PP 单机游戏 SDK

开发参考说明书

修订记录

归档日期	版本	说明	作者
2015-09-07	1.4.2	完善微信支付，无需再安装插件	PP 助手安卓版
2015-08-26	1.4.1	SP 短代支持下行短信验证码方式，完善电信 manifest 声明	PP 助手安卓版
2015-08-17	1.4.0	新增账号登录功能	PP 助手安卓版
2015-08-07	1.3.0	新增运营商四网融合计费 SDK	PP 助手安卓版
2015-08-03	1.2.0	支持微信支付	PP 助手安卓版
2015-07-09	1.1.0	支持支付宝支付，优化短代支付耗时	PP 助手安卓版
2015-06-23	1.0.0	PP 单机游戏 SDK 开发参考说明书	PP 助手安卓版

目录

1. 接入规范（重要必读！！！）.....4
1.1. APPID 和 APIKEY 配置.....4
1.2. 游戏包名&图标规范.....4
1.3. 接口调用要求.....5
1.4. 支付渠道选择.....6
2. 开发环境搭建.....6
2.1. 导入 SDK 工程.....6
2.2. 配置 AndroidManifest.xml.....7
2.3. 放入运营商计费文件（无需运营商支付方式可忽略此章节）.....13
3. API 接口参数说明.....14
3.1. 消息通知机制.....14
3.2. 通知回调参数.....15
4. 编码接入.....19
4.1. 注册环境.....19
4.2. 初始化接口.....19
4.2.1 方法定义.....19

4.2.2 参数说明.....20

4.2.3 使用方式.....20

4.2.4 实例代码.....21

4.3. 品宣闪屏接口.....21

4.3.1 方法定义.....21

4.3.2 参数说明.....22

4.3.3 使用方式.....22

4.3.4 实例代码.....22

4.4. 支付接口.....23

4.4.1 方法定义.....23

4.4.2 参数说明.....23

4.4.3 使用方式.....24

4.4.4 实例代码.....24

4.5. 退弹接口.....25

4.5.1 方法定义.....25

4.5.2 参数说明.....25

4.5.3 使用方式.....26

4.5.4 实例代码.....26

4.6. 用户登录接口.....26

4.6.1 方法定义.....26

4.6.2 参数说明.....27

4.6.3 使用方式.....27

4.6.4 实例代码.....27

5. 代码混淆.....28

附 1.API 类具体使用 demo 工程.....31

1. 接入规范（重要必读！！！）

本文档面向安卓开发者。

1.1. APPID 和 APIKEY 配置

1）使用客户端结果发放道具的 CP

	调试自测阶段	正式提审阶段
APPID	1021	开放平台分配的 APPID

2）使用服务端结果发放道具的 CP

	调试自测阶段	正式提审阶段
APPID	1021	开放平台分配的 APPID
APIKEY	nstjae9dPzYLQM8EZE1ZLvHM mpeB7D8ab56IOaEpBXZsxnsM eiaKGHH4jgG0JDF2	开放平台分配的 APIKEY

注：换成正式 APPID 之后无需再自测了，如自测将出现没有可用的支付方式提示，这是正常现象。CP 只要在测试 id 下支付正常就行，替换成正式 id 后可放心提审。

1.2. 游戏包名&图标规范

1）包名

游戏客户端的 AndroidManifest.xml 中游戏包名定义加上“.pp”后缀。

如原游戏包名为 package="com.lori.game"，则修改为 package="com.lori.game.pp"

2）游戏 icon 图标

使用能表明 PP 渠道的游戏图标，在 AndroidManifest.xml 中指定。如原图标为 android:icon="@drawable/icon"，则修改为 android:icon="@drawable/iconpp"。其中 iconpp 为在原游戏图标基础上制作的能表明 PP 渠道的图标文件。

