加吹一吹接口 3、新增6.2.7 添加gps透传接口 |
| V2.0.4c | <ul style="list-style-type: none"> 1、新增6.2.8 显示导航栏接口 2、新增6.2.9 隐藏导航栏接口 3、新增6.2.10查看视频流横竖屏状态 4、新增6.2.11 获取导航栏状态接口 |

| | |
|--------|---|
| | 5、新增6.2.12 设置剪切板接口 6、更新音视频SDK,优化信令连接超时 |
| V2.0.5 | 1、新增6.2.13发送消息给云机应用接口 2、新增6.2.14 实例级别的屏幕截图接口 3、新增2.8 添加SDK需要的第三方库 |
| V2.0.6 | 1、更新3.3 设置视频流旋转方式 2、更新3.4 新添流分辨率改变回调 3、更新3.7 开始拉流接口,添加视频渲染参数 |

一、简介

SDK主要功能是为APP赋予云手机使用能力，可以通过SDK连接云手机，完成对云手机的一系列操作，包括SDK aar包、接入文档。

二、接入环境

2.1、集成sdk

SDK包含xxx.aar，请将aar文件复制到项目app\libslibs目录下

在app的build.gradle文件中进行引用:implementation fileTree(dir: 'libs', include: ['*.aar'])

2.2、sdk初始化（必须项, 请在主应用的application中初始化）

```
VcOperateManager.getInstance().init(Context)
```

注:主应用尽量不要使用SDK的日志工具，XLogUtil为SDK专用的日志工具，便于出现问题时定位

2.3 相关读写SD卡权限(权限申请由接入方申请，用于日志记录和文件相关操作接口, 必须项)

2.4 最小sdk 版本支持 24

2.5 音频和相机透传需要的相关权限

[复制代码](#)

```
<!-- 音频和相机透传相关权限-->
Manifest.permission.CAMERA,
Manifest.permission.RECORD_AUDIO
```

注意：摄像头穿透和麦克风穿透需要去申请相机和麦克风权限，否则功能无法使用，上面两个权限需要接入方在代码中进行权限动态申请

2.6 SDK日志查看方式

方式一：logcat | grep VCSdk

方式二：申请SDK读写权限成功后，SD卡根目录vc_sdk文件夹，可以查看SDK日志

2.7 混淆配置

[复制代码](#)

```
-dontwarn org.mediakit.R$layout
-dontwarn org.mediakit.R$drawable

-keepclasseswithmembernames class * {
    native <methods>;
}
-keep class org.webrtc.** { *; }
-keep class com.vc.videosdk.**{ *;}
-keep class org.mediakit.**{ *;}
```

2.8 添加SDK需要的第三方库(必须项, 否则SDK接口会异常)

[复制代码](#)

注意:为了避免和接入方的网络库冲突,因此,SDK使用的网络库由接入方引入,考虑到兼容性问题,
如果主项目中有使用下面网路库且版本小于下面的库,请务必保证至少升级到下面的版本
/*retrofit网络库*/

```
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.14.9'
```

三、拉流相关接入和操作说明

3.1 在拉流Activity页面xml布局中构建拉流VcSurfaceView和键盘透传需要的EditText, 这两个是必须项

▽ 复制代码

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/root_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/darker_gray"
    tools:ignore="MergeRootFrame">

    <EditText
        android:id="@+id/edt_keyboard_input"
        android:layout_width="@dimen/dp_1"
        android:layout_height="@dimen/dp_1"
        />

    <org.webrtc.SurfaceViewRenderer
        android:id="@+id/surfaceView_Renderer"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1" />
</FrameLayout>
```

3.2 获取 SurfaceViewRenderer

```
val mSurfaceViewRenderer = findViewById(R.id.surfaceView_Renderer)
```

注：获取的mSurfaceViewRenderer，在3.7 startConnect()时传入

创建音视频流相关设置接口需要的VcSurfaceView对象



复制代码

```
//音视频SDK日志回调，通过VcSurfaceView构造函数传入，如果不需要日志则传null
val mVcSurfaceView= VcSurfaceView(this,object :Loggable{
    override fun onLogMessage(message: String?, severity:
Logging.Severity?, tag: String?) {

    }
})
```

3.3 设置视频流旋转方式

由竖屏切换到横屏时，设置旋转方式,这里有两种旋转方式,根据需要,从以下接口中选择一个调用,必须调用

1、设置旋转上层的Activity

mVcSurfaceView?.setAutoRotaActivity(this)

2、设置旋转视频流

mVcSurfaceView?.setAutoRotaFlow()

3.4 设置拉流监听

mVcSurfaceView?.setVcStreamListener(this)



复制代码

```
interface VcStreamListener {
    //信令服务器Nats连接失败
    fun onDisconnected( peerName:String)

    /**
    *鉴权状态回调
    *peerName:连接名
    *code:状态码 (0:成功 1:失败)
    *descriptions:失败原因
    */
}
```

