

九游游戏_SDK_开发参考说明书_Android API_2023（简化版）

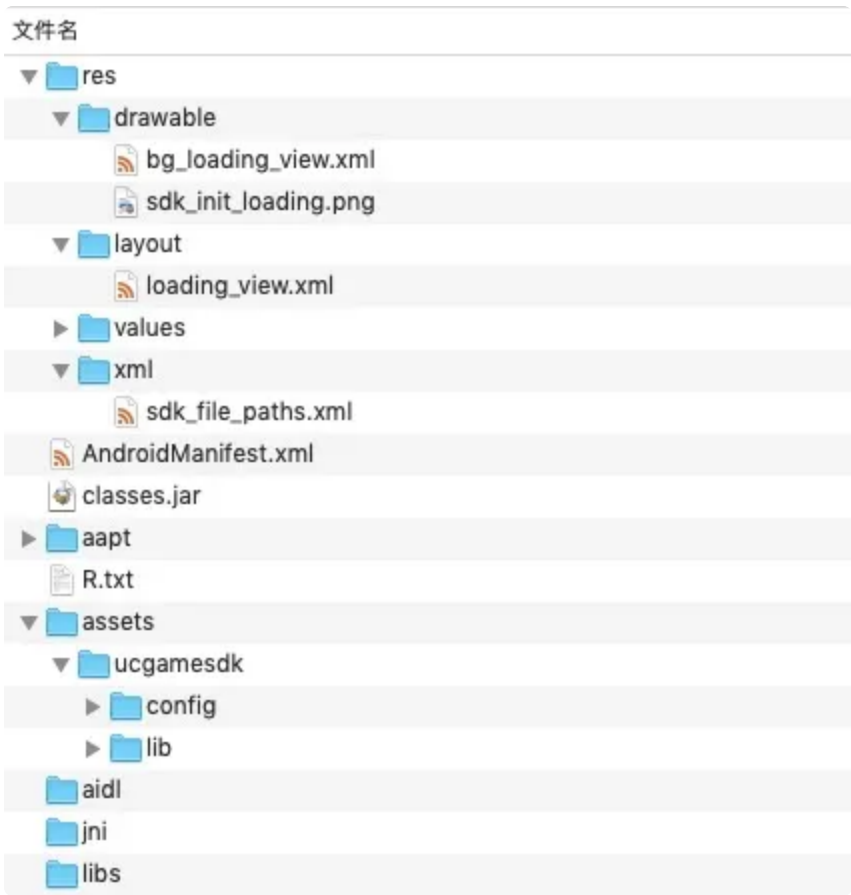
1. 开发环境搭建

以下以Eclipse和Android Studio为例，说明在开发游戏时使用九游游戏SDK的环境设置：

强烈推荐使用Android Studio方式接入 aar 包

1.1 Eclipse

将九游游戏SDK接入依赖库包中的 net-sdk-x.x.x.aar 解压，将得到以下目录：

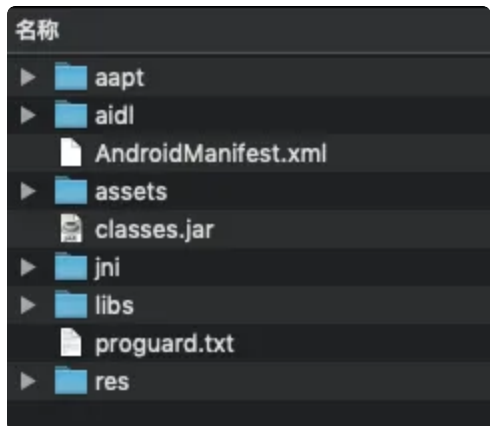


其中游戏工程需要使用的资源，说明如下：

- res目录下，全部拷贝到游戏工程的res目录下；

- classes.jar拷贝到游戏工程libs目录下；
- assets目录下的文件或文件夹拷贝到游戏工程的assets目录；

将alipaySdk-x.x.x-xxxxx.aar文件解压，将得到以下目录：



其中游戏工程需要使用的资源，说明如下：

- classes.jar 改名成 alipaySdk.jar，拷贝到游戏工程的libs目录下；
- proguard.txt 里面的内容，复制到项目的混淆文件中

1.2 Android Studio

将九游游戏SDK接入包中的net-sdk-x.x.x.aar和alipaySdk-x.x.x-xxxxx.aar文件和ugpsdk-xxx.aar拷贝到游戏工程的aar libs目录并修改游戏Module的build.gradle文件，添加repositories域：

```
1 repositories {  
2     flatDir {  
3         dirs 'aar'  
4     }  
5 }
```

如果已经存在了repositories域，可添加到已存在的域中即可

为dependencies域添加打包aar的配置：

```
dependencies {
    // 其他配置...

    // 将libs目录下的aar文件打入游戏apk包
    fileTree(dir: 'libs', include: '**/*.aar')
        .each { File file ->
            dependencies.add("compile", [name: file.name.lastIndexOf('.').with {
                it != -1 ? file.name[0..

```

蓝线框内为必填项

注：aar接入方式时，2个aar内的Androidmanifest.xml文件已经包含SDK接入所需的权限及activity声明，无需在游戏工程再次写入，否则造成冲突； 但其中“cn.uc.gamesdk.activity.PullupActivity”的 data android:scheme 需要根据不同游戏配置不同的ng+gameId，故此activity需要在游戏manifest内重新写入，并通过配置tools:node="replace" 来覆盖aar的manifest配置。

1.3 修改AndroidManifest.xml文件

打开您项目的AndroidManifest.xml文件，进行如下修改：

指定游戏图标需使用带有九游角标的图标；

包名加上“.aligames后缀”（已在九游上线过的游戏，保持原包名不改）；

1. 增加权限声明（参见AndroidManifest.xml文件示例中所示的权限为SDK所必需的，游戏可根据需要额外增加其它权限声明）；
2. 删除1个旧alipay所用声明 com.alipay.sdk.auth.AuthActivity
3. 增加2个alipay所用的activity声明com.alipay.sdk.app.PayResultActivity、com.alipay.sdk.app.AlipayResultActivity
4. 增加 *cn.gundam.sdk.shell.activity.ProxyActivity* 的声明(注：launchMode请保持standard或者不设置)。
5. 增加 *cn.gundam.sdk.shell.activity.ThemeProxyActivity* 的声明(注：launchMode请保持standard或者不设置)。
6. 增加 *cn.gundam.sdk.shell.content.FileProvider* 的声明
 - a. 检查 android:authorities="\${applicationId}.gamesdk.fileprovider"的最终打包输出
 - b. 正常情况下，*\${applicationId}* 会被自动替换成游戏的包名
7. 设置supports-screens 节点；
8. 增加 *cn.uc.paysdk.SDKActivity* 的声明
9. 增加 *cn.uc.paysdk.service.SDKService* 的声明