3）闪屏图片

建议 CP 选择自行闪屏，需按照 SDK 压缩包里附带的 UI 设计规范以及素材进行闪屏设计并替换已有闪屏图片，用以表明来自于 PP 的渠道。

CP 也可调用 SDK 的闪屏品宣接口，用以表明来自于 PP 的渠道。

1.3. 接口调用要求

单机游戏 SDK 主要提供以下 5 个接口（接口具体信息参见后文）：

1) 初始化接口（必选）

2) 品宣闪屏接口（可选，CP 若无闪屏必须调用该接口展示闪屏，CP 若有闪屏可参照 LOGO 文件夹中的规范自行闪屏）

3) 支付接口（可选，支持四网融合支付、短代 SP 支付、支付宝支付、微信支付，平台可根据 CP 需求灵活配置）

4) 登录接口（可选，可通过调用接口获取登录用户信息）

5) 退弹接口（必选）

1.4. 支付渠道选择

若 CP 已有运营商计费文件 ,建议使用运营商三网/四网+微信支付+支付宝三种支付渠道，注意下文中的支付接口 paycode 参数传入指定内容即可走运营商通道。

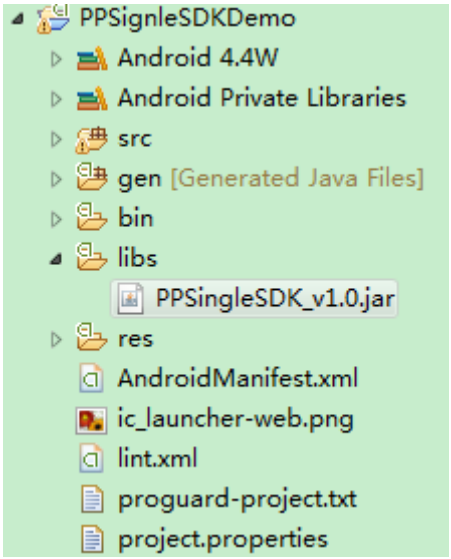
若 CP 没有运营商计费文件，建议使用短代 SP+微信支付+支付宝三种支付渠道，注意下文中的支付接口 paycode 参数不传内容即可走短代 SP 通道。

2. 开发环境搭建

2.1. 导入 SDK 工程

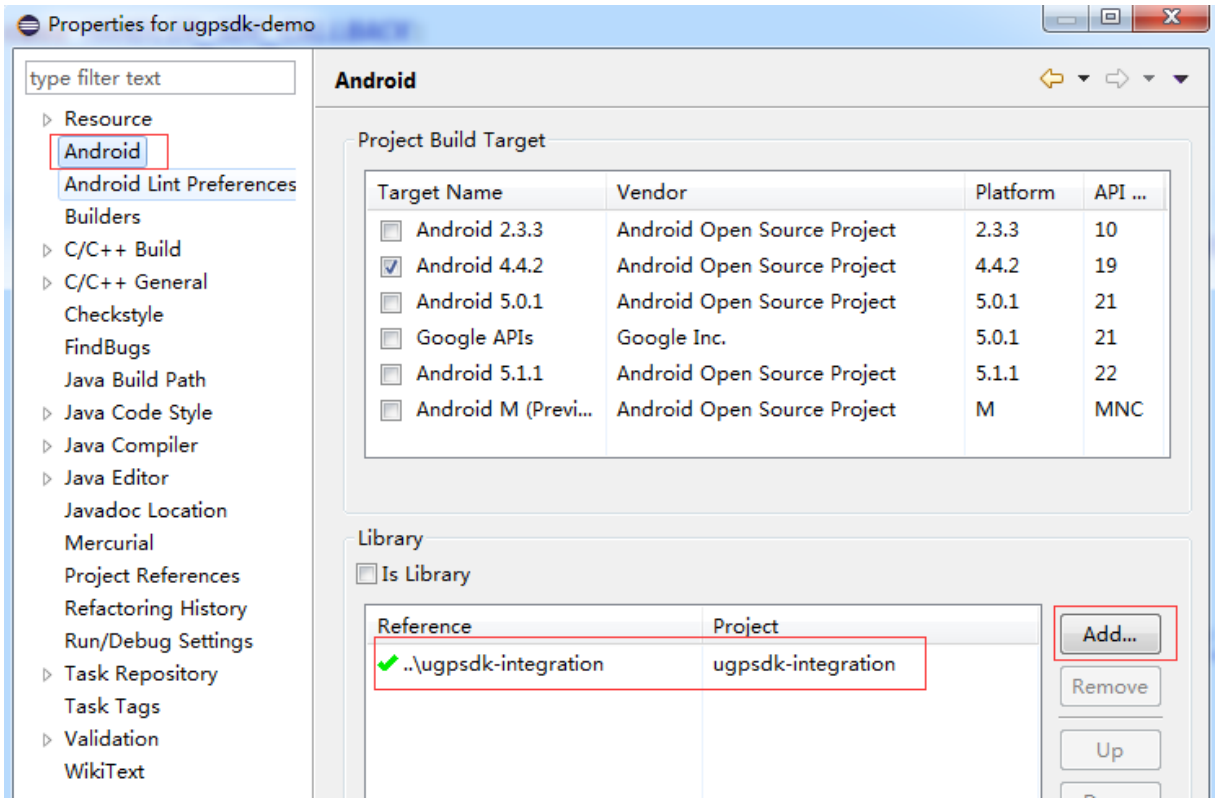
以下以 Eclipse 为例，说明在 Eclipse 中开发游戏时使用游戏 SDK 的环境设置：