```
*/  
fun onAuthResult(peerName:String, code:Int, descriptions:String)
```

```
//音视频通道连接成功回调
```

```
fun onIceConnected(peerName:String)
```

```
//异常断开,回调给调用方
```

```
fun onAbnormalDisconnected()
```

注意：这里之前版本是由调用方去重连；新版本中,由SDK去执行重连,将重连结果回调给调用方

```
/**视频流信息
```

```
* frameWidth: 视频宽
```

```
* frameHeight:视频高
```

```
* frameRate:帧率-每秒接收的帧数
```

```
* delay:网络往返延迟
```

```
* receivePerSecond:视频流每秒接收的数据,单位:KB
```

```
* packetLossRate:丢包率
```

```
* decodeTime:解码时间,单位:毫秒
```

```
*/
```

```
fun streamInfo(  
    frameWidth: Int?,  
    frameHeight: Int?,  
    frameRate: String?,  
    delay: Float,  
    receivePerSecond: Int,  
    packetLossRate: Double,  
    decodeTime: String  
)
```

```
/**
```

```
* 重连成功
```

```
*/
```

```
fun reconnectSuccess()
```

```
/**
```

```
* 重连失败
```

```
*/
```

```
fun reconnectFail()
```

```
/**
```

* 设置音视频分辨率失败(调用3.13,如果失败,会回调,返回失败原因;成功则不回调)

```
*/
```

```
fun videoSizeSettingFail(reason:String)
```

```

    /**流分辨率改变回调
    * videoWidth:视频宽
    * videoHeight:视频高
    * rotation:旋转角度
    */
    fun onFrameResolutionChanged(videoWidth: Int, videoHeight: Int,
rotation: Int)

```

3.5 设置ice server（必须项，请向后台开发人员拿相关接口获取）

值格式: ip:port ， 例如: 183.62.127.92:3478

由接口返回ip和port, 然后按照上面的格式拼成iceServer值传给SDK, 用于音视频拉流

获取步骤:1、从后台提供的实例推流Token V2接口获取coturnHttp=coturnInfo.coturnHttp

2、用coturnHttp拼成http地址去访问, 返回ip和port

http:// +coturnInfo.coturnHttp+/api/v2/net/forward

示例:



复制代码

```
mVcSurfaceView?.setIceServer("183.62.127.92:3478")
```

注:这里的值需替换为自己从后台接口获取的值

3.6 设置键盘穿透需要的EditText

```
val mEdtKeyboard=findViewById(R.id.edt_keyboard_input)
```

```
mVcSurfaceView?.setCloubKeyEdt(mEdtKeyboard)
```

3.7 开始拉流(注意顺序, 前面6个接口要先在拉流前设置)

| 参数 | 必选 | 类型 | 说明 |
|---------------|----|--------|----------------|
| signalAddress | 是 | String | 信令服务器 pass- |

| | | | |
|----------------------|---|---------------------|---|
| | | | szsy.phone om:4432(仅 回为准) 来 源: /api/v m/token/ge 字段 |
| instanceId | 是 | String | 实例id 来 源: /api/v 接口instan |
| videoCodec | 是 | String | 视频编码 码: 传H264 VP8 |
| fps | 是 | Int | 帧率 值选择 35、40、45 |
| width | 是 | Int | 视频流宽 例 根据需要设 |
| height | 是 | Int | 视频流高 例 根据需要设置 |
| streamToken | 是 | String | 拉流鉴权to 来 源: /api/v2 oken/get接 |
| render | 是 | SurfaceViewRenderer | 音视频流渲 |
| forwardServerAddress | 是 | String | 转发服务器 来源:/api/v 接口public |
| ip | 是 | String | 实例ip 来源:/api/v2 ip字段 |

| | | | |
|--------------|---|-----------------|---|
| controlToken | 是 | String | 控制token 来源:/api/v2/i ken/get接口 |
| videoQuality | 是 | VideoQuality | 视频流画质 |
| scaleType | 否 | StreamScaleType | 设置渲染显示 1、SCALE 比例显示 2、SCALE 满显示 3、 SCALE_AS ;按真实宽高 |

注：videoCodec传VP8时,需要调用3.19，将编码格式设置为软编,目前VP8不支持硬编码

*视频流宽高组合:720*1280、1080*1920、540*960,正常用720*1280即可,云机默认分辨率为720*1280,

*具体可根据实际情况设置



复制代码

```
fun startConnect(
    signalAddress: String,
    instanceId: String,
    videoCodec: String,
    fps: Int,
    width: Int,
    height: Int,
    streamToken: String,
    render: SurfaceViewRenderer,
    forwardServerAddress: String,
    ip: String,
    controlToken: String,
    videoQuality: VideoQuality,
    scaleType: StreamScaleType
```

)

3.8 停止拉流

在退出页面onDestroy()中调用

```
mVcSurfaceView?.stopConnect()
```

3.9 设置重连次数(不设置，断连时，默认重连5次)

