Android篇

1．SDK

1.1 Unity编译打包 2

1.1.1 [unity5.0.X及以上版本接入](#unity50x及以上版本)2

1.1.2 [unity5.0.X及以上版本特殊情况接入](#unity50x某个版本以上方法不适用)2

1.1.3 [unity5.0.x之前版本接入](#unity50x之前版本不支持)3

1.2导出eclipse项目或android studio项目打包

1.2.1 [android studio项目接入](#androidstudio项目接入)3

1.2.2 [Eclipse项目接入](#Eclipse项目接入)3

2．API使用说明

2.1 unity 可调用接口

2.1.1[初始化](#初始化)4

2.1.2[打开悬浮框](#打开悬浮框)4

2.1.3[关闭悬浮框](#关闭悬浮框)5

2.1.4[切换“场景”](#切换场景)5

2.1.5[登录](#登录)5

2.1.6[退出登录](#退出登录)6

2.1.7[切换账号](#切换账号)6

2.1.8[上传游戏中角色信息](#上传游戏中角色信息)7

2.1.9[支付](#调用支付接口)9

2.1.10[屏类型](#返回屏类型)11

2.2 特殊接口

2.2.1[下载游戏参数.](#下载游戏参数)12

2.2.2[设备验证](#验证设备)12

2.3个别callback

2.3.1[悬浮框进度](#拖动悬浮框进度条)13

3．附录

3.1 [对接文件清单](#sdk文件清单)13

服务端

1．API

1.1 [支付](#支付成功对游戏商的通知接口)14

Android篇

1.sdk接入(**按打包编译方式分为以下两种**):

**1.1 Unity编译打包**

1.1.1 unity5.0.X及以上版本

1、在assets文件下创建文件夹Plugins/Android，将提供的wzsdk\_libRelease\_x.x.x.aar或者aar解压的库文件夹以及jar文件复制进去即可（注意：不要与我们的jar文件重复，请见[文件清单](#sdk文件清单)）

2、若存在游戏工程AndroidManifest，可以在里面添加以下游戏对接参数，不存在则可以直接在aar或者库文件里面的AndroidManifest修改相应的参数，（注意一般unity5中高版本以及android studio支持AndroidManifest自动合并，所以权限以及android组件不需要在游戏工程AndroidManifest中添加，如果不支持合并，请手动合并，需要添加的内容在可参考给出的AndroidManifest.xml，合并权限以及相应的service、activity等android组件信息）

<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="x.x.x.x">

<application>

<provider//用于自动安装更新apk 适配android 7.0不支持file://  
 android:name="android.support.v4.content.FileProvider"  
 android:authorities="xxx.xxx.xxx..fileProvider"//xxx换成包名  
 android:exported="false"  
 android:grantUriPermissions="true">  
 <meta-data  
 android:name="android.support.FILE\_PROVIDER\_PATHS"  
 android:resource="@xml/file\_paths" />  
 </provider>

<meta-data  
 android:name=" WZ\_APP\_ID "  
 android:value="xxxxxx"/>  
</application>

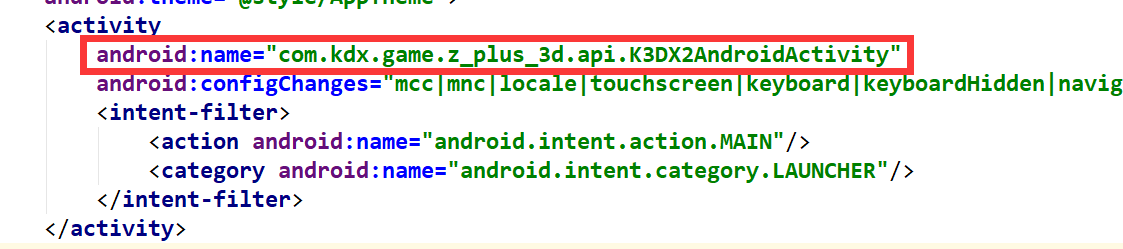
</manifest>

1.1.2 若unity5.0.x 某个版本以上方法不适用，可参照此方法

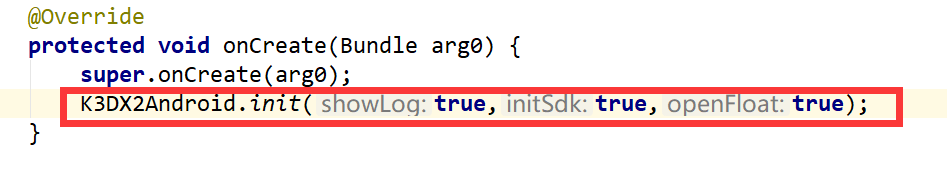
1、请参照[1.1.1中的1、2](#unity50x及以上版本)