1) 将 SDK 压缩包里的 PPSingleSDK_v*.jar 复制到游戏项目的 libs 目录下。



2) 如果有接入运营商支付方式，需要将 ugpsdk-integration 工程导入 Eclipse，并在自己的工程中引用此 lib 工程：

- 游戏工程右键 → Properties → Android → Add → 选取库工程 → OK



注意：我们 integration 工程 libs 目录里面提供了两个文件夹



建议 CP 不要新建其他文件夹（比如 armeabi-v7a），Android 系统目前支持以下七种不同的 CPU 架构：ARMv5，ARMv7，x86，MIPS，ARMv8，MIPS64 和 x86_64，每一种都关联着一个相应的 ABI。运营商支付 SDK 仅仅提供了 armeabi 的 so 库，就算是用户为 ARMv7 架构的系统，如果没有 armeabi-v7a 这个目录，Android 会去 armeabi 寻找对应的 so 库，否则将不会，那么就会导致找不到 so 库的错误。

如果 CP 一定需要新建目录，请将 so 库对应拷贝一份到所建目录（因为 ABI 不同，对 so 库的处理有细微差别，我们不保证拷贝过去的 so 库完全正常使用，请充分自测验证）。

2.2. 配置 AndroidManifest.xml

打开您项目的 AndroidManifest.xml 文件，进行如下修改。

1）包名加上“.pp 后缀”；

2）指定游戏图标为能够表明 PP 渠道的游戏图标（PP 渠道的游戏图标可基于原游戏图标进行制作，能够表明 PP 渠道即可）；

3）增加权限声明（下图所示的权限为 SDK 所必需，游戏可根据需要额外增加其它权限声明）；

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.AUTHENTICATE_ACCOUNTS" />
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.USE_CREDENTIALS" />
<uses-permission android:name="android.permission.READ_CONTACTS" />

<!-- 运营商SDK权限添加开始 -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.GET_PACKAGE_SIZE" />
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="android.permission.CALL_PHONE" />
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
<uses-permission android:name="android.permission.READ_SMS" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
<uses-permission android:name="android.webkit.permission.PLUGIN" />
<!-- 运营商 SDK 权限添加結束 -->
```

4）需要在 AndroidManifest.xml 文件中声明游戏专属的 APPID（测试阶段使用测试

appid1021，正式提审阶段替换成开放平台展示的正式 appid）

```
<meta-data android:name="APPID" android:value="1021"/>
```

5）需要增加单机游戏 SDK 所用的 activity 声明，如下

```
<!— 单机游戏 SDK 声明 -->

<activity
    android:name="pluginsdk.ps.local.PPProxyActivity"
    android:screenOrientation="portrait" />
<activity
    android:name="cn.uc.paysdk.SDKActivity"
    android:configChanges="keyboardHidden|orientation|screenSize"
    android:launchMode="singleTask"
    android:theme="@android:style/Theme.Translucent.NoTitleBar"
    android:windowSoftInputMode="adjustPan|stateAlwaysHidden|adjustResize" />
<activity
    android:name="cn.uc.gamesdk.sa.activity.ProxyActivity"
    android:configChanges="keyboardHidden|orientation|screenSize"
    android:theme="@android:style/Theme.Translucent.NoTitleBar"
    android:windowSoftInputMode="adjustResize" >
    <intent-filter>
        <action android:name="cn.uc.gamesdk.sdkweb" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>

