目录

[一、 对外接口 2](#_Toc10722)

[1、 回调监听 2](#_Toc17903)

[2、 初始化 2](#_Toc31738)

[3、 登录 2](#_Toc16240)

[4、 支付 2](#_Toc30653)

[5、 退出游戏 2](#_Toc27773)

[6、 切换帐号 2](#_Toc1079)

[7、 提交角色信息 3](#_Toc11848)

[8、 各生命周期接口。 3](#_Toc8906)

[二、 内部逻辑与功能 4](#_Toc23001)

[1、 全局application入口-SophixStubApplication 4](#_Toc28891)

[2、 原入口JMApplication 4](#_Toc18807)

[3、 初始化 5](#_Toc12328)

[4、 登录接口 5](#_Toc3638)

[5、 手机号登录 6](#_Toc21146)

[6、 帐号登录 6](#_Toc27380)

[7、 游客登录 6](#_Toc11721)

[8、 一键登录 7](#_Toc19536)

[9、 浮窗 7](#_Toc23444)

[10、 支付接口 7](#_Toc19818)

[11、 切换（注销） 8](#_Toc14454)

[12、 退出游戏 8](#_Toc5980)

[三、 数据log\_SDK（广点通/头条/百度） 9](#_Toc13693)

[1、 调用时机。 9](#_Toc6611)

[2、 内部逻辑 9](#_Toc8979)

[3、 存放目录 10](#_Toc19770)

[四、 穿山甲广告sdk 11](#_Toc763)

[1、 调用时机 11](#_Toc29512)

[2、 回调结果 11](#_Toc8997)

[3、 打包存放 11](#_Toc12530)

# 对外接口

此处仅声明，具体调用参数请参考对外接入文档

## 回调监听

ApiListenerInfo

## 初始化

JiMiSDK.*initInterface（*context，appId,appKey,InitListener*）*

## 登录

JiMiSDK.*login*(context, **appId**, **appKey**,ApiListener);

## 支付

JiMiSDK.payment(context, paymentInfo, ApiListenerInfo)

## 退出游戏

JiMiSDK.*exit*(context, ExitListener)

## 切换帐号

JiMiSDK.*switchAccount*

## 提交角色信息

JiMiSDK.*setExtData*(context,type,roleid,rolename,level,gender,serverno,zoneName,balance,power,viplevel,roleCTime,roleLevelMTime,ext);

## 各生命周期接口。

onCreate(Activity activity)

onStop(Activity activity)

onPause(Activity activity)

onResume(Activity activity)

onRestart(Activity activity)

onDestroy(Activity activity)

~~onStart(Activity activity)~~ 【无该常规周期】

onNewIntent(**final** Intent intent)

onActivityResult(Activity activity, **int** requestCode, **int** resultCode, Intent data)

# 内部逻辑与功能

## 全局application入口-SophixStubApplication

a.实际上在热修复版本以后，jmsdk项目或者经由打包脚本后入口application是ophixStubApplication阿里的热修复类。

b.在该类attachBaseContext初始化阿里热修复的jar

c.在该类中通过注解的方式引入原本的application类@SophixEntry（），等于在入口application以前多做了ophixStubApplication里面的操作，然后才走原本的入口JMapplication代码。

## 原入口JMApplication

1. processName 进程判断

*因为项目集成了友盟推送，友盟推送自己会创建一个进程，加上app默认的进程，一共就有2个进程了，导致Application的onCreate执行2次，线程也就创建了2次。*

所以需要判断是否当前进程是自己的进程才走oncreate逻辑。

1. 友盟推送

通过getPushParam判断mainfest清单有无友盟参数（打包时需要友盟推送会给参数打进包的清单里），有参数则初始化友盟，无参数则默认走友盟统计。

1. 推送注意事项，在调用友盟推送时，如我们的代码在com.jmhy.sdk包，默认是用包名如com.qyz去寻找代码时会找不到导致初始化失败，需要在初始化友盟时setResourcePackageName设置资源包名
2. 用包里aseets目录的**jmhy.properties**的appid、appkey去全局初始化数据sdk