2、根据主Activity的定义方式分为：

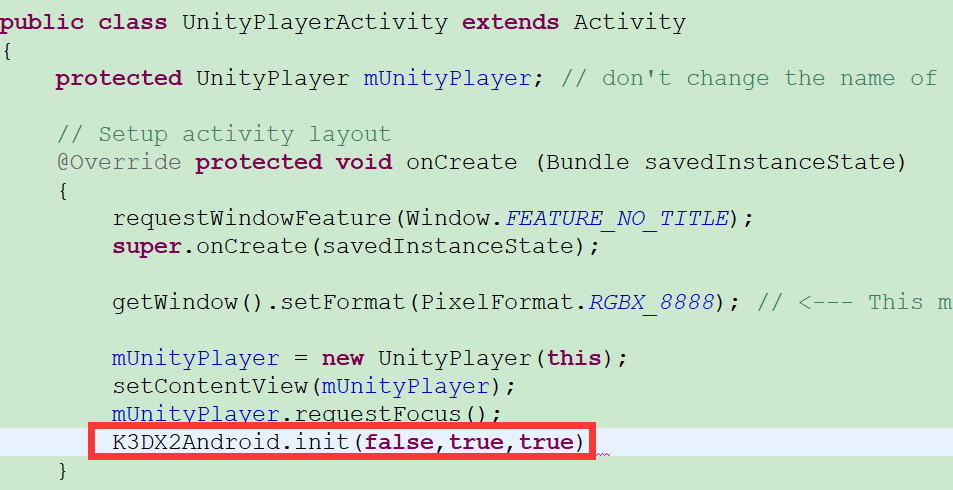
A、游戏工程主Activity为unity原生UnityPlayerActivity，非自定义Activity，可以将AndroidManifest中的MainActivity设为：com.kdx.game.z\_plus\_3d.api.K3DX2AndroidActivity，如图



B、游戏工程主Activity为自定义Activity，可引入sdk: wzsdk\_libRelease\_x.x.x中的class.jar，并在游戏主Activity 的onCreate() 中调用K3DX2Android.init(false,true,true)



注意：如果直接在UnityPlayerNativeActivity中修改，务必将此代码放最后：



1.1.3 unity5.0.x之前版本不支持，请按[1.2](#导出eclipse项目)接入

**1.2** **导出eclipse项目或android studio项目：**

1.2.1 android studio项目接入：

1)、将wzsdk\_libRelease\_x.x.x.aar拷贝到libs下

2)、配置build.gradle

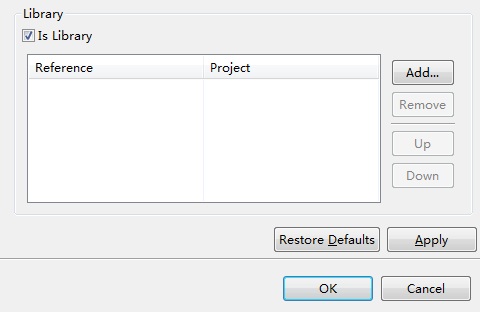
repositories { flatDir { dirs 'libs' } }  
dependencies {  
 compile(name: ' wzsdk\_libRelease\_x.x.x', ext: 'aar')

}

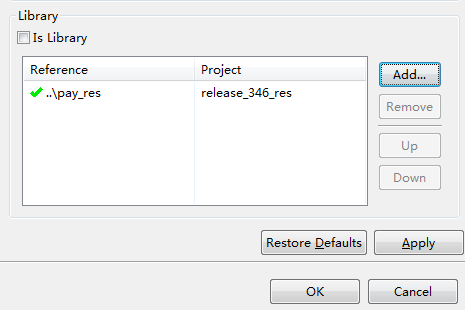
3)、参照**[1.1.1](#unity50x及以上版本)**配置AndroidManifest 如果运行时报权限或者没有悬浮框没出现等问题，可能是因为没有合并manifest，请参照给出的AndroidManifest.xml，合并权限以及相应的service、activity等android组件信息