10. 增加 `cn.gundam.sdk.shell.service.ResProxyService` 的声明

修改后的AndroidManifest.xml文件如下所示：

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3 package="cn.lori.fighter.aligames" android:versionCode="1" android:vers
4 ionName="1.0"
5
6 <supports-screens
7     android:anyDensity="true"
8     android:largeScreens="true"
9     android:normalScreens="true"
10    android:resizeable="true"
11    android:smallScreens="true" />
12
13    <uses-sdk android:minSdkVersion="14" android:targetSdkVersion="26" />
14
15    <uses-permission android:name="android.permission.INTERNET" />
16    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"
17 />
18    <uses-permission android:name="android.permission.ACCESS_NETWORK_STAT
19 E" />
20    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STOR
21 AGE" />
22    <uses-permission android:name="android.permission.READ_PHONE_STATE" /
23 >
24    <uses-permission android:name="android.permission.GET_TASKS" />
25    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDO
26 W" />
27    <uses-permission android:name="android.permission.RECORD_AUDIO"/>
28    <uses-permission android:name="android.permission.MODIFY_AUDIO_SETTIN
29 GS"/>
30
31    <uses-permission android:name="android.permission.READ_SETTINGS" />
32    <uses-permission android:name="android.permission.WRITE_CALENDAR" />
33    <uses-permission android:name="android.permission.REQUEST_INSTALL_PAC
34 KAGES" />
35    <uses-permission android:name="com.android.alarm.permission.SET_ALAR
36 M" />
37    <uses-permission android:name="android.permission.READ_EXTERNAL_STORA
38 GE" />
39
40    <queries>
41        <package android:name="cn.ninegame.gamemanager" />
42    </queries>
```

```

36     <application android:icon="@drawable/ic_launcher_uc" android:label="@
37     string/app_name" >
38         .....
39         .....
40         <activity
41             android:name="cn.gundam.sdk.shell.activity.ProxyActivity"
42             android:configChanges="keyboardHidden|orientation|screenSize"
43             android:label="@string/app_name"
44             android:theme="@android:style/Theme.Translucent.NoTitleBar"
45             android:windowSoftInputMode="adjustResize" >
46             <intent-filter>
47                 <action android:name="cn.uc.gamesdk.sdkweb" />
48                 <category android:name="android.intent.category.DEFAULT"
49             />
50             </intent-filter>
51         </activity>
52
53         <activity
54             android:name="cn.gundam.sdk.shell.activity.ThemeProxyActivit
55             y"
56             android:configChanges="keyboardHidden|orientation|screenSize"
57             android:label="@string/app_name"
58             android:theme="@android:style/Theme.NoTitleBar"
59             android:windowSoftInputMode="adjustResize">
60             <intent-filter>
61                 <action android:name="cn.uc.gamesdk.sdkweb"/>
62                 <category android:name="android.intent.category.DEFAULT"/
63             >
64             </intent-filter>
65         </activity>
66
67         <!-- 请检查当前打包配置中 applicationId 是否为正确的游戏包名，如游戏包名为cn.u
68         c.gamesdk.demo -->
69         <!-- 打包结果应该是： android:authorities="cn.uc.gamesdk.demo.gamesdk.fi
70         leprovider" -->
71         <!-- 如有需要可以手动替换 android:authorities="${applicationId}.gamesdk.
72         fileprovider" 中的${applicationId}为游戏包名 -->
73         <provider
74             android:name="cn.gundam.sdk.shell.content.FileProvider"
75             android:authorities="${applicationId}.gamesdk.fileprovider"
76             android:exported="false"
77             android:grantUriPermissions="true"
78             tools:replace="android:authorities">
79             <meta-data
80                 android:name="android.support.FILE_PROVIDER_PATHS"
81                 android:resource="@xml/sdk_file_paths"/>
82         </provider>

```

```

77      <!-- android:taskAffinity 填上游戏的包名, 如游戏包名为cn.uc.gamesdk.d
emo, 则下面填 cn.uc.gamesdk.demo.diff -->
78      <!-- data android:scheme 里填上"ng+当前游戏的gameId",如游戏ID是12345
79      6,则填上ng123456 -->
80      <activity
81          android:name="cn.uc.gamesdk.activity.PullupActivity"
82          android:theme="@android:style/Theme.Translucent"
83          android:taskAffinity="游戏包名.diff"
84          android:excludeFromRecents="true"
85          android:label="PullupActivity"
86          android:launchMode="singleTop"
87          android:exported="true"
88          tools:node="replace">
89          <intent-filter>
90              <action android:name="android.intent.action.VIEW"/>
91              <category android:name="android.intent.category.DEFAULT"
92              E"/>
93              <category android:name="android.intent.category.BROWSABL
94              <data android:scheme="ng游戏ID" />
95          </intent-filter>
96      </activity>
97      <!-- ucpay sdk begin -->
98      <activity
99          android:name="cn.uc.paysdk.SDKActivity"
100          android:configChanges="keyboardHidden|orientation|screenSize"
101          android:theme="@android:style/Theme.Translucent.NoTitleBar"
102          android:windowSoftInputMode="adjustPan|stateAlwaysHidden|adju
103          stResize" />
104
105      <service android:name="cn.uc.paysdk.service.SDKService" />
106      <!-- ucpay sdk end -->
107      <!-- alipay sdk begin -->
108      <activity
109          android:name="com.alipay.sdk.app.H5PayActivity"
110          android:configChanges="orientation|keyboardHidden|navigation"
111          android:exported="false"
112          android:screenOrientation="behind" >
113      </activity>
114      <activity
115          android:name="com.alipay.sdk.app.H5AuthActivity"
116          android:configChanges="orientation|keyboardHidden|navigation"
117          android:exported="false"
118          android:screenOrientation="behind" >
119      </activity>
120      <activity
121          android:name="com.alipay.sdk.app.PayResultActivity"
122          android:configChanges="orientation|keyboardHidden|navigation"

```

```

120         android:exported="true"
121     android:launchMode="singleInstance"
122     android:theme="@android:style/Theme.Translucent.NoTitleBar" >
123         <intent-filter>
124             <category android:name="android.intent.category.DEFAULT"
125             />
126         </intent-filter>
127     </activity>
128     <activity
129         android:name="com.alipay.sdk.app.AlipayResultActivity"
130         android:exported="true"
131         android:launchMode="singleTask"
132         android:theme="@android:style/Theme.Translucent.NoTitleBar" >
133     </activity>
134     <!-- alipay sdk end -->
135
136     <service
137         android:name="cn.gundam.sdk.shell.service.ResProxyService"
138         android:enabled="true"
139         android:exported="true" />
140
141 </application>
</manifest>

```

2. API总体机制说明

2.1 SDK合规说明

SDK合规规则说明

- 1、初始化时，游戏传入是否已申请权限的标志，SDK程序按需申请所需的权限
 - 避免游戏申请后，再申请，导致重复申请而违规
 - 依赖的权限已经在《九游隐私政策》申请
 - 技术对接，参考SDK接入文档的init接口说明
- 2、如果权限用户没有授权，SDK程序不会调用对应的API，避免违规
- 3、针对用户已经授权的权限，SDK程序通过缓存策略降低对应API的调用，有效控制读取的频率

详情请看：

<https://usdpdown.game.uodoo.com/cpssgame/resources/publicfiles/template/九游游戏SDK合规指南.pdf>

2.2 消息通知机制

SDK的大部分API调用使用了事件通知的方式回调给游戏,游戏通过创建一个SDKEventReceiver对象实现SDK的每个事件类型并通过UCGameSDK.defaultSdk().registerSDKEventReceiver(eventReceiver)接口注册后,就可以获取接口调用的结果。

创建SDKEventReceiver的方法大致如下:

▼ Java