<!-- End of 单机游戏 SDK 声明 -->
```

6）需要增加单机游戏 SDK 内部 service 声明，如下

```
<service android:name="cn.uc.paysdk.service.SDKService" />
```

7）需要增加支付宝使用的 activity 声明，如下

```
<!-- 支付宝需要的activity声明 -->
```

```
    <activity
        android:name="com.alipay.sdk.app.H5PayActivity"
        android:configChanges="orientation|keyboardHidden|navigation"
        android:exported="false"
        android:screenOrientation="behind" />
    <activity
        android:name="com.alipay.sdk.auth.AuthActivity"
        android:configChanges="orientation|keyboardHidden|navigation"
        android:exported="false"
        android:screenOrientation="behind" />
    <!-- 支付宝需要的activity声明 -->
```

8）需要增加 now 微信 SDK 支付的 activity 声明

```
    <!-- now 微信SDK支付相关添加开始 -->
    <activity
        android:name="com.ipaynow.plugin.activity.PayMethodActivity"
        android:theme="@android:style/Theme.Dialog"
        android:configChanges="orientation|keyboardHidden|screenSize"
        android:exported="false"
        android:screenOrientation="unspecified"/>
     <activity
android:name="com.ipaynow.plugin.inner_plugin.wechat_plugin.activity.WeChatNotifyAc
tivity"
        android:theme="@android:style/Theme.NoDisplay"
        android:configChanges="orientation|keyboardHidden|screenSize"/>
    <!-- now 微信 SDK 支付相关添加结束 -->
```

9）需要增加运营商支付需要的 activity 声明（无需运营商支付方式可忽略此节），如下

9.1）移动 MM 接入，需要添加 APPID 和 APPKEY

```
<!-- 如果支持移动MM支付需添加下面的值 -->
<meta-data
    android:name="MMBILLING_APPID"
    android:value="appid:300008973569" />
<meta-data
    android:name="MMBILLING_APPKEY"
```

```
    android:value="appkey:044B0F69808C6151552A90ACF757A323" />
```

9.2）移动南京基地接入，需要将下面四行加到启动 Activity 即可（参见 demo）

```
<!-- 移动游戏基地移动游戏基地相关添加开始 -->
<intent-filter>
    <action android:name="android.intent.action.CHINAMOBILE_OMS_GAME" />
    <category android:name="android.intent.category.CHINAMOBILE_GAMES" />
</intent-filter>
<!-- 移动游戏基地移动游戏基地相关添加结束 -->
```

9.3）电信接入，需要根据实际情况修改渠道号，并添加以下 Activity 声明

```
    <!-- 中国电信 SDK 支付相关添加开始 -->
    <!-- 中国电信渠道号 -->
    <meta-data android:name="EGAME_CHANNEL" android:value="80010395" />
    <meta-data android:name="EGAME_UNINS" android:value="false" />
    <meta-data android:name="EGAME_ALIPAY" android:value="false" />
    <meta-data android:name="EGAME_UNIPAY" android:value="false" />
    <activity
           android:name="cn.egame.terminal.paysdk.EgamePayActivity"
           android:configChanges="orientation|keyboard|keyboardHidden"
           android:screenOrientation="portrait"
           android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen" />
    <activity
        android:name="cn.play.dserv.EmpActivity"
        android:configChanges="keyboard|keyboardHidden|orientation"
        android:exported="true" />
    <service
        android:name="cn.play.dserv.DService"
        android:enabled="true"
        android:exported="false"
        android:label="dservice"
        android:process=":dservice_v1" />
    <receiver
        android:name="cn.play.dserv.DsReceiver"
        android:process=":dservice_v1" >
        <intent-filter>
            <action android:name="cn.play.dservice" />
        </intent-filter>
```

```
    </receiver>
    <!-- 中国电信 SDK 支付相关添加结束 -->
```

9.4）联通接入，需要修改下面黄色背景字段标记的前缀为实际应用的包名

```
<!-- 中国联通SDK支付相关添加开始 -->
<service
android:name="com.unicom.wostore.unipay.paysecurity.SecurityServiceFramework" >
    <intent-filter>
        <!-- 请务必将action name 按照应用包名
+".unicom.wostore.unipay.securityserviceframework"进行组装 -->
        <action android:name="您的包名.unicom.wostore.unipay.securityserviceframework"
/>
    </intent-filter>
</service>