1.2.2 Eclipse项目接入

1)、将sdk: wzsdk\_libRelease\_x.x.x 工程导入eclipse，并在Properties->Android中选中此工程为library工程。如图



2）、主工程关联library

在Properties->Android中将sdk工程设置为主工程项目的library，如图

3）、参照[**1.1.1**](#unity55X及以上版本)配置AndroidManifest，如果报权限或者没有悬浮框没出现等问题，那是因为没有合并manifest，请参照给出的AndroidManifest.xml，合并权限以及相应的service、activity等android组件信息

# 2、API接口

**2.1****以下为unity可调用接口**

2.1.1 初始化（由裸眼3D插件默认调用，无需对接）

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void Init（boolean showLog, boolean initSdk, boolean openFloat） |
| 入参说明 | 初始化参数，详见表2 |
| 同步回调 | Void |
| 异步回调 | 初始化的反馈，详见表3回调参数说明 |

表2.参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 默认值 | 说明 |
| showLog | boolean | 是 | False | 是否打印日志 |
| initSdk | boolean | 否 | false | 是否初始化SDK |
| openFloat | boolean | 否 | false | 是否打开悬浮框 |

表3回调参数

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 | | | |
| callbackType | String | 是 | 回调接口类型 | | | |
| code | Int | 是 | 通过网络请求获取游戏账号支持信息1、成功 0、失败 | | | |
| data | String |  | Data具体数据 | | | |
| 参数名称 | 类型 | 必须 | 说明 |
| enable\_user | Int | 是 | 是否支持账号系统：1支持0不支持 |
| enable\_sub | Int | 是 | 是否支持小账号：1支持，0不支持 |

示例：

{

"callbackType": "init",

"code": 1,

"data":{"enable\_user":"1","enable\_sub":"1"}

}

}

2.1.2 打开悬浮框（由裸眼3D插件默认调用，无需对接，可按需调用）

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void openWidget(int positionX, int positionY) |
| 入参说明 | 设置悬浮框初始化位置，详见表2 |
| 同步回调 | Void |
| 异步回调 | 无 |

表2.参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 默认值 | 说明 |
| positionX | int | 是 | 0 | 悬浮框初始化位置X坐标 |
| positionY | Int | 是 | 手机高度一半 | 悬浮框初始化位置Y坐标 |

2.1.3 关闭悬浮框（由裸眼3D插件默认调用，无需对接，可按需调用）

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void closeWidget() |
| 入参说明 | 无 |
| 同步回调 | Void |
| 异步回调 | 无 |

2.1.4 unity切换“场景”（这里的场景不对应游戏的scene,而是某种特定裸眼3D参数的情形）时会调用此方法，无需对接

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void enterSScene(String SPUid) |
| 入参说明 | SPUid |
| 同步回调 | Void |
| 异步回调 | 无 |

2.1.5 登录

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void login() |
| 入参说明 | 无 |
| 同步回调 | Void |
| 异步回调 | 登录后的反馈，详见表2回调参数说明 |

表2 回调参数

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 | | | |
| callbackType | String | 是 | 回调接口类型 | | | |
| code | Int | 是 | 登录1、成功 0、失败 | | | |
| data | String | 是 | Data具体数据 | | | |
| 参数名称 | 类型 | 必须 | 说明 |
| userCode | String | 是 | 用户唯一标志 |

示例：

{

"callbackType": "login",

"code": 1,

"data": "{" userCode":" bnVsbF8xMjg"}"

}

2.1.6 退出登录（这里提供接口，2.1.7为操作流程）

一、接口详情

表1.

|  |  |
| --- | --- |
| 方法定义 | void void logout() |
| 入参说明 | 无 |
| 同步回调 | Void |
| 异步回调 | 退出登录后的反馈，详见表2回调参数说明 |

表2 回调参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 |
| callbackType | String | 是 | 回调接口类型 |
| code | Int | 是 | 退出登录1、成功 0、失败 |