```
1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {
2     @Subscribe(event = SDKEventKey.ON_INIT_SUCC)
3     private void onInitSucc() {
4         appendText("初始化成功");
5     }
6 };
```

2.1.1 注册与反注册

在游戏Activity onCreate中调用

UCGameSDK.defaultSdk().registerSDKEventReceiver(eventReceiver)注册

在游戏Activity onDestroy 中调用

UCGameSDK.defaultSdk().unRegisterSDKEventReceiver(eventReceiver)反注册

注意：如果回调方法涉及UI操作，请务必保证在UI线程执行；如果回调方法涉及耗时操作，请务必保证在异步线程执行。

2.1.2 关键字段说明如下

字段	说明
SDKEventReceiver	所有接收SDK事件的对象必须从SDKEventReceiver派生，或者直接如上例创建SDKEventReceiver的对象，对象创建出来后调用registerSDKEventReceiver方法注册后，即完成了事件接收的准备。
Subscribe	注册SDK事件的注解，event表示事件的类型，可选值请参考SDKEventKey类。
onInitSucc	此例中用于处理“初始化成功”事件的方法定义，方法名可以随意取，方法参数规格需参照各SDK功能接口的定义说明。

2.1.3 SDKEventKey事件如下

方法	说明
ON_INIT_SUCC	初始化成功
ON_INIT_FAILED	初始化失败
ON_LOGIN_SUCC	登录成功
ON_LOGIN_FAILED	登录失败
ON_CREATE_ORDER_SUCC	订单创建成功
ON_PAY_USER_EXIT	支付界面退出
ON_LOGOUT_SUCC	注销成功
ON_LOGOUT_FAILED	注销失败
ON_EXIT_SUCC	退出成功
ON_EXIT_CANCELED	取消退出，选择继续游戏
ON_ACCOUNT_SWITCH_REQUEST	切换账号请求

3. API使用说明

所有的api都需要通过获取UCGameSdk的实例对象来调用，UCGameSdk对象可以通过以下方法获取：

▼Java

```
1  UCGameSdk sdk = UCGameSdk.defaultSdk();
```

此为一个单例对象，可以在任意地方获取调用。

3.1 初始化SDK(必须接入)

首先，您需要在程序开始的地方初始化UCGameSdk对象，通过调用initSdk方法初始化九游游戏SDK。在初始化失败的状态下，游戏 不应 继续调用SDK的其余API，此方法必须在UI线程中调用。游戏启动后，非初始化失败或异常情况下， 不允许多次调用初始化接口 。

3.1.1 方法定义

▼ Java

```
1 void initSdk(final Activity activity, final SDKParams params)
2     throws AliLackActivityException
```

3.1.2 参数说明

输入参数说明:

参数	说明
activity	游戏的Activity对象，不可为null。
params	不可传null。

异常说明:

异常名称	说明
AliLackActivityException	未传入Activity时抛出此异常。

SDKParams参数说明（定义在SDKParamKey）：

参数定义	参数名称	类型	是否必填	说明
GAME_PARAMS	gameParams	ParamInfo	是	1. setGameId将游戏gameId传入该参数从开放平台-游戏管理-SDK接入-获取参数那里拿 2、 setOrientation设置屏幕方向，PORTRAIT竖屏、LANDSCAPE横屏，不设置默认为竖屏，横屏游戏必须设置为LANDSCAPE

GAME_HAD_REQUEST_PERMISSION	params	boolean	否	true, 表示游戏已经申请权限, sdk内部将不再申请权限 默认false
-----------------------------	--------	---------	---	---

3.1.3 代码示例

```

1  ParamInfo gpi = genGameParams();//请自行生成对象
2  gpi.setGameId(119474); // 从九游开放平台“SDK接入”获取自己游戏的参数信息
3
4  //设置SDK屏幕方向
5  //LANDSCAPE: 横屏, 横屏游戏必须设置为横屏, PORTRAIT: 竖屏
6  gpi.setOrientation(UCOrientation.PORTRAIT);
7
8  SDKParams sdkParams = new SDKParams();
9  sdkParams.put(SDKParamKey.GAME_PARAMS, gpi);
10 //如果游戏已经申请了权限, 不想sdk主动请求权限, 要通过参数告知sdk
11 sdkParams.put(SDKParamKey.GAME_HAD_REQUEST_PERMISSION, true);
12 UCGameSdk.defaultSdk().initSdk(this.getActivity(), sdkParams);

```

init方法没有返回值, 初始化结果通过事件通知回调给游戏:

```

1  private SDKEventReceiver eventReceiver = new SDKEventReceiver() {
2
3      @Subscribe(event = SDKEventKey.ON_INIT_SUCC)
4      private void onInitSucc() {
5          appendText("初始化成功");
6      }
7
8      @Subscribe(event = SDKEventKey.ON_INIT_FAILED)
9      private void onInitFailed(String desc) {
10         appendText("初始化失败:" + desc);
11     }
12 };

```

3.2 登录(必须接入)

3.2.1 方法定义

```

1 void login(Activity activity, SDKParams params)
2     throws AliLackAcitvityException, AliNotInitException

```

3.2.2 参数说明

输入参数说明:

参数	说明
activity	游戏的Activity对象，不可为null。
params	可传null。

异常说明:

异常名称	说明
AliNotInitException	未初始化或正在初始化时调用此接口将抛出此异常。
AliLackAcitvityException	Activity对象为null时将抛出此异常。

3.2.3 代码示例

账号登录

```

1 try {
2     UCGameSdk.defaultSdk().login(this.getActivity(), sdkParams);
3 } catch (AliNotInitException e) {
4     //未初始化或正在初始化时，异常处理
5 } catch (AliLackAcitvityException e) {
6     //activity为空，异常处理
7 }

```

3.2.4 login 回调结果

像初始化接口那样在事件接收对象处理回调结果

```

1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {
2
3     @Subscribe(event = SDKEventKey.ON_LOGIN_SUCC)
4     private void onLoginSucc(String sid) {
5         //sid即token，需发送给游戏服务器做登录校验获取accountId用户唯一标识，客户端
        无法获取用户唯一标识
6         appendText("登录成功,sid:" + sid);
7     }
8
9     @Subscribe(event = SDKEventKey.ON_LOGIN_FAILED)
10    private void onLoginFailed(String desc) {
11        appendText("登录失败:" + desc);
12    }
13
14 };

```

当登录成功时SDKEventKey.ON_LOGIN_SUCC事件被触发，同时通过带回登录sid给游戏，sid为动态token，每次登录都会改变，需发送给游戏服务器做登录校验获取accountId用户唯一标识，客户端无法获取用户唯一标识，参考服务端接口协议1.3.1章节——用户会话验证接口

3.3 充值(如开计费必须接入，必须先完成合同签约才能调，否则支付调起报03错误码)

3.3.1 充值操作

3.3.2 方法定义