<meta-data android:name="wostore_billing_otherpay" android:value="false" />
<meta-data android:name="wostore_billing_chinaunicom" android:value="true" />
<meta-data android:name="wostore_billing_chinamobile" android:value="false" />
<meta-data android:name="wostore_billing_chinatele" android:value="false" />
<!-- 中国联通SDK支付相关添加结束 -->
```

10）需要增加分辨率适配设置，如下

```
    <supports-screens
        android:anyDensity="true"
        android:largeScreens="true"
        android:normalScreens="true"
        android:resizeable="true"
        android:smallScreens="true" />
```

完整的 AndroidManifest .xml 参考文件请查看 demo 工程。

2.3. 放入运营商计费文件（无需运营商支付方式可忽略此章节）

计费文件支付中国移动（移动 MM、南京基地）、中国电信、中国联通等 4 种支付渠道，如果无计费文件并且需要运营商支付方式请咨询到相应的运营商的开发者社区进行申请申报或咨询商务人员。

1. 中国移动南京基地：

在 ugpsdk-integration 工程的 assets 目录下包含移动南京基地支付 sdk 登录及计费相关数据文件。游戏开发者需要将该目录下所有的文件拷贝到游戏工程的 assets（因为打包不会将 lib 工程的 assets 打进去），该目录下提供了测试使用的 Charge.xml、ConsumeCodeInfo.xml，待游戏提交至移动平台后，请替换该文件内容（替换成游戏实际业务及计费信息）。

注意：该两个文件是计费的核心数据，请勿擅自修改其内容，否则计费失败。另：请将 ugpsdk-integration\assets\Config.xml 也一并复制到游戏 assets 目录，该文件与计费无关，是 SDK 的版本标识，但是必须包含。

2. 中国电信：

如果需要接入中国电信 SDK 短信支付，需要到电信游戏平台进行申报，然后下载系统生成的配置文件，即“feeInfo.dat”，并将其放置游戏工程的 assets 目录，才能正常进行中国电信单机 sdk 短信支付。

3. 中国联通：

如果需要接入中国联通 SDK 短信支付，需要到联通沃商店平台进行申报，然后得到包含计费文件的 jar 包“Multimode_UniPay_base.jar”和“Multimode_UniPay_payinfo.jar”并替换“ugpsdk-integration”工程中对应的 jar 包即可。

3. API 接口参数说明

3.1. 消息通知机制

单机游戏 SDK 目前分为品宣，支付和登录 3 大模块，分别采取以下形式进行消息通知。

1） 品宣，采用回调接口的形式通知 CP 对应品宣操作完成的消息。抽象方法定义在接口 PPBrandPubCallback 接口类中。

```
public interface PPBrandPubCallback {
    /**
     * 对应品宣操作处理完成
     */
    void onBrandPubCompleted();
}
```

2） 支付，采用事件监听的方式通知 CP 支付模块相关消息。抽象方法定义在接口 SDKCallbackListener 接口类中。

注意：回调方法可能在任何线程中回调，UI 相关的处理务必抛到主线程中进行处理

```
public interface SDKCallbackListener {
    /**
     * 支付模块相关操作成功. 该方法将有可能在任何线程中被回调
     * @param status 响应状态
     * @param response 响应数据实体
     */
    public void onSuccessful(int status, Response response);

    /**
     * 支付模块操作失败. 该方法将有可能在任何线程中被回调
     * @param error 错误数据实体
     */
    public void onErrorResponse(SDKError error);
}
```

3） 登录，采用回调接口的形式通知 CP 用户登录状态的消息。

```
public interface PPOnLoginCallback {
```

```
    /**
     * 登录成功回调
     * @param user 登录的用户bean
     */
    public void onSuccess(PPLoginUser user);
    /**
     * 登录失败回调
     * @param errorCode 失败码
     * @param errorMsg 失败信息
     */
    public void onFailure(int errorCode, String errorMsg);
    /**
     * 用户取消登录回调
     */
    public void onCancel();