```
mVcSurfaceView?.setReconnectStreamCounts(int count)
```

3.10 发送键盘事件

```
mVcSurfaceView?.sendKeyEvent(event: KeyType) :Boolean
```



复制代码

KeyType有以下类型:

MENU, //菜单按键

HOME, //Home按键

BACK, //返回按键

VOLUME_UP, //音量加按键

VOLUME_DOWN //音量减按键

~~3.11 设置最大码率 (已废弃)~~

```
mVcSurfaceView?.setMaxBitrate(int rate)
```

~~3.12 拉流过程中更改卡分辨率 (已废弃)~~

```
setCardSize(int width, int height, int density)//density-dpi
```

3.13 拉流过程中更改视频流分辨率

```
mVcSurfaceView?.setVideoSize(int width, int height)
```

3.14 群控功能(即操作主屏，其他分屏联动，具体开发和方案请找后台开发人员对接)

3.15 首帧到达回调

`onFirstFrameRendered()`

3.16 设置音频开关（true 启用 false停用）

`mVcSurfaceView?.setAudioSwitch(isOpen: Boolean)`

3.17 暂停推流

`mVcSurfaceView?.pauseStream()`

3.18 恢复推流

`mVcSurfaceView?.restoreStream()`

3.19 设置编码方式

`mVcSurfaceView?.setEncodType(type: EndcodingType)`



复制代码

HARD, //硬编码
SOFTWARE//软编码

3.20 设置键盘类型

`mVcSurfaceView?.setKeyboardType(type: KeyboardType)`



复制代码

LOCAL, //真机键盘
CLOUD //云机键盘

3.21 设置视频流画质

`mVcSurfaceView?.setVideoQuality(quality: VideoQuality)`



复制代码

```
SMOOTH, //流畅
STANDARD_DEFINITION, //标清
HIGH_DEFINITION, //高清
SUPER_DEFINITION //超清
```

3.22 设置视频流帧率

`mVcSurfaceView?.setFrameRate(frameRate: Int)`

注意：frameRate最大上限值为3.7 startConnect()接口传入的fps值,即如果startConnect fps传入了50,那这里的frameRate上限值为不超过50,具体上限值根据实际使用场景传值即可

3.23 自定义视频流画质值

`mVcSurfaceView?.setCustomQuality(qualityValue: Int):String`

qualityValue范围建议：300-8000

返回结果：

成功：Success

失败：返回具体的失败原因

注：设置视频流画质值,正常情况,使用3.21 定义好的接口可满足视频流画质设置。**调用此接口,需要平台配置添加支持,否则将设置失败。**除非对画质有特殊需求,否则可忽略该接口。

3.24 拉流资源释放



复制代码

```
//退出页面,即在 onDestroy() 中执行以下接口,进行关闭拉流和释放资源
```

```
mVcSurfaceView?.release()
```

注：退出时务必记得调用该接口

四、实例批量操作接口

注:这里的接口支持批量操作,如果是拉流页面相关操作,请优先使用 六、消息订阅中的单个实例操作接口

4.1 总述:

- 1、SDK定义了一个操作管理类VcOperateManager, 给接入方统一调用
- 2、定义一个OperatorListener监听, 接入方调用接口后, 返回执行结果
- 3、请求时, instanceId用来判断连接的唯一性, 请严格按照实例的实际instanceId传入

注意：操作同一个实例时,以下接口为串行

监听示例:

```
/**
```

```
 * @Description : OperatorListener 将执行结果回调给调用方
```

```
instanceId--实例id
```

```
code--native代理返回的状态码
```

```
message--native代理返回的消息内容
```

```
requestId---每次请求时, SDK自动创建的id, 用来区别接口请求,操作接口异常时可用来定位问题
```

```
*/
```

```
interface OperatorListener {
```

```
    fun operatorResult(requestId:String,instanceId:String,code:Int, message:String)
```

```
}
```

| code | 说明 |
|------|-------------------------------|
| -2 | 远程端在超时时间内,未响应消息 |
| -1 | 关闭连接或连接发生了异常 |
| 0 | 请求成功 |
| 1 | 正常执行失败 |
| 2 | 请求数据解析失败 |
| 3 | 请求任务处理超时 |
| 4 | 使用了0作为request id |
| 9 | 未知异常 |
| 401 | Token Unauthorized token校验未通过 |

4.2 详细功能接口

4.2.1 启动应用

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |


```
VcOperateManager.getInstance().startApp(
    "1223",
    "192.168.134.11:49912"
    Constant.WEB_SOCKET_IP,
    Constant.WEB_SOCKET_TEST_TOKEN,
    "com.minhui.networkcapture",
    object : OperatorListener {
        override fun operatorResult(requestId: String,instanceId:
String, code: Int, message: String) {
            //调用方将在回调中拿到执行结果
        }
    })
```

4.2.2 停止应用

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/l ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/l ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/l ist接口ip字段 |
| token | 是 | String | 操作token |

| | | | |
|----------------------|---|------------------|---|
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |
| packageName | 是 | String | 包名 |
| timeout | 是 | Int | 超时时间,单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



复制代码