```

1 void pay(Activity activity, SDKParams params)
2     throws AliLackAcitivityException, AliNotInitException, IllegalArgumentException
    xception

```

3.3.3 参数说明

输入参数说明：

参数	说明
----	----

activity	游戏的Activity对象，不可为null。
params	不可传null。

异常说明:

异常名称	说明
AliNotInitException	未初始化或正在初始化时调用此接口将抛出此异常。
IllegalArgumentException	SDKParams为空时调用此接口将抛出异常。
AliLackAcitivityException	传入的activity为空时抛出此异常。

SDKParams参数说明（定义在SDKParamKey）：

参数定义	参数名称	类型	是否必填	说明
CALLBACK_INFO	callbackInfo	String	否	cp自定义信息，在支付结果通知时回传,CP可以自己定义格式,长度不超过250
AMOUNT	amount	String	是	充值金额（字符串类型），最多保留小数点后2位，单位为元。例：10.00，整数金额可不传小数点部分，建议金额由服务端格式化好，客户端直接使用。 注：7.6.0.0插件版本起，不支持传入金额为0。
NOTIFY_URL	notifyUrl	String	否	服务器通知地址，如果为空则以开放平台配置地址作为通知地址，优先取客户端地址(建议通过游戏服务端配置参数后读取)，长度不超过100
CP_ORDER_ID	cpOrderId	String	否	cp充值订单号，需要保证用户每次充值订单号的唯一性，长度不超过30
ACCOUNT_ID	accountId	String	是	服务端调用登录会话验证接口verifySession返回的accountId，即用户唯一标识
SIGN_TYPE	signType	String	是	名类型，MD5或者RSA， 该参数不需要参与签名 目前只支持MD5

SIGN	sign	String	是	签名结果，该参数不需要参与签名MD5 (签名内容+apiKey)；服务端生成签名算法参考下面说明
------	------	--------	---	--

注：sign必须由服务器生成，另以上所有参数值请从服务器获取，不能直接在客户端生成，确保客户端与服务器数据一致，并且apiKey不允许写在客户端

签名算法：

1. 将支付请求信息中所有参数名称key按照字典顺序排列(a-z)，值为空串需参与签名，值为null的参数不需要参与签名，sign和signType也不参数签名。（注意签名的key为上表内第二列“参数名称”,非第一列的“参数定义”）

所有参数名称按照参数名的字典排序连接起来组成待签名数据，格式：p1=v1p2=v2p3=v3...，签名内容不应包含“&”符号，拼接签名内容时需把“&”符号剔除

3.根据signType的设定，进行相应MD5或RSA签名；

对于MD5签名，将上述方式生成的待签名数据，拼上密钥后进行MD5运算。

对于RSA签名，则是将上述方式生成的待签名数据和密钥，直接进行RSA签名；

MD5签名结果忽略大小写；

假如apiKey为123456，待签名原始串示例如下（签名内容+apiKey）：

Java

```

1  accountId=123452132amount=100.00callbackInfo=xxxxxcpOrderId=XXXXXXnotifyUrl
   =[[http://192.168.1.1/notifypage.do123456]{.underline}](http://192.168.1.1/
   notifypage.doroleId=102123456)

```

签名算法示例代码：


```
1  /**
2   * 签名工具方法
3   * @param reqMap
4   * @return
5   */
6  public static String sign(Map<String, String> reqMap, String signKey) {
7      //将所有key按照字典顺序排序
8      TreeMap<String, String> signMap = new TreeMap<String, String>(reqMap);
9      StringBuilder stringBuilder = new StringBuilder(1024);
10     for (Map.Entry<String, String> entry : signMap.entrySet()) {
11         // sign和signType不参与签名
12         if ("sign".equals(entry.getKey()) || "signType".equals(entry.getKey()) || "signType".equals(entry.getKey())){
13             continue;
14         }
15         //值为null的参数不参与签名
16         if (entry.getValue() != null) {
17             stringBuilder.append(entry.getKey()).append("=").append(entry.getValue());
18         }
19     }
20     //拼接签名密钥
21     stringBuilder.append(signKey);
22     //剔除参数中含有的'&'符号
23     String signSrc = stringBuilder.toString().replaceAll("&", "");
24     return md5(signSrc).toLowerCase();
25 }
```

3.3.4 代码示例

```
1  SDKParams params = new SDKParams();
2  sdkParams.put(SDKParamKey.CALLBACK_INFO, "{\"test\":true}");
3  sdkParams.put(SDKParamKey.NOTIFY_URL, "http://192.168.1.1/notifypage.do");
4  sdkParams.put(SDKParamKey.AMOUNT, "2.33");
5  sdkParams.put(SDKParamKey.CP_ORDER_ID, "20160000101001");
6  sdkParams.put(SDKParamKey.ACCOUNT_ID, "123");
7  sdkParams.put(SDKParamKey.SIGN_TYPE, "MD5");
8  sdkParams.put(SDKParamKey.SIGN, "游戏服务器根据以上信息签名后的结果");
9  // 以上字段的值都需要由游戏服务器生成,各字段详细说明, 见SDKParamKey参数表
10
11 try {
12     UCGameSdk.defaultSdk().pay(this.getActivity(), sdkParams);
13 } catch (IllegalArgumentException e) {
14     //传入参数错误异常处理
15 } catch (AliNotInitException e) {
16     //未初始化或正在初始化时, 异常处理
17 } catch (AliLackActivityException e) {
18     //activity为空, 异常处理
19 }
```

3.3.5 支付回调