示例：

{

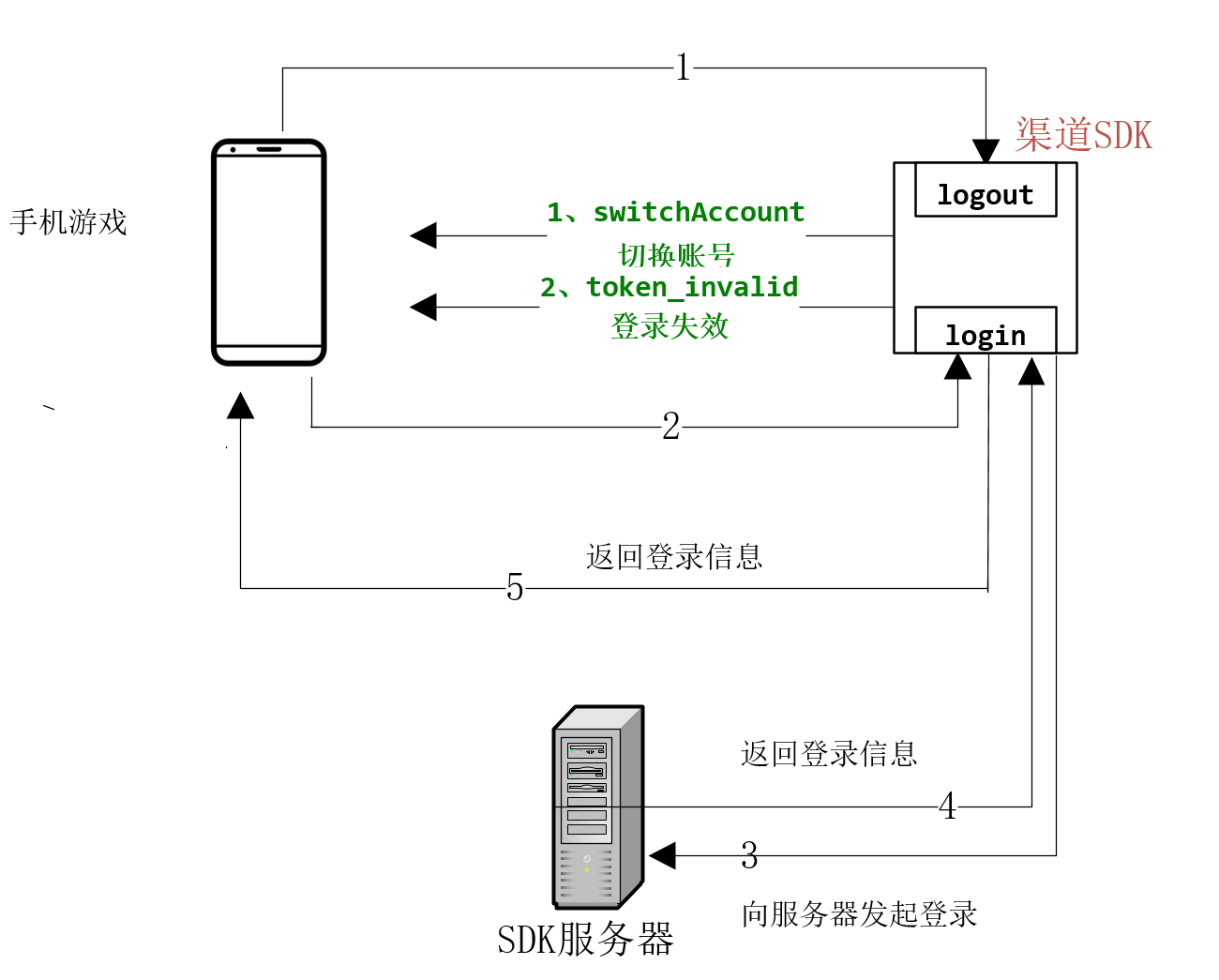
"callbackType": " logout ",

"code": 1

}

2.1.7 SDK退出账号、切换账号（包括切换小账号）

操作流程：



1. Android Sdk 端在界面上切换账号、退出登录、账号token失效时，会callback switchAccount或者token\_invalid消息，格式如表1，当游戏端接收到该callback时，调用游戏端退出登录操作例如保存数据，以及SDK 提供的logout接口，无需等结果返回
2. 游戏端调用SDK提供login接口
3. SDK向SDK服务器发起登录
4. SDK服务器返回用户信息
5. SDK callback 登录信息至游戏，游戏收到成功消息后，做相应处理