}
```

3.2. 通知回调参数

具体回调参数说明如下:

参数	说明
status	状态码,定义在 SDKStatus 中,具体的可以参考下表说明. (CP 方具体接入时无需使用该字段做判断)
response	响应数据实体

参数 status 状态说明:

状态码	说明
SUCCESSFUL (1)	API 调用成功，表明操作已成功完成。
FAILURE (0)	API 调用失败

NO_INIT（-1）	表明没有初始化的时候不能执行 API 接口调用。如果返回此状态码，游戏需要先执行初始化操作。

参数 Response 信息说明:

函数名称	含义	
getStatus()	SDKStatus.SUCCESSFUL	成功标识
	SDKStatus.FAILURE	失败标识
getType()	Response.LISTENER_TYPE_INIT	SDK 初始化成功
	Response.LISTENER_TYPE_PAY	订单支付成功（含历史订单）
	Response.LISTENER_TYPE_EXIT	SDK 退出成功
getData()	当类型为**订单支付成功**时，返回订单的 JSON 数据，其它类型时为空值	
getTradeId()	订单交易流水号	
setMessage(String)	Response.OPERATE_SUCCESS_MSG	“00”表示接收成功
	Response.OPERATE_FAIL_MSG	“01”表示接收失败
	注意：该值由游戏方在回调类型为**订单支付成功**的回调函数中响应，代表游戏方已经成功收到响应消息。如果游戏不设置值，支付 SDK 会认为游戏没有接收到消息，会按相应周期和时机回调订单消息，直到游戏方确认。游戏应该验证收到此回调信息后，调用 **setMessage(Response.OPERATE_SUCCESS_MSG)** 确认收到。	

订单 json 数据结构如下：

KEY 值	说明

PayResponse.PAY_MONEY	支付的金额	
PayResponse.CP_ORDER_ID	CP 订单号	
PayResponse.TRADE_ID	交易号	
PayResponse.PRO_NAME	商品名称	
PayResponse.PAY_TYPE	支付类型	101 运营商支付 207 支付宝钱包支付 208 微信支付（插件式） 209 微信支付 305 移动短信支付 306 联通短信支付 307 电信短信支付 800 银行卡支付
PayResponse.ORDER_STATUS	订单状态	00 成功 01 失败 02 退款
PayResponse.ORDER_FINISH_TIME	订单完成时间	
PayResponse.EXT_INFO	商品扩展信息（如游戏币的数量，道具的描述等）该信息为后台配置后返回的信息	
PayResponse.ATTACH_INFO	支付时（SDKProtocolKeys.ATTACH_INFO）对应的附加透传信息（按原样返回）	

订单 json 数据示例如下：

```
{
```

```
  "CP_ORDER_ID": "1242514254154154",  // CP订单号
  "ORDER_FINISH_TIME": "20140814114005", // 订单完成时间
  "ORDER_STATUS": "00", // 订单状态 00 成功 01 失败 99 退款
  "PAY_TYPE": "301", // 支付类型-101 运营商支付，207 支付宝
  "PRO_NAME": "100 game coins",  // 商品名称
  "TRADE_ID": "20140814113934170012", // 交易号
  "PAY_MONEY": "50.000", // 支付金额
  "EXT_INFO": "ext_info_sample", // 商品扩展信息
  "ATTACH_INFO": "attach_info_sample" // 附加透传信息
}
```

参数 error 结构如下：

函数	说明
getCode()	错误码定义（详见 SDKStatus 中错误码的定义）
getMessage()	错误信息内容

SDKStauts 错误码一览表：

错误码	说明
ERROR_CODE_UNINITIALIZED	未成功初始化支付 SDK
ERROR_CODE_PARAMS_INVALID	传入参数不合法
ERROR_CODE_UNAUTHRIZE	SDK 未正常授权
ERROR_CODE_AUTHRIZE_ERROR	SDK 服务器授权过程中发生错误
ERROR_CODE_NULL_CALLBACK_METHOD	无回调方法
ERROR_CODE_CONTEXT_IS_NULL	上下文对象为空
ERROR_CODE_INTENT_IS_NULL	意图对象为空

ERROR_CODE_PARAMS_IS_NULL	传入参数为空
ERROR_CODE_PAY_FAIL	支付过程发生失败
ERROR_CODE_SDK_RESOURCE_ERROR	SDK 资源内容发生错误

4. 编码接入

4.1. 注册环境

如果您需要接入运营商 SDK 支付， 那么您需要在程序 Application 类（此 Application 必须在 AndroidManifest 文件的 Application 标签中定义的那个）里面的 onCreate()方法， 通过 PPSingleSDK.registerEnvironment 方法来注册环境信息，此方法无回调信息。