```
fun unInstallApp(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,//ws 连接鉴权token
    packageName: String,
    timeout: Int,//单位：毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)
```

4.2.4 隐藏应用

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|------------------|---|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/ ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



 复制代码

```
fun hideApp(  

```

```

instanceId: String,
forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
ip: String,
token: String,//ws 连接鉴权token
timeout: Int,//单位：毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
listener: OperatorListener
)

```

4.2.5 显示应用

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |
| timeout | 是 | Int | 超时时间,单位：毫秒 普通请求建议设 |

| | | | |
|-------------|---|------------------|---|
| token | 是 | String | 操作token来 源:/api/v2/instance/ control/token/get接 口token字段 |
| rootSwitch | 是 | Int | root开关 0禁用, 1启 用 |
| packageList | 是 | List<String> | 应用包名集合 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |


[复制代码](#)

```
fun rootApp(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,//ws 连接鉴权token
    rootSwitch: Int,//root开关 0禁用，1启用
    packageList: List<String>,//应用包名集合
    timeout: Int,//单位：毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)
```

4.2.7 全局应用root

| 参数 | 必选 | 类型 | 说明 |
|------------|----|--------|------|
| instanceId | 是 | String | 实例id |

| | | | |
|----------------------|---|------------------|---|
| | | | 来 源:/api/v2/instance/ ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| rootSwitch | 是 | Int | 0禁用, 1启用 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



复制代码

```
fun systemRoot(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,
    rootSwitch: Int,//0禁用，1启用
```

```
        timeout: Int, //单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外
        listener: OperatorListener
    )
```

4.2.8 开启预览图推流

这个接口和其他接口有区别, 开启后, 返回图片数据, 因此, 额外定义了一个 PreviewOperatorListener

[复制代码](#)

```
/**
 * @Description : PreviewOperatorListener 将预览图执行结果回调给调用方
 * requestId---每次请求时, SDK自动创建的id, 用来区别接口请求, 操作接口异常时可用
 * 来定位问题
 */
interface PreviewOperatorListener {
    fun operatorResult(requestId:String, instanceId:String, code:Int,
        message:String)
    fun previewData(bitmap:Bitmap) //实例预览图数据
}
```

| 参数 | 必选 | 类型 | 说明 |
|------------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddresses | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip |

| | | | |
|----------|---|-----------------------------|---|
| | | | 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| interval | 是 | String | 预览图推流图片每 秒数量，最低为1， 每秒1张图 |
| width | 是 | String | 预览图宽 |
| height | 是 | String | 预览图高 |
| quality | 是 | String | 预览图质量 |
| timeout | 是 | Int | 超时时间,单位：毫 秒 普通请求建议设 置超时时间30秒， 特殊接口比较耗时的 除外 |
| listener | 是 | PreviewOperatorList ener | 接口操作回调监听 |



复制代码

```

fun sysPreviewStart(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,//实例ip
    token: String,//ws 连接鉴权token
    interval: String,//预览图推流图片每秒数量，类型为 string。最低为1，每秒1
张图
    width: String,//预览图宽，类型为 string
    height: String,预览图高，类型为 string
    quality: String,//预览图质量，类型为 string值范围： 1-100

```

```

        timeout: Int,//单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外
        listener: PreviewOperatorListener
    )

```

复制代码

```

VcOperateManager.getInstance().sysPreviewStart(
    "1223",
    "192.168.134.11:49912"
    Constant.WEB_SOCKET_IP,
    Constant.WEB_SOCKET_TEST_TOKEN,
    "1",
    "640",
    "480",
    "90",
    object : PreviewOperatorListener {
        override fun operatorResult(instanceId: String, code: Int,
message: String) {
            XLogUtil.d("sysPreviewStart result
code:$code,,,message:$message")

        }

        override fun previewData(bitmap: Bitmap) {
            XLogUtil.d("返回的预览图数据 previewData bitmap:$bitmap")
            runOnUiThread {
                iv_picture?.setImageBitmap(bitmap)
            }
        }
    })

```

4.2.9 停止预览图推流(开启了预览图后, 退出/预览图不可见时, 记得调用以下接口关闭预览图, 切记!!!)

| 参数 | 必选 | 类型 | 说明 |
|------------|----|--------|-----------------------------------|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/I |

| | | | |
|----------------------|---|------------------|---|
| | | | ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



复制代码

```

fun sysPreviewStop(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,//ws 连接鉴权token
    timeout: Int,//单位: 毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)

```

4.2.10 虚拟定位

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|------------------|---|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/l ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/l ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/l ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| longitude | 是 | string | 经度 |
| latitude | 是 | string | 纬度 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |

[复制代码](#)