```

1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {
2
3     @Subscribe(event = SDKEventKey.ON_CREATE_ORDER_SUCC)
4     private void onCreateOrderSucc(OrderInfo orderInfo) {
5         if (orderInfo != null) {
6             StringBuilder sb = new StringBuilder();
7             sb.append(String.format("'orderId': '%s'", orderInfo.getOrderId
            ());
8             sb.append(String.format("'orderAmount': '%s'", orderInfo.getOrd
            erAmount()));
9             sb.append(String.format("'payWay': '%s'", orderInfo.getPayWay()
            ));
10            sb.append(String.format("'payWayName': '%s'", orderInfo.getPayW
            ayName()));
11
12            Log.i(CLASS_NAME, "此处为订单生成回调，客户端无支付成功回调，订单是否
            成功以服务端回调为准: callback orderInfo = " + sb);
13        }
14    }
15
16    @Subscribe(event = SDKEventKey.ON_PAY_USER_EXIT)
17    private void onPayUserExit(OrderInfo orderInfo) {
18        if (orderInfo != null) {
19            StringBuilder sb = new StringBuilder();
20            sb.append(String.format("'orderId': '%s'", orderInfo.getOrderId
                ());
21            sb.append(String.format("'orderAmount': '%s'", orderInfo.getOrd
                erAmount()));
22            sb.append(String.format("'payWay': '%s'", orderInfo.getPayWay()
                ));
23            sb.append(String.format("'payWayName': '%s'", orderInfo.getPayW
                ayName()));
24
25            Log.i(CLASS_NAME, "支付界面关闭: callback orderInfo = " + sb);
26        }
27    }
28 }
29 };

```

充值界面错误码说明（必看）

错误码	原因	说明
-----	----	----

01	请求参数为空	解析数据时请求信息为空或不完整，要求必填的字段不能为空
02	请求数据无法解密	解密失败
03	游戏不存在或未配置充值信息	原因可能是游戏未签约，或签约审核未通过，请等待签约通过后过一小时再测试
04	商品不存在或已下线	商品已下线，请确认是否游戏已下架停运
05	游戏未开通任何支付方式	游戏未开通任何支付方式，请联系技术接口人处理
06	用户登录信息已失效	无登录状态，请检查是否调用logout注销登录
07	cp支付信息为空	
08	cp支付信息签名为空	sign字段为空，该字段必传，不可为空
09	cp支付信息签名验证不通过	sign错误 1. 对照上面的签名算法核查签名算法是否有错 2. 检测传递的apikey与初始化的gameId是否为开放平台分配的游戏参数
10	cp支付信息统一账号验证不通过	用户登录成功后服务器校验返回的账号标识accountId与支付传递的accountId不一致
11	使用错误的serverId	传递的serverId错误,要求使用0
12	回调地址未配置	客户端传递和服务端信息均无通知地址
13	渠道支付功能被关闭	渠道因违规暂停支付功能
14	amount金额格式不正确	amount支付金额不是数字格式
15	callbackInfo回调信息格式不正确	callbackInfo长度超过250
16	CP订单号格式不正确	cpOrderId长度超过30
17	通知地址格式不正确	notifyUrl长度超过100

98	请求参数不合法	解析授权参数时失败
99	系统异常	创建会话失败、获取商品支付方式信息失败

3.3.2.1 生产环境联调

在生产环境联调时，需使用真实有效的充值卡进行测试，可购买小面额的充值卡用于测试。

3.3.2.2 关于充值提交

使用充值卡充值涉及到多方的支付平台，充值结果不能实时获得，有时会有相当时间的延迟才能收到充值结果。因而整个充值是一个两段的过程：①充值提交提交，状态有“成功”和“失败”，成功则生成充值订单，并将订单号返回给游戏；②通过第三方支付平台进行实际支付，此时也有“成功”和“失败”状态，“成功”意味着充值卡扣费完成。

充值订单提交的“成功”并不意味着充值卡扣费的最终完成，有许多因素会造成充值卡扣费的失败，如：卡已使用过、同一充值卡使用过于频繁、支付联网故障等。充值订单提交后，游戏服务器通过“充值结果回调接口”收到充值结果通知的“成功”消息后，才标志着充值的真正完成。

如果充值订单提交后未能及时收到充值结果通知，应耐心等待，也可以通过客服人员追查充值情况。此时用户如果反复使用同一张卡提交充值订单，反而会造成充值的失败。

3.3.2.3 关于定额充值

SDK支持游戏指定是否限制用户只能按照指定金额进行充值，做法是：在pay方法的amount参数中指定金额。如果指定了金额，用户在充值时则不能自由选择或输入金额，只能充入指定的金额；设置定额充值的游戏服务端收到回调信息必须校验amount值与客户端下单时传递的是否一致

3.4 提交游戏角色数据信息(可选接入，客户端或服务端任选一接入，建议服务端接入)

游戏SDK要求游戏在运行过程中提交一些用于运营需要的扩展数据，这些数据通过扩展数据提交方法进行提交，要求走服务端接入，因客户端数据存在被篡改风险，若贵司依然走客户端接入，游戏内营销活动被刷造成的损失由游戏贵司承担。

数据提交时机：

用户创建角色或已有角色进入游戏后
当用户的角色等级发生变化后

3.4.1.1 方法定义

▼

Java

```
1 void submitRoleData(Activity activity, SDKParams params)
2 throws AliLackAcitivityException, AliNotInitException, IllegalArgumentException
   tion
```

3.4.1.2 输入参数说明

参数	说明
activity	游戏activity，不能为空。
params	玩家信息参数，不能为空。

异常说明：

异常名称	说明
AliNotInitException	未初始化或正在初始化时调用此接口将抛出此异常。
IllegalArgumentException	SDKParams为空时调用此接口将抛出异常。
AliLackAcitivityException	activity为空时调用此接口将抛出异常。

游戏数据参数（定义在SDKParamKey）：

参数定义	参数名称	类型	是否必填	说明
STRING_ROLE_ID	roleId	String	是	角色ID,长度不超过50
STRING_ROLE_NAME	roleName	String	是	角色名称,长度不超过50

LONG_ROLE_LEVEL	roleLevel	long	是	角色等级,长度不超过10
LONG_ROLE_CTIME	roleCTime	long	是	角色创建时间(单位: 秒), 长度10, 获取服务器存储的时间, 不可用手机本地时间
STRING_ZONE_ID	zoneId	String	是	区服ID,长度不超过50
STRING_ZONE_NAME	zoneName	String	是	区服名称,长度不超过50

3.4.1.3 代码示例

Java

```

1  long roleLevelM = 12;
2  long roleCTimeM = 1456397360;
3
4  SDKParams params = new SDKParams();
5  params.put(SDKParamKey.STRING_ROLE_ID, "xx_sxx");
6  params.put(SDKParamKey.STRING_ROLE_NAME, "法师");
7  params.put(SDKParamKey.LONG_ROLE_LEVEL, roleLevelM);
8  params.put(SDKParamKey.LONG_ROLE_CTIME, roleCTimeM);
9  params.put(SDKParamKey.STRING_ZONE_ID, "xxxx");
10 params.put(SDKParamKey.STRING_ZONE_NAME, "区服1");
11 try {
12     UCGameSdk.defaultSdk().submitRoleData(this.getActivity(), sdkParams);
13 } catch (IllegalArgumentException e) {
14     //传入参数错误异常处理
15 } catch (AliNotInitException e) {
16     //未初始化或正在初始化时, 异常处理
17 } catch (AliLackActivityException e) {
18     //activity为空, 异常处理
19 }

```

注：此方法没有回调，调用后如接口有报异常，可能为参数类型错误

3.4.1.4 自检工具

开放平台-SDK接入-调试工具（数据接口），添加登录游戏对应的九游账号后过一分钟，在游戏内提交数据可在该界面看到数据展示，如无显示数据，可能是添加的账号不对应，或数据接口调用有异常或接口未执行，请通过运行日志排查

3.5 退出账号

九游游戏SDK提供了logout接口用于退出当前登录的账号，并将结果回调给游戏。

对于需要支持账号切换（退出当前登录账号）的游戏，应在退出账号回调成功后再次调起登录接口。

3.5.1 方法定义

▼ Java

```
1 void logout(Activity activity, SDKParams params)
2     throws AliLackAcitivityException, AliNotInitException
```

输入参数说明：

参数	说明
activity	游戏Activity对象，不可为空。
params	传null

异常说明：

异常名称	说明
AliNotInitException	未初始化或正在初始化时调用此接口将抛出此异常。
AliLackAcitivityException	Activity对象为null时将抛出此异常。

3.5.2 代码示例