表1 回调参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 |
| callbackType | String | 是 | 回调接口类型 |
| code | Int | 是 | 切换账号1、成功 0、失败 |

示例：

{

"callbackType": " switchAccount ",

}

2.1.8 上传游戏中角色信息

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void setRoleInfo(String roleInfo) |
| 入参说明 | roleInfo 角色信息，为json结构，详见表2参数说明 |
| 同步回调 | Void |
| 异步回调 | 提交角色信息后的反馈，详见表3回调参数说明 |

表2.参数说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 默认值 | 说明 |
| userId | int | 是 | 0 | 用户id 由SDK自动填充 |
| app\_id | Int | 是 | 0 | 游戏id 由SDK自动填充 |
| role\_type | Int | 是 | 1 | 数据类型，1为进入游戏，2为创建角色，3为角色升级，4为退出 5 充值 |
| server\_id | String | 是 | 0 | 区服id |
| server\_name | String | 是 | 0 | 区服 |
| role\_id | String | 是 | 0 | 角色id |
| role\_name | String | 是 | 0 | 角色 |
| role\_level | String |  |  | 等级 |
| role\_vip | String |  |  | 玩家vip等级，如果没有，请填0。 |
| party\_name | String |  |  | 工会 帮派名称 |
| role\_balence | Float |  |  | 游戏币余额 |
| rolelevel\_ctime | Int |  |  | 角色创建时间 |
| rolelevel\_mtime | Int |  |  | 角色升级时间 |
| subId | Int |  |  | 小号id由SDK自动填充 |

示例：

{“userId”:’ 11’, “app\_id”:’ 11’, “role\_type”:’ 11’, “server\_id”:’ 11’, “server\_name”:’ 11’, “role\_id”:’ 11’, “role\_name”:’ 11’, “subId”:’ 11’, “role\_level”:’ 11’, “role\_vip”:’ 11’, “party\_name”:’ 11’, “role\_balence”:’ 11’, “rolelevel\_ctime”:’ 11’, “rolelevel\_mtime”:’ 11’}

表3 回调参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 |
| callbackType | String | 是 | 回调接口类型 |
| code | Int | 是 | 1、上传角色信息成功 0、失败 |

示例：

{

"callbackType": "setRole",

"code": 1

}

二、C#参考代码：

JsonData roleInfo = new JsonData();

roleInfo["role\_type"] = 1;

roleInfo["server\_id"] = "123.23.23";

roleInfo["server\_name"] = "123";

roleInfo["party\_name"] = "partyname";

roleInfo["role\_id"] = "Server\_id";

roleInfo["role\_level"] = "111";

roleInfo["role\_vip"] = "222";

roleInfo["role\_balence"] = 12f;

roleInfo["role\_name"] = "qwe";

System.DateTime currentTime = new System.DateTime();

roleInfo["rolelevel\_ctime"] = "" + currentTime.Millisecond / 1000;

roleInfo["rolelevel\_mtime"] = "" + currentTime.Millisecond / 1000;