## 初始化

1. 字段*init*重复初始化判断，初始化只执行一次，请求拿到返回成功后该字段改true，后续即使再次被调用初始化接口也不再继续执行。
2. 权限申请，进入权限申请页面PermissionActivity，首次拒绝申请后会再次点页面上的授权会直接跳转到设置界面去设置
3. 权限全授予后进行下一步，初始化JLibrary、获取oaid。
4. Debug调试模式判断，isDebugMode判断是调试模式会弹出内容提示toast在页面上。
5. WebAPI接口初始化，各请求的url、配置装填
6. Appid、key非空判断、supportEnglish是否支持英文（语言国际化）
7. 发起初始化请求，是否安装qq/wx/ali检测、模拟器检测（蓝牙、光传感器、部分设备信息中的特殊参数、CPU、模拟器特有文件、系统原生属性等）
8. 初始化请求成功。拿到的各字段赋予客户端，皮肤、弹窗地址、在线上报时间等。数据sdk初始化，根据字段ali\_hot\_fix请求热修复补丁包。开启pushService做轮询在线上报

## 登录接口

a.检查是否初始化过，未初始化则不再继续往下执行

b.调试模式检查，是调试模式则开始弹出toast提示日志

c.选择登录

**is\_reg\_login\_on**字段，服务端的登录开关，如果为0则关闭登录。

*isChangeGuestAccount*浮窗改密码/游客改帐号后调登录进入，后续在切换那统一说明

本地缓存有数据，根据*is\_auto\_login\_on*字段(自动登录开关）跳入自动登录或者普通登录页面

1. 登录页面。根据初始化返回的皮肤字段，装填自动/普通登录的皮肤（目前9套，有部分弃用的）
2. 自动登录。Is\_auto\_login\_on开关打开的情况，根据变量判断跳转到自动登录还是普通登录页。自动登录页根据皮肤装载对应UI，根据isGuestSwitchAccount字段区分是否改号而来的。

普通自动登录读取帐号列表里最近登录的第一条帐号显示为用户名xxx，欢迎进入游戏。

如果是改完帐密注销来的自动登录，用该字段的用户名登录。

1. 普通登录。根据*ismobillg字段*判断进入对应皮肤的登录首页还是自动登录失败而来跳入帐号登录节目。

## 手机号登录

a.皮肤9在首页输入完手机号点登录先到输入验证码页面。其他1-8皮肤在当前页面输密码不用跳转。

b.登录回调收到后根据字段upass返回有数据则为第一次注册，跳转到注册页面，设置密码，发起注册请求。 无数据则为正常手机号登录。

c.登录请求成功。保存账密、登录方式和时间等数据到本地。保存弹窗url。跳转到欢迎登录。根据字段控制弹出浮窗。装载webview。

## 帐号登录

a.校验账号密码输入框非空，发起登录请求。登录成功保存账密、时间、登录方式等信息。

设置url\_after\_login登录弹窗url字段。回调游戏登录成功。

b.皮肤1-8进来帐号登录框(皮肤9无此需求)，检查本地是否有缓存的账号密码，如果有会自动填进账密框，密码为\*显示。可直接点登录

## 游客登录

a.点游客登录按钮直接触发游客登录请求，本台设备信息如已在后台注册过帐号（且未解绑），则服务端视该设备已注册过，是登录请求，upass字段无内容。 去后台解绑、或者新设备点游客按钮，会返回upass字段，服务端分配账号密码、跳转到保存账密截图页面。而后进入登录