```

1 try {
2     UCGameSdk.defaultSdk().logout(this.getActivity(), null);
3 } catch (AliNotInitException e) {
4     //未初始化或正在初始化时，异常处理
5 } catch (AliLackActivityException e) {
6     //activity为空，异常处理
7 }

```

3.5.3 logout回调

```

1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {
2
3     @Subscribe(event = SDKEventKey.ON_LOGOUT_SUCC)
4     private void onLogoutSucc() {
5         appendText("退出账号成功");
6         // 切换账号时，可以再次调起登录接口
7     }
8
9
10    @Subscribe(event = SDKEventKey.ON_LOGOUT_FAILED)
11    private void onLogoutFailed() {
12        appendText("退出账号失败");
13    }
14 };

```

3.6 退出SDK(必须接入)

当游戏退出前 **必须** 调用该方法，进行清理工作。如果游戏直接退出，而不调用该方法，可能会出现未知错误，导致程序崩溃。

3.6.1 调用说明（重要）

该接口需要游戏在退出前调用，游戏等待该接口的回调通知后，再执行游戏的退出。

此接口需要在UI线程中调用。

3.6.2 接口定义

▼

Java

```
1 public void exit(Activity activity, SDKParams params)
2     throws AliLackAcitivityException, AliNotInitException
```

3.6.3 参数说明

输入参数说明：

参数	说明
activity	游戏Activity对象
params	传null。

异常说明：

异常名称	说明
AliNotInitException	未初始化或正在初始化时调用此接口将抛出此异常。
AliLackAcitivityException	Activity对象为null时将抛出此异常。

3.6.4 代码示例

▼

Java

```
1 try {
2     UCGameSdk.defaultSdk().exit(this.getActivity(), null);
3 } catch (AliNotInitException e) {
4     //未初始化或正在初始化时，异常处理
5 } catch (AliLackAcitivityException e) {
6     //activity为空，异常处理
7 }
```

3.6.5 exit回调

```
1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {  
2     @Subscribe(event = SDKEventKey.ON_EXIT_SUCC)  
3     private void onExitSucc() {  
4         appendText("SDK退出");  
5     }  
6  
7     @Subscribe(event = SDKEventKey.ON_EXIT_CANCELED)  
8     private void onExitCanceled() {  
9         appendText("放弃退出, 继续游戏");  
10    }  
11 };
```

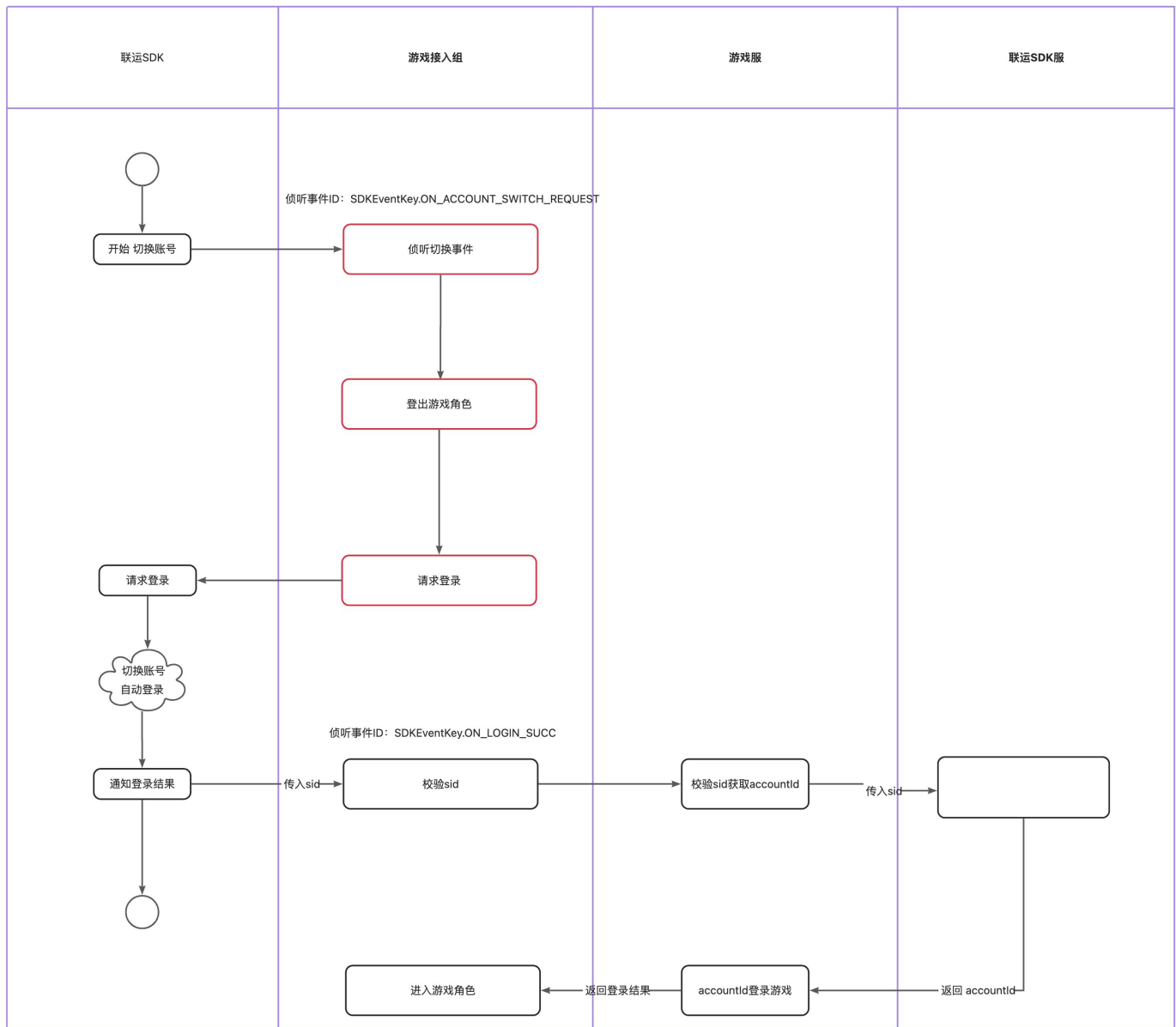
3.7 切换账号接口（选接）

SDK提供的被动切换账号的接口，需要游戏响应账号切换动作，最终协同完成用户账号和游戏角色的切换；更详细的信息可以查看：[《联运SDK切换账号接入指南-简化方案》](#)。

3.7.1 调用说明（重要）

该接口需要游戏响应切换指令，游戏需要退出当前游戏角色，并重新调用登录请求。

具体流程图：



3.7.2 切换账号的回调

代码示例