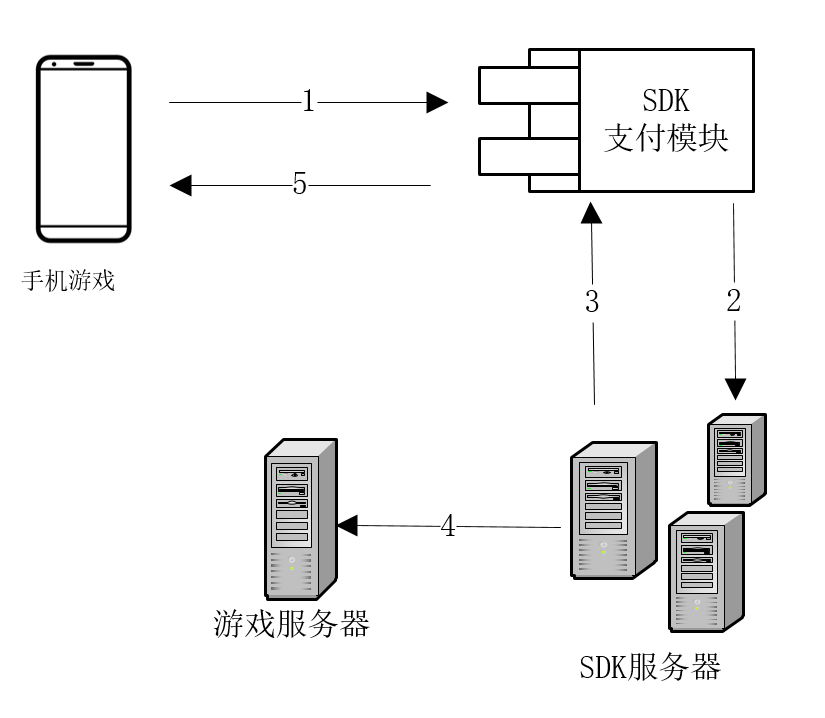
AndroidJavaObject o = new AndroidJavaObject(androidapiclass);

o.CallStatic("setRoleInfo", roleInfo.ToJson());

注意：其中userId、app\_id、subId由android sdk端自动设置，不需要游戏里面设置

2.1.9 调用支付接口

***支***付流程

******

1. 游戏客户端调用SDK提供的支付接口发起支付；
2. SDK客户端调用相应支付方式引导用户完成支付；
3. 支付完成或失败、终止后，SDK客户端会返回支付结果给应用客户端的支付模块；
4. 支付成功后，SDK服务端通过[服务篇1.1](#支付成功对游戏商的通知接口)支付结果通知接口将支付结果发送至游戏方提供的游戏服务端支付结果通知地址；
5. SDK将支付结果callback 给游戏

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void showPay(String payInfo) |
| 入参说明 | PayInfo支付信息，为json结构，详见表2参数说明 |
| 同步回调 | Void |
| 异步回调 | 支付成功失败信息，详见表3回调参数说明 |

表2.参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 默认值 | 说明 |
| device\_id | int | 是 | 0 | 设备ID，由SDK自动填充 |
| app\_id | Int | 是 | 0 | 游戏id，由SDK自动填充 |
| goods\_type | Int | 是 | 1 | 商品类型(1,2,3) |
| goods\_name | String | 是 | 0 | 商品名称 |
| goods\_sn | String | 是 | 0 | 商品编号 |
| goods\_number | Int | 是 | 0 | 商品数量 |
| goods\_price | Double | 是 | 0 | 商品单价 |
| goods\_game\_gold | String |  |  | 虚拟货币金额 |
| agentId | Int |  |  | 渠道 |

表3回调参数

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 | | | |
| callbackType | String | 是 | 回调接口类型 | | | |
| code | Int | 是 | 支付信息返回 1成功 0失败 | | | |
| data | String | 是 | Json数据，具体参数说明： | | | |
| 参数名称 | 类型 | 必须 | 说明 |
| orderNumber | String | 是 | 订单号 |
| userCode | String | 是 | 用户唯一识别号 |
| appId | String | 是 | 游戏应用ID |
| orderAmount | String | 是 | 订单价格 |
| orderTitle | String | 是 | 商品名称 |
| payTime | String | 是 | 支付时间 |
| goodsSn | String | 是 | 商品编号 |
| gameName | String | 是 | 游戏名称 |

示例：

此方法返回以下参数

{

"callbackType": "pay",

"code": 1,

"data":{"appId":"","orderNumber":"A201803261329277068",

"userCode":"bnVsbF8xMjg","orderAmount":"0.01","orderTitle":"memeyu","payTime":"","goodsSn":"1","gameName":""}

}

二、C#参考代码：

JsonData jd = new JsonData();

jd["goodsNumber"] = 1;

jd["goodsSn"] = "1";

jd["goodsPrice"] = 0.01f;

jd["goodsName"] = "金币";

jd["goodsGameGold"] = "100";

jd["goodsType"] = 1;

jd["agentId"] = 945;