```
/**
  * 注册环境信息，如果有接入运营商才需要调用此方法
  * @param application
  * @return
  */
public static void registerEnvironment(Application application)
```

4.2. 初始化接口

4.2.1 方法定义

```
/**
 * 初始化单机游戏SDK
 * @param context 不得传空，如果在Activity里调用请传Activity的context
 * @param listener
 *          支付模块监听器，可为空，如果为空则只作初始化操作，不为你更新设置新的listener
 */
public void initSingleSDK(final Context context, final SDKCallbackListener listener)
```

4.2.2 参数说明

参数	说明
context	Application 或 Activity 的 context 不得传空，如果在 Activity 里调用请传 Activity 的 context
listener	支付模块监听器 可为空，如果为空依然会进行初始化操作，如果当前已经设置过监听器了，则会将初始化结果回调给当前设置的监听器

4.2.3 使用方式

使用方式有如下 2 种，游戏方可根据自身情况选取适合自己的方式：

A. 全局统一处理支付模块消息回调（建议方式）

1. 必须在 Application 或游戏主页面（菜单页面）的 onCreate()方法里调用初始化接口

2. 传入实例化的监听器接口，接收支付模块所有信息并统一处理。

B. 在需支付功能的界面处理支付模块消息回调

1. 必须在 Application 的 onCreate()方法里调用初始化接口，监听器可传空。

2. 在所有需支付功能的 Activity 的 onCreate 方法里再次调用初始化接口，并传入实例化的监听器接口，那么最后调用初始化接口的 activity，就会收到后续支付模块的消息回调。

3. 该方法建议直接在所有支付 Acitivity 的基类里统一调用，每次调用初始化接口，都会有一次初始化成功或失败的回调。比如游戏方可根据初始化成功的回调才展示支付按钮等需求。

4.2.4 实例代码

```
PPSingleSDK.getInstance().initSingleSDK(this, new SDKCallbackListener() {
    @Override
    public void onSuccessful(int status, Response response) {
        Log.e(TAG, "onSuccessful -> " + response);
        switch (response.getType()) {
            // 初始化成功
            case Response.LISTENER_TYPE_INIT:
                Log.d(TAG, "SDK did init successfully ");
                    break;
            // 下单成功确认(包括补单成功确认)
            case Response.LISTENER_TYPE_PAY:
                Log.i(TAG, "SDK did pay successfully");
                // 确认已收到订单成功信息（一定要确认啊）
                response.setMessage(Response.OPERATE_SUCCESS_MSG);
                break;
            // 退出成功
            case Response.LISTENER_TYPE_EXIT:
                Log.d(TAG, "SDK did exit successfully");
                break;
        }
        // 注意：这边回调过来时是在子线程,UI操作请自行抛到主线程中执行
    }
    @Override
    public void onErrorResponse(SDKError error) {
        Log.e(TAG, "onErrorResponse -> " + error);
        switch (error.getCode()) {

        }
    }
});
```

4.3. 品宣闪屏接口

4.3.1 方法定义

```
/**
```

```
 * 展示品宣闪屏
 * @param context
 * @param callback 闪屏回调，如果不需要单机游戏SDK的闪屏，可传空
 */
public void showBrandPublicity(
            final Activity context, final PPBrandPubCallback callback)
```

4.3.2 参数说明

参数	说明
context	必须传 Activity 的 context
callback	闪屏回调，注意：如果不需要单机游戏 SDK 的闪屏，可传空

4.3.3 使用方式

1. 在应用的配置的 LaunchActivity 的 onCreate 方法里调用该接口
2. 传入实例化的品宣回调接口
3. 在回调接口里再展示本身 Activity 的界面,或跳转到游戏界面

4.3.4 实例代码

```
PPSingleSDK.getInstance().showBrandPublicity(this, new PPBrandPubCallback() {
    /*
     * 对应品宣操作处理完成
     * 该回调方法在主线程中回调
     */
    @Override
    public void onBrandPubCompleted() {
        startActivity(new Intent(LaunchActivity.this, MainActivity.class));
        finish();
```