```
1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {  
2  
3     .....省略若干代码.....  
4  
5     @Subscribe(event = SDKEventKey.ON_ACCOUNT_SWITCH_REQUEST)  
6     private void onAccountSwitchRequest(final String sid) {  
7         // 注意: sid是兼容旧版本保留下来的, 9.5.10.2版本及以上请不要使用sid!  
8     }  
9  
10    .....省略若干代码.....  
11  
12 };  
13
```

当用户从游戏启动器启动游戏，同时要求切换账号时，

SDKEventKey.ON_ACCOUNT_SWITCH_REQUEST事件被触发；【注意：sid是兼容旧版本保留下来的，9.5.10.2版本及以上请不要使用sid！】

场景示例

在声明SDK回调的地方，进行切换账号的流程处理：

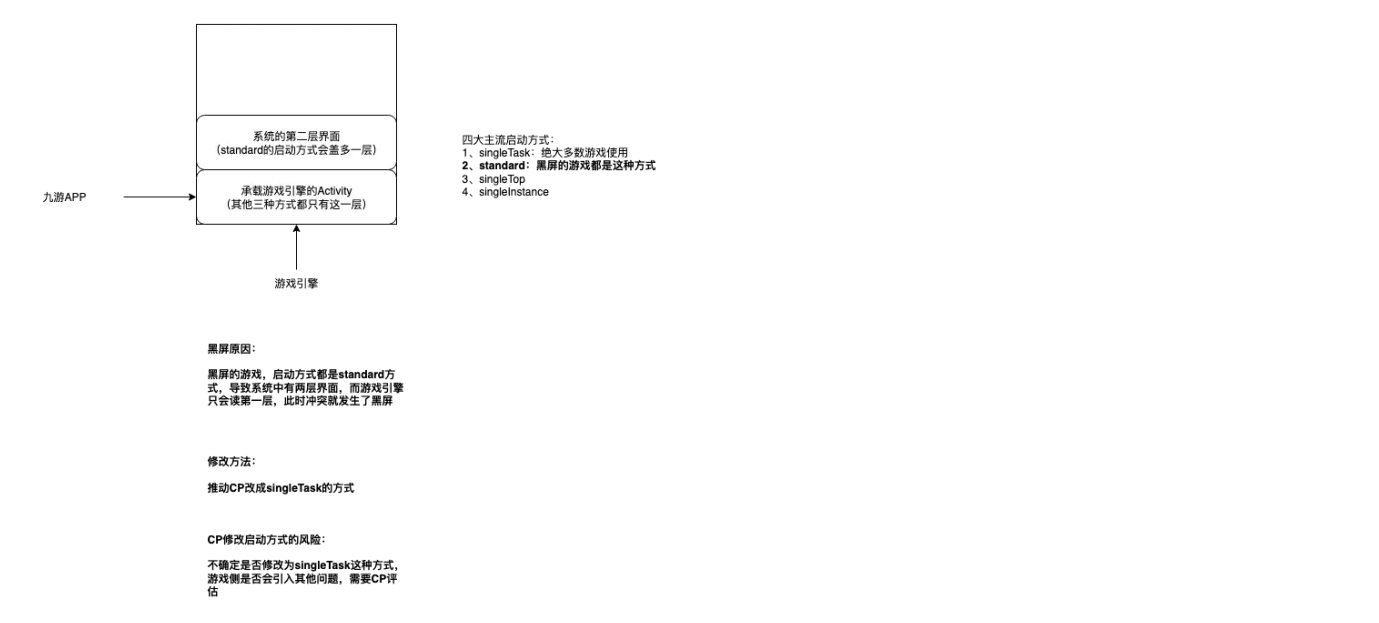
```

1 private SDKEventReceiver eventReceiver = new SDKEventReceiver() {
2
3     .....省略若干代码.....
4
5     // 步骤1, 侦听SDK切换账号指令
6     @Subscribe(event = SDKEventKey.ON_ACCOUNT_SWITCH_REQUEST)
7     private void onAccountSwitchRequest(final String sid) {
8         //注意: sid是兼容旧版本保留下来的, 9.5.10.2版本及以上请不要使用sid!
9         //步骤2, CP接入组先退出当前游戏角色
10        //这里需要注意:
11        //    1. 【重要】必须退出当前游戏角色, 否则会因为“切换场景未登出游戏角色”被打
回
12        logoutGameRole();
13        //步骤3, 和正常登录一样, 调用sdk做登录请求
14        //这里需要注意:
15        //    1. 如果游戏的退出和重新登录在native层是异步的,
16        //        那requestLogin()应该由cp自行决定在合适的节点调用,
17        //        不要求必须放在onAccountSwitchRequest()方法中;
18        //    2. 【重要】如果游戏首页本身会自动触发登录,
19        //        那onAccountSwitchRequest()中不需要额外调用requestLogin();
20        //        否则会因为“切换账号场景重复登录”被打回;
21        //    3. requestLogin()需要保证在角色已退出的情况下调用;
22        requestLogin();
23    }
24
25    .....省略若干代码.....
26
27 };
28
29 private void logoutGameRole() {
30     CP接入组需要实现游戏角色的登出;
31 }
32
33 private void requestLogin() {
34     try {
35         UCGameSdk.defaultSdk().login(this.getActivity(), null);
36     } catch (AliNotInitException e) {
37         //未初始化或正在初始化时, 异常处理
38     } catch (AliLackActivityException e) {
39         //activity为空, 异常处理
40     }
41 }

```

3.7.3 切换账号后游戏黑屏问题（重要）

为了避免游戏在运营期，在用户发起切换账号后遇到游戏黑屏等异常，强烈建议接入时，检查游戏的Manifest文件，游戏主Activity的launchMode配置是否为singleTask，其原理如下：



3.8 业务接口(可选接入)

提供通用调用业务接口能力，通过定义好的业务类型及业务参数传达指令，执行具体业务操作并把结果通知给接入方。

3.8.1 调用说明（重要）

该接口需要初始化成功后调用。

此接口需要在UI线程中调用。

3.8.2 接口定义

Java

```
1 public void execute(Activity activity, String callbackId, String action, SDKParams params) throws AliLackAcitivityException, AliNotInitException
```

3.8.3 参数说明

输入参数说明：

参数	说明
activity	游戏Activity对象
params	业务指令执行的参数,根据不同业务指令去配置

异常说明：

异常名称	说明
AliNotInitException	未初始化或正在初始化时调用此接口将抛出此异常。
AliLackAcitivityException	Activity对象为null时将抛出此异常。

3.8.4 代码示例

Java

```

1  SDKParams sdkParams = new SDKParams();
2  sdkParams.put(SDKParamKey.ACTION, "showPage");
3  sdkParams.put(SDKParamKey.BUSINESS, "bindPhone");
4  sdkParams.put(SDKParamKey.ORIENTATION, 1);
5
6  try {
7      UCGameSdk.defaultSdk().execute(getActivity(), sdkParams);
8  } catch (AliLackAcitivityException e) {
9      e.printStackTrace();
10     printMsg("需要activity");
11 } catch (AliNotInitException e) {
12     e.printStackTrace();
13     printMsg("需要初始化");
14 } catch (Exception e) {
15     e.printStackTrace();
16     printMsg("Exception: " + e.getMessage());
17 }

```

添加SDK退出回调：