AndroidJavaObject o = new AndroidJavaObject(androidapiclass);

o.CallStatic("showPay", jd.ToJson())

2.1.10 返回屏类型 （由裸眼3D插件默认调用，无需对接，可按需调用）

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | int getScreenType() |
| 入参说明 | 无 |
| 同步回调 | 屏幕交织类型：普通屏 0；直排左右1；直排右左 2；斜排 3; |
| 异步回调 | 无 |

此方法通过两种方式获取屏幕类型

1、后台服务器 当应用打开之后 会向后台发送验证设备请求，后台会根据机型返回相关参数例如直排、斜排、是否为异常机型，如果是正常机型则根据后台返回的参数判断屏幕类型，如果是异常机型或者根据后台返回的参数判断的屏幕类型有问题则通过方式2获取；

2、本地jar

注意：当手机中配置文件标志位debug，则直接使用第二种方法

**2.2以下为ANDROID自行调用接口无需对接**

2.2.1 void getConfigUrL()

下载游戏参数

2.2.2验证设备

一、接口详情：

表1.

|  |  |
| --- | --- |
| 方法定义 | void deviceCheck() |
| 入参说明 | 无 |
| 同步回调 | Void |
| 异步回调 | 获取设备信息成功失败信息，详见表2回调参数说明 |

表2回调参数

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 | | | |
| callbackType | String | 是 | 回调接口类型 | | | |
| code | Int | 是 | Init请求通过网络请求设备信息1、成功 0、失败 | | | |
| data | String | 是 | Json数据，具体参数说明： | | | |
| 参数名称 | 类型 | 必须 | 说明 |
| ScreenType | String | 是 | 屏幕交织类型：  普通屏 0，  直排左右1;  直排右左 2;  斜排 3; |
| ScreenW | String | 是 | 屏幕宽 |
| ScreenH | String | 是 | 屏幕高 |
| default\_parallax | String | 是 | 默认3D调节参数 |
| ConfigPath | String | 是 | 3D配置文件下载地址 |

示例：

{

"callbackType": "deviceCheck ",

"code": 1,

"data": {

"ScreenType": "2",

"ScreenW": "1920",

"ScreenH": "1080",

"default\_parallax": "0.0",

"ConfigPath":"/storage/emulated/0//K3DX/GAME/Z+Game/Config/com.unity.TrainingDay2016/3dconfig.xml"

}

}

**2.3****个别callback**

除了在相关接口中介绍的callkack外，还有以下callback

2.3.1拖动悬浮框进度条 无需对接

回调参数

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 | | | |
| callbackType | String | 是 | 回调接口类型 | | | |
| code | Int | 是 | 1、成功 0、失败 | | | |
| data | String | 是 | Data具体数据 | | | |
| 参数名称 | 类型 | 必须 | 说明 |
| percent | Double | 是 | 滑动条百分比 |

示例：

{

"callbackType": "stereocamerap",

"code": 1,

"data": {

"percent": 0.3400000035762787

}

}

### 3、附录

**3.1** **sdk文件清单**

|  |  |
| --- | --- |
| 文件名 | 用途 |
| wzsdk\_libRelease\_x.x.x.aar | Sdk |
| android-support-v4.jar | Android支持库 |
| gson-2.8.1.jar | Gson用于解析json |
| okhttp-3.9.1.jar okio-1.13.0.jar | 网络请求 |
| picasso-2.5.2.jar | 图片加载jar |
| AndroidManifest.xml |  |

服务端

### 1、API

**1.1 支付结果通知接口**

请求链接: 由游戏商提供从数据库读取

请求数据

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 类型 | 必须 | 说明 |
| orderNumber | String | 是 | 订单号 |
| userCode | String | 是 | 用户唯一识别号 |
| appId | int | 是 | 游戏应用ID |
| orderAmount | double | 是 | 订单价格 |
| orderTitle | String | 是 | 商品名称 |
| payTime | String | 是 | 支付时间 |
| goodsSn | String | 是 | 商品编号 |
| gameName | String | 是 | 游戏名称 |

请求回调 SDK最多请求三次 ,收到success会终止请求

|  |  |
| --- | --- |
| 值 | 用途 |
| success | 收到success会终止请求 |
| fail | 继续请求,总共请求三次 |