```
/**
 * 虚拟定位
 * ● longitude: 经度, 类型为 string。
 * ● latitude: 纬度, 类型为 string
 * 注: 云机系统用的是WGS-84坐标系, 故这里的经纬度应该是WGS-84坐标系,
 * 如果不是该坐标系需要进行转换
 */
fun virtualLocation(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址, 格式:
123.138.156.4:44912
    ip: String,
    token: String,//ws 连接鉴权token
    longitude: String,
    latitude: String,
    timeout: Int,//单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外
    listener: OperatorListener
)
```

4.2.11 剪切板

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip |

| | | | |
|----------------------|---|------------------|---|
| | | | 来 源:/api/v2/instance/ ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| width | 是 | string | 宽 |
| height | 是 | string | 高 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



复制代码

```
/**
 * 设置分辨率
 * ● width: 宽, 类型为 String
 * ● height: 高, 类型为 String
 */
```

```

fun sysResolution(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：123.138.156.4:44912
    ip: String,
    token: String,//ws 连接鉴权token
    width: String,
    height: String,
    timeout: Int,//单位：毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)

```

4.2.13 设置板卡DPI

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |

VcOperateManager.getInstance().getSdkVersion()

4.2.16 设置设备DeviceId

(用以同一个AK时多应用和多端连接支持，要求设置，不设置的话，SDK会默认给一个固定的DeviceId)

VcOperateManager.getInstance().setDeviceId(int deviceId)

注：用VcOperateManager 操作相关接口前设置，建议在Application中设置，由调用方获取设备唯一Id传入SDK。

4.2.17 启用应用

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|---|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/ ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |

| | | | |
|-------------|---|------------------|---|
| | | | ist接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/ist接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |
| packageName | 是 | String | 包名 |
| timeout | 是 | Int | 超时时间,单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



复制代码

```
/**
 *禁用应用
 */
fun disableApp(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,
    packageName: String,
    timeout: Int,//单位: 毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)
```

4.2.19 应用数据清理

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|------------------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |
| packageName | 是 | String | 包名 |
| timeout | 是 | Int | 超时时间,单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |



复制代码

/**

```

* 应用数据清理
*/
fun cleanAppData(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,
    packageName: String,
    timeout: Int,//单位：毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)

```

4.2.20 实例屏幕截图

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|--|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/ ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ |

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|------------------|---|
| instanceId | 是 | String | 实例id 来源:/api/v2/instance/list接口instanceId字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |
| brand | 是 | String | 品牌 |
| model | 是 | String | 型号 |
| timeout | 是 | Int | 超时时间,单位: 毫秒 普通请求建议设置超时时间30秒, 特殊接口比较耗时的除外 |
| listener | 是 | OperatorListener | 接口操作回调监听 |

```

fun sysNewDevice(
    instanceId: String,
    forwardServerAddress:String,//转发服务器地址，格式：
123.138.156.4:44912
    ip: String,
    token: String,
    brand: String,//品牌
    model: String,//型号
    timeout: Int,//单位：毫秒 普通请求建议设置超时时间30秒，特殊接口比较耗时的除外
    listener: OperatorListener
)

```

友情提示：如果使用 **VcOperateManager**操作相关接口异常，请把返回的**requestId**和 操作的接口记录下来，以及在**SDK** 卡根目录下的**vc_sdk**文件夹，拷贝日志,方便**SDK**定位问题。

例如：启动应用异常，那就把调用启动应用operatorResult回调返回的requestId记录



```

VcOperateManager.getInstance().startApp(
    instance_id,
    forwardServerAddress,
    ip,
    token,
    packageName,
    TIME_OUT_VALUE,
    object : OperatorListener {
        override fun operatorResult(
            requestId: String,
            instanceId: String,
            code: Int,
            message: String
        ) {
        }
    }
)

```

五、文件操作接口

5.1 总述

- 1、SDK定义了一个文件操作管理类VcFileUtil，给接入方统一调用

2、定义了一个FileOperatorListener文件操作监听，接入方调用接口后，返回执行结果

3、文件相关操作,涉及SD卡读写权限,请调用方务必保证申请了权限,否则接口将操作失败

4、文件操作涉及内存读写操作,故文件上传和下载调用方应该控制并发数量,建议1-3个,如果是大文件,建议串行单个执行

FileOperatorListener返回参数介绍

✓

复制代码

```
interface FileOperatorListener {  
    //filePath 下载文件时,下载成功会返回文件的保存路径  
    fun onSuccess(message:String,requestId:String,filePath:String)  
  
    fun onFail(message:String,requestId:String)  
    //上传和下载进度值  
    fun onProgress(progressValue:Int)  
}
```

5.2 详细功能接口

5.2.1 文件上传

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|--------|--|
| filePath | 是 | String | 要上传的源文件路径 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/ ist接口publicIp字段 |
| ip | 是 | String | 实例ip |

| | | | |
|------------|---|----------------------|---|
| | | | 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| uploadType | 是 | String | 1-上传到相册 2-上 传到sdcard |
| listener | 是 | FileOperatorListener | 接口操作回调监听 |



复制代码