```

1 private SDKEventReceiver executeReceiver = new SDKEventReceiver() {
2
3     @Subscribe(event = SDKEventKey.ON_EXECUTE_SUCC)
4     private void onExecuteSucc(String msg) {
5         //执行指令成功
6         printMsg("onExecuteSucc msg=" + msg);
7     }
8
9     @Subscribe(event = SDKEventKey.ON_EXECUTE_FAILED)
10    private void onExecuteFailed(String msg) {
11        //执行指令失败
12        printMsg("onExecuteFailed msg=" + msg);
13    }
14
15 }

```

3.8.5 ShowPage指令

3.8.5.1 参数说明

action业务类型说明：

参数	说明
showPage	打开sdk指定的页面

SDKParams参数说明

参数定义	参数名称	类型	必填	说明	默认值
ACTION	action	string	是	业务操作指令	/
BUSINESS	business	string	是	页面名	/
ORIENTATION	orientation	int	否	1 强制竖屏 2 强制横屏 0 不变，用户原来手机默认状态	0

showPage的回调处理事件

回调	说明	值
ON_SHOW_PAGE_RESULT	页面里面发送的消息结果	110001

business页面类型说明

页面名	说明	页面类型	回调result	回调字段说明	调用时机
bindPhone	绑定手机页	webview	{"code":1,"msg":"绑定成功"}	code (1: 成功 0: 失败) msg (结果返回描述)	登陆后

3.8.5.2 代码示例

注册回调事件

Java

```
1 private SDKEventReceiver showPageReceiver = new SDKEventReceiver() {
2     @Subscribe(event = SDKActionEventKey.ShowPageAction.ON_SHOW_PAGE_RESULT
3 )
4     private void onShowPageResult(String business, String result) {
5         //showPage页面发送消息结果
6         printMsg("onShowPageResult business=" + business + " result=" + result);
7     }
8 };
```

SDKActionEventKey需自行定义，可参考：

Java

```
1 public class SDKActionEventKey {
2     public static class ShowPageAction {
3         public static final int ON_SHOW_PAGE_RESULT = 110001;
4     }
5 }
```

3.9 可选个人信息配置(可选接入)

3.9.1 调用说明（重要）

该接口需要初始化调用之前调用。

此接口需要在UI线程中调用。

3.9.2 接口定义

▼ Java

```
1 public void setPrivacyManager(SdkPrivacyManager privacyManager) throws AliNotInitException
```

3.9.3 参数说明

输入参数说明：

参数	说明
privacyManager	个人信息配置管理器

异常说明：

异常名称	说明
AliNotInitException	初始化成功后调用此接口将抛出此异常。

3.9.4 代码示例

```
1 // 必须在初始化调用之前设置
2 try {
3     // 设置SdkPrivacyManager后，SDK内降不再采集可选的个人信息，
4     // 直接采用接入方注入的SdkPrivacyManager相关方法提供的返回值
5     UCGameSdk.defaultSdk().setPrivacyManager(new SdkPrivacyManager() {
6         @Override
7         public boolean hadShowPrivacyDlg() {
8             // 返回游戏是否已经展示过隐私弹窗
9             return super.hadShowPrivacyDlg();
10        }
11
12        @Override
13        public String getImei() {
14            // 返回游戏采集到的Imei，不采集请直接传空字符串
15            return "";
16        }
17
18        @Override
19        public String getImsi() {
20            // 返回游戏采集到的Imsi，不采集请直接传空字符串
21            return "";
22        }
23
24        @Override
25        public String getAndroidId() {
26            // 返回游戏采集到的AndroidId，不采集请直接传空字符串
27            return "";
28        }
29    });
30 } catch (AliNotInitException e) {
31     e.printStackTrace();
32 }
33 // 初始化
34 UCGameSdk.defaultSdk().initSdk(this.getActivity(), sdkParams);
```

4. 接入规范

“游戏客户端”集成“游戏SDK”开发时，须遵循以下要求：

4.1. 游戏包名称命名规范

游戏客户端的AndroidManifest.xml中游戏包名定义加上“.aligames”后缀。如原游戏包名为 package="com.lori.app", 则修改为 package="com.lori.app .aligames ", 已在九游上线过的游戏, 保持原包名不改

4.2. icon图片

游戏图标需增加九游角标, 角标位置在接入包内“角标logo及使用说明”文件夹下, 添加角标后在AndroidManifest.xml中指定。如android:icon="@drawable/icon"。其中icon为在原游戏图标基础上制作的能表明九游渠道的图标文件。

以上两点规范都表现在AndroidManifest.xml中, 以下为一个AndroidManifest.xml的例子:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3  package="cn.lori.fighter.aligames" android:versionCode="1" android:versi
   onName="1.0" >
4      .....
5      <application android:icon="@drawable/ic_launcher_uc"
6                  android:label="@string/app_name" >
7          .....
8          .....
9      </application>
10 </manifest>
```

4.3. 代码混淆要求

如果游戏发布时采用proguard进行代码混淆, 请在proguard配置文件加入以下代码, 以避免对SDK进行混淆, 否则会造成SDK部分功能不正常。

```
1  -printmapping 'mapping.txt'
2  # this is very dangerous that cause bug
3  -ignorewarnings
4
5  -dontskipnonpubliclibraryclassmembers
6  -dontshrink
7  -dontoptimize
8  -keepattributes Exceptions,InnerClasses,Signature,Deprecated,SourceFile,Line
   NumberTable,LocalVariable*Table,*Annotation*,Synthetic,EnclosingMethod
9
10 -keepclasseswithmembers class * extends cn.gundam.sdk.shell.event.SDKEventR
    eceiver
11
12 ▾ -keep class cn.uc.**{
13     <methods>;
14     <fields>;
15 }
16
17 ▾ -keep class cn.gundam.**{
18     <methods>;
19     <fields>;
20 }
21
22 ▾ -keep class android.**{
23     <methods>;
24     <fields>;
25 }
26
27 ▾ -keep class org.json.**{
28     <methods>;
29     <fields>;
30 }
```

5. 游戏客户端开发要点（重要，必读）

“游戏客户端”集成“SDK客户端”的开发工作要点如下：

实现“初始化SDK”功能：（必须接入）

- a. 实现初始化回调处理；
- b. 调用初始化SDK方法；

实现“登录”功能：（必须接入）

- a. 实现登录回调处理
- b. 在回调中获取sid，传给游戏服务器端验证；

提交游戏角色数据（可选接入）

- a. 创建角色/已有角色进入游戏后提交游戏角色数据接口；
- b. 游戏内升级时调用提交游戏角色数据接口

实现“充值”功能（开计费必须接入）：

- a. 实现充值回调处理；
- b. 设置充值订单信息；
- c. 调用充值方法；

实现“退出SDK”功能（必须接入）

- a. 调用退出SDK方法；

5.1 【温馨提示1】 必须在UI线程中调用的接口列表如下：

初始化接口（initSDK）；

登录接口（login）

充值接口（pay）；

退出接口（exitSDK）

5.2 【温馨提示2】：

引用SDK后，发生编译错误时，可以试试执行一下IDE的“Clean”（或类似方法）以清除不一致的编译缓存文件。

附录一：接入注意事项及常见问题解决方案

请参考以下帖子：

- <https://aligames.open.uc.cn/document/doc/detail/323>

网游SDK接入常见问题：