## 一键登录

a.本机手机号一键登录。阿里云分配的token。如有双卡槽。用的是当前的数据卡槽的手机号。调用一键登录方法拿到阿里云的token，提给服务端走一键登录接口（参数与自动登录接近）

b.一键登录失败。关闭一键登录UI。回到极米登录UI的fragment。

## 浮窗

a.登录成功后左侧显示浮标。皮肤1-8点击浮标为展开椭圆选择条，有3个账号/福利/客服按钮可供点击打开居中的悬浮窗。皮肤9为贴屏幕左侧悬浮窗。右侧有关闭按钮。

b.浮标拖动时右下角出现隐藏区域，拖进隐藏区域会震动提醒。在此区域离开up事件弹出对话框 可以隐藏浮标。

c.加载gif图，js事件监听，适配软键盘、设置透明度等操作

d.activity→动态添加界面 activity跳转周期会导致打开浮窗时游戏（底部）界面暂停。改用动态添加不再暂停。

e.浮标根据*float\_icon\_url*给的icon字段，如果无返回则用包里的浮标icon，如果有返回则用本地保存的gif图，加载该gif图当浮标。

## 支付接口

a.创建订单。paymentInfo中根据游戏传来的价格、商品等信息。到服务端先创建订单。成功后有订单号，以供后续支付请求用

b.支付页面。创建订单成功后展示支付页面，webview加载支付地址。加载过程有gif加载过度。Webview过滤js跳转微信（**weixin://wap/pay**）/支付宝（**alipays://platformapi/startapp**）

c.js方法中收到结果后回调游戏支付成功/失败/取消

## 切换（注销）

a.方法名是切换，其实是个注销接口。调用会退出当前帐号。请求注销接口

## 退出游戏

a.有对手机返回键做监听，按返回键或者手动调用该方法弹出退出对话框，点确定先给游戏通知退出。然后结束进程。

# 数据log\_SDK（广点通/头条/百度）

## 调用时机。

a.初始化激活：全局application oncreate 里首次执行数据sdk初始化，使用的参数为apk包里（打包时写入）的**jmhy\_config.json** 文件的参数。

b.注册上报：各皮肤的游客注册、手机号注册、帐号注册请求回调成功时，调用数据sdk注册上报register方法。

c.注册补报：登录成功后，返回的字段里is\_package\_new字段为1时说明有漏报的，发起数据sdk的注册上报

d.登录上报：所有登录逻辑中，回调CP通知登录结果时，会统一在这方法中，调用用数据sdk的登录上报

e.支付上报：web的js回调中，支付通知成功/失败，调用数据sdk支付方法进行上报，参数支付信息里取，成功/失败发起调用时的状态为true/false区分

f.支付去重：因有过重复上报的情况，每次调用数据sdk支付上报时会用个变量存储订单号，当前订单号不等于存储的订单号时上报。

g.支付补报：初始化成功后开启的online轮询中，Channel\_event字段有数据时，说明需要补充支付上报。用服务端返的支付参数channel\_event调用数据sdk支付上报的方法，发起支付上报

## 内部逻辑

a.在jmsdk中根据jmhy\_config.json文件（打包生成） 的jm\_log\_sdk字段来判定，是gdt/jrtt/bd的数据sdk，调用该文件内传的类名

b.数据上报：在数据sdk初始化、注册、登录、支付及部分生命周期方法中，除了调用数据sdk的方法通知广点通/头条/百度，还会调用数据上报接口HOST + **/report/channel** 通知我方服务端

## 存放目录

1. jar：

通过gradle生成log\_sdk的jar，内部就放着2个类，api网络封装调用/report/channel的类及调用第三方（头条/广点通/百度）数据sdk的类。

1. 打包存放：

打包时jar文件放在log\_sdk目录，如其他sdk一样，复制第三方jar、res、assets、mainfest等资源及我方生成的jar，一一对应放入logsdk目录下。

这部分资源会根据打包时后台给的参数打对应数据sdk进包里

# 穿山甲广告sdk

## 调用时机

极米sdk中，web的js方法loadAd被调用时会调用它。用web端给的广告appid初始化广告SDK。用web端给的广告id加载广告。

## 回调结果

广告播放完毕、准备、加载等各节点会有回调。调用web端开放的方法**adEndCallBack**通知结果

## 打包存放

打包时存放在ad\_sdk中，方法与其他sdk存放一样。构建jar、拷贝资源、根据参数打包打入sdk中