```

/**文件上传
 * filePath:要上传的文件路径
 * token:用于身份验证的 token
 * ip:实例ip地址
 * uploadType: 1-上传到相册    2-上传到sdcard
 * forwardServerAddress:转发服务器地址 值格式: 183.62.127.91:44912
 */

fun uploadFile(
    filePath: String,
    token: String,用于身份验证的 token(实例控制token)
    ip: String,
    uploadType: String,
    forwardServerAddress:String,//转发服务器地址, 格式:
123.138.156.4:44912
    listener: FileOperatorListener
)

```

调用示例: VcFileUtil.getInstance().uploadFile()

5.2.2 文件下载

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|----------------------|---|
| filePath | 是 | String | 要下载的文件路径 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:/api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:/api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:/api/v2/instance/control/token/get接口token字段 |
| listener | 是 | FileOperatorListener | 接口操作回调监听 |


[复制代码](#)

```
/**
 * 文件下载
 * filePath:要下载的文件路径
 * token:用于身份验证的 token(实例控制token)
 * ip:实例ip地址
 * forwardServerAddress:转发服务器地址 值格式: 183.62.127.91:44912
 */
fun downloadFile(
    filePath: String,
    token: String,
    ip: String,
    forwardServerAddress:String,//转发服务器地址, 格式:
123.138.156.4:44912
    listener: FileOperatorListener
)
```

5.2.3 实例截图文件下载

| 参数 | 必选 | 类型 | 说明 |
|----------------------|----|----------------------|--|
| filePath | 是 | String | 要下载的文件路径 |
| forwardServerAddress | 是 | String | 转发服务器地址 来源:./api/v2/instance/list接口publicIp字段 |
| ip | 是 | String | 实例ip 来源:./api/v2/instance/list接口ip字段 |
| token | 是 | String | 操作token 来源:./api/v2/instance/control/token/get接口token字段 |
| listener | 是 | FileOperatorListener | 接口操作回调监听 |

[复制代码](#)

```
/**
 * 下载实例截图文件
 * filePath:要下载的文件路径,这里的文件路径,从4.2.20接口中获取
 * token:用于身份验证的 token(实例控制token)
 * ip:实例ip地址
 * forwardServerAddress:转发服务器地址 值格式: 183.62.127.91:44912
 *
 * 注意: 文件大小必须小于等于50M,不然会下载失败,该接口仅用于实例截图,
 * 文件下载请用5.2.2
 */
fun downloadScreenshotFile(
```

```
filePath: String,
token: String,
ip: String,
forwardServerAddress:String,
listener: FileOperatorListener
)
```

六、消息订阅

6.1 总述

连接上拉流后,可以通过建立连接,订阅相关消息,实现云机透传, 如云机屏幕亮度透传、云机剪切板透传等,**TopicManager**为消息订阅管理类,通过该类进行相关初始化和订阅操作,**TopicManager**操作对象为单个实例,在拉流页面使用,具体功能调用看详细功能接口。

友情提示：1、该消息通道为长连接通道,如果没退出拉流页面,监听到断连时,调用方需要进行重连

2、如果是从预览图进入拉流,得先关闭预览图通道,收到关闭预览图回调后再进入拉流

3、请在3.7 **startConnect()**的**onlceConnected()**回调中调用**buildConnect()**进行消息订阅(),即拉流成功才进行云机相关操作

6.2 详细功能接口

6.2.1 建立连接

| 参数 | 必选 | 类型 | 说明 |
|------------|----|--------|-----------------------------------|
| instanceId | 是 | String | 实例id 来 源:/api/v2/instance/l |

| | | | |
|----------------------|---|----------------------------|---|
| | | | ist接口instanceId 字段 |
| forwardServerAddress | 是 | String | 转发服务器地址 来 源:/api/v2/instance/ ist接口publicIp字 段 |
| ip | 是 | String | 实例ip 来 源:/api/v2/instance/ ist接口ip字段 |
| token | 是 | String | 操作token 来 源:/api/v2/instance/ control/token/get接 口token字段 |
| timeout | 是 | Int | 超时时间,单位: 毫 秒 普通请求建议设 置超时时间30秒, 特殊接口比较耗时的 除外 |
| listener | 是 | SubscribeTopicListe ner | 接口操作回调监听 |



复制代码

```

/*
*instanceId: 实例id
*forwardServerAddress: 转发服务器地址 值格式: 183.62.127.91:44912
*ip: 实例ip地址
*token:接口操作ws连接控制token,token从后台服务器获取
*listener:消息监听回调,接口操作结果由SubscribeTopicListener返回
*/
fun buildConnect(

```

```

        instanceId: String,
        forwardServerAddress: String,
        ip: String,
        token: String,
        listener: SubscribeTopicListener
    )

/**
 *SubscribeTopicListener接口返回参数说明
 *isConnected: false-连接失败/连接中断  true-连接成功
 *isSubSuccess:false  订阅失败    true-订阅成功
 *message:成功时为 "success",失败时为相应的错误信息
 *requestId:SDK内部生成的请求id,区分接口调用,用于排查问题
 */
interface SubscribeTopicListener {
    /*连接结果 isConnected:true 为建立连接通道成功,连接成功后调用消息订阅接口
    *这里需要注意: 连接上后,回调收到 isConnected:false, 此时为通道断开了,
    *如果此时还没退出拉流,还需要保持消息通道,则需要进行重连,即重新调用
    buildConnect
    */
    fun connectResult(isConnected:Boolean)
    //订阅屏幕亮度结果(用于云机屏幕亮度值透传到真机)
    fun subScreenBrightResult(isSubSuccess:Boolean,message:
String,requestId:String)
    //订阅剪切板透传结果(用于云机剪切板内容透传到真机)
    fun subClipboardTransmissResult(isSubSuccess:Boolean,message:
String,requestId:String)
    //返回云机屏幕亮度结果值
    fun screenBrightChange(value:Int)
    //返回云机剪切板内容值
    fun clipboardCallback(content:String)
    //云机返回的消息
    fun messageCallback(packageName:String,message:String)
}

```

6.2.2 消息订阅



复制代码

注: 以下接口,根据接入方自身需求,选择调用,需要的功能才进行订阅

//订阅云机屏幕亮度透传

TopicManager.getInstance().subscribeScreenBrightnessTopic()

//订阅云机剪切板透传

```
TopicManager.getInstance().subscribeClipboardTopic()  
//订阅消息透传(云机应用和真机客户端互发消息)  
TopicManager.getInstance().subscribeMessageTransmissTopic()
```

6.2.3 消息订阅使用示例

复制代码

```
TopicManager.getInstance().buildConnect(  
    mInstanceId,  
    forwardServerAddress,  
    mIp,  
    token,  
    object : SubscribeTopicListener {  
  
        override fun connectResult(isConnected: Boolean) {  
            XLogUtil.d("${TAG}connectResult  
isConnected:$isConnected,,isNeedReConnectSubscribeChannel:$isNeedReConn  
ectSubscribeChannel,,connectNum:$connectNum")  
            if (isConnected) {  
                connectNum = 1  
                注意:要isConnected=true,即连接上才能进行订阅和其他接口操作,  
否则将操作失败  
                //订阅屏幕亮度  
  
TopicManager.getInstance().subscribeScreenBrightnessTopic()  
                //订阅剪切板透传  
                TopicManager.getInstance().subscribeClipboardTopic()  
                //订阅消息透传(云机应用和真机客户端互发消息)  
  
TopicManager.getInstance().subscribeMessageTransmissTopic()  
            } else {  
                //这里重连逻辑仅供参考,具体逻辑调用方根据需要开发  
                if (isNeedReConnectSubscribeChannel) {  
                    connectNum++  
                    if (connectNum <=5) {  
                        buildSubscribeConnect()  
                    } else {  
                        XLogUtil.d("${TAG}buildSubscribeConnect fail,已达  
到重连次数 connectNum:$connectNum")  
                    }  
                }  
            }  
        }  
    }  
}
```

```

/**
 * 屏幕亮度值改变
 */
override fun screenBrightChange(value: Int) {
    //实例屏幕亮度变化返回的亮度值,拿到该值进行真机亮度值设置

}

/**
 * 云机操作了复制/剪切操作,剪切板返回
 */
override fun clipboardCallback(content: String) {
    XLogUtil.d("${TAG}clipboardCallback content:$content")

    //将云机实例复制、剪切的内容设置到真机剪切板
    val clipboardManager =
getSystemService(Context.CLIPBOARD_SERVICE) as ClipboardManager
    val clip = ClipData.newPlainText("simple text", content)
    clipboardManager.setPrimaryClip(clip);
}

/**
 * 订阅剪贴板透传结果
 */
override fun subClipboardTransmissResult(
    isSubSuccess: Boolean,
    message: String,
    requestId: String
) {
    XLogUtil.d("${TAG}subClipboardTransmissResult
isSubSuccess:$isSubSuccess,,,message:$message,,,requestId:$requestId")

}

/**
 * 订阅屏幕亮度结果
 */
override fun subScreenBrightResult(
    isSubSuccess: Boolean,

```

```

        message: String,
        requestId: String
    ) {
        XLogUtil.d("${TAG}subScreenBrightResult
isSubSuccess:$isSubSuccess,,,message:$message,,,requestId:$requestId")
    }
}

```

6.2.4 关闭连接（不使用时记得关闭连接通道!!!）



复制代码

```

override fun onDestroy() {
    TopicManager.getInstance().disconnect()
}

```

6.2.5 设置吹一吹



复制代码

```

/**
 * 云机模拟吹一吹功能
 * value: 取值范围 int 5-10秒,单位:秒
 * isSuccess:true-设置成功 false-设置失败
 * message: 成功时为 "success",失败时为相应的错误信息
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */
fun setttingBlow(value:Int)
调用示例:

TopicManager.getInstance().setttingBlow(value,object:StreamOperatorListe
ner{
    override fun onCallback(
        isSuccess: Boolean,
        message: String,
        requestId: String
    ) {

    }

})

```

6.2.6 设置摇一摇

复制代码

```
/**
 * 云机模拟摇一摇功能
 * data: 取值范围 int 3-10秒
 * isSuccess:true-设置成功 false-设置失败
 * message: 成功时为 "success",失败时为相应的错误信息
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */
fun setttingShake(value:Int)
调用示例:

TopicManager.getInstance().setttingShake(value.toInt(),object:StreamOperatorListener{

    override fun onCallback(
        isSuccess: Boolean,
        message: String,
        requestId: String
    ) {

    }

})
```

6.2.7 设置GPS

复制代码

```
/**
 * 设置云机GPS
 * isSuccess:true-设置成功 false-设置失败
 * message: 成功时为 "success",失败时为相应的错误信息
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 * 注: 云机系统用的是WGS-84坐标系,故这里的经纬度应该是WGS-84坐标系,
 * 如果不是该坐标系需要进行转换
 */
fun setttingGps(longitude: String, latitude: String)
调用示例:

TopicManager.getInstance().setttingGps(longitude,latitude,object
:StreamOperatorListener{

    override fun
onCallback(
```

```

Boolean,
String,
String
回结果

isSuccess:
message:
requestId:
) {
    //设置Gps返
}
})

```

6.2.8 显示导航栏



复制代码

```

/**
 * 云机显示导航栏
 * isSuccess:true-设置成功 false-设置失败
 * message: 成功时为 "success",失败时为相应的错误信息
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */

TopicManager.getInstance().displayBar(object :StreamOperatorListener{
    override fun onCallback(isSuccess: Boolean, message:
String, requestId: String) {

    }
})

```

6.2.9 隐藏导航栏



复制代码

```

/**
 * 云机隐藏导航栏
 * isSuccess:true-设置成功 false-设置失败
 * message: 成功时为 "success",失败时为相应的错误信息
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */

```

```

TopicManager.getInstance().hideBar(object :StreamOperatorListener{
    override fun onCallback(isSuccess: Boolean, message:
String, requestId: String) {

        }
    })

```

6.2.10 查看视频流横竖屏状态



复制代码

```

/**
 * 获取屏幕方向
 *message: 成功时为 "success",失败时为相应的错误信息
 *orientation: 方向, 类型为 int
 * 0-竖屏
 * 1-横屏
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */

TopicManager.getInstance().getScreenOrientation(object:ScreenOrientation
Listener{
    override fun onResult(orientation: Int, message:
String, requestId: String) {

        }
    })

```

6.2.11 获取导航栏状态



复制代码

```

/**
 *获取导航栏状态
 *message: 成功时为 "success",失败时为相应的错误信息
 *status 状态值:0显示, 1隐藏
 * requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */

TopicManager.getInstance().getBarStatus(object:BarStatusListener{
    override fun onResult(
        status: Int,
        message: String,

```

```

        requestId: String
    ) {

    }
}
})

```

6.2.12 设置剪切板(实例级别)



复制代码

```

/**
 *设置剪切板(将真机的剪切板内容设置到云机)
 *isSuccess:true-设置成功 false-设置失败
 *message: 成功时为 "success",失败时为相应的错误信息
 *requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 */
TopicManager.getInstance().setClipboard(clipboardContent,object:StreamOp
eratorListener{
    override fun onCallback(
        isSuccess: Boolean,
        message: String,
        requestId: String
    ) {

    }

}
})

```

6.2.13 发送消息给云机应用



复制代码

```

/**
 *packageName:云机应用的包名
 *message:要发送的消息内容
 *注: 如果需要在集成SDK的客户端发送消息给云机的某一个应用,该云机应用需要额外集成
 *AIDL服务端,如果不需要该功能,可忽略该接口
 */
fun sendMessageToApp(packageName: String, message: String)
调用示例:
TopicManager.getInstance().sendMessageToApp(packageName, message)

```

6.2.14 实例屏幕截图(实例级别)



复制代码

```
/**
 *参数说明
 *quality-图片质量:值范围: 1-100
 *注:不建议设置太小,否则太模糊,一般设置100即可
 * isSuccess:true-设置成功 false-设置失败
 *message: 成功时为 "success",失败时为相应的错误信息
 *requestId:SDK内部生成的请求id,用于区分接口调用,用于排查问题
 *filePath:截图保存的文件路径
 *友情提示: 截图需要存储写权限,由接入方申请,否则截图失败
 */

fun sysScreenshot(quality: Int,listener: ScreenShotListener)

调用示例:
TopicManager.getInstance().sysScreenshot(100, object:
ScreenShotListener{
    override fun onResult(
        isSuccess: Boolean,
        message: String,
        requestId: String,
        filePath: String
    ) {

    }

})
```