```
    }
});
```

4.4. 支付接口

4.4.1 方法定义

```
/**
 * 调用支付功能
 * @param context 不得传空，必须传Activity的context
 * @param data 不得传空，在data里设置支付相关参数
*/
public void doPayment(final Activity context, final Intent intent)
```

4.4.2 参数说明

参数	说明
context	必须传 Activity 的 context
intent	不得传空，用 putExtra 向 intent 里设置支付相关参数

支付参数说明（参数的属性值在 SDKProtocolKeys 中定义）：

Intent 的 KEY	说明
CP_ORDER_ID	可选参数，客户的唯一订单号，最多 50 字符 （订单号需唯一，对多笔订单使用同一个订单号支付会导致支付失败）
AMOUNT	必选参数，订单金额
PRODUCT_NAME	必选参数，商品（道具）名称，最多 50 字符
APP_NAME	必选参数，应用（游戏）名称

NOTIFY_URL	可选参数，服务器通知地址，用于通知游戏服务器支付结果，详见《PP 单机游戏 SDK_开发参考说明书_服务器接口》
ATTACH_INFO	可选参数， 透传信息（透传），最多 50 字符，字符集为[a-zA-Z0-9_]字母下划线组合，用于 CP 附带支付相关信息参数，在支付结果完成时会直接回调信息参数
PAY_CODE	可选参数，如果需要运营商支付能力，则此参数为运营商 SDK 支付必选参数，运营商支付平台提供的计费代码（需要根据 sim 卡号传入不同运营商平台提供的 paycode，判断运营商的方法见代码示例里面的 getOperatorByMnc 方法。如果是移动的卡，因为移动有移动 MM 和南京基地两个支付 sdk，所以：如果 payCode.lennth == 3；走南京基地支付；否则走移动 MM 支付）
TOOLS_DESC	中国电信 SDK 支付可选参数：商品描述，最多200字符

4.4.3 使用方式

1. 封装指定参数到 intent 里，注意**订单金额**参数类型为 String 类型
2. 调用支付接口，支持短信、支付宝和微信支付
3. 在已有的支付模块监听器的实例里，等待结果通知

4.4.4 实例代码

```
Intent intent = new Intent();
// 可选参数，客户的唯一订单号(订单号需唯一，对多笔订单使用同一个订单号支付会导致支付失败)
intent.putExtra(SDKProtocolKeys.CP_ORDER_ID,
String.valueOf(System.currentTimeMillis()));
```

```
// 必选参数，订单金额
intent.putExtra(SDKProtocolKeys.AMOUNT, "2");
// 必选参数，商品（道具）名称
intent.putExtra(SDKProtocolKeys.PRODUCT_NAME, "测试游戏");
// 必选参数，应用（游戏）名称
intent.putExtra(SDKProtocolKeys.APP_NAME, "测试商品");
// 可选参数，服务器通知地址
intent.putExtra(SDKProtocolKeys.NOTIFY_URL, "");
// 可选参数，透传信息（透传）
intent.putExtra(SDKProtocolKeys.ATTACH_INFO, "");
// 如果支持运营商sdk短代支付，那么需要先判断用户手机号，然后传入相对应的paycode
// demo现在是写死的paycode，实际需要cp传计费点
intent.putExtra(SDKProtocolKeys.PAY_CODE, getPaycode(getApplicationContext()));
// 中国电信SDK支付可选参数：商品描述
intent.putExtra(SDKProtocolKeys.TOOLS_DESC, "中国电信商品描述");
PPSingleSDK.getInstance().doPayment(MainActivity.this, intent);
```

4.5. 退弹接口

4.5.1 方法定义

```
/**
 * 展示退弹应用推荐
 * @param context
 * @param callback
 */
public void showExitGameWindow(
            final Activity context, final PPBrandPubCallback callback)
```

4.5.2 参数说明

参数	说明
context	不得传空，必须传 Activity 的 context
callback	不得传空，当用户点击确认离开游戏会触发该回调

4.5.3 使用方式

1. 可选择重写 Activity onBackPressed 方法里调用该接口
2. 传入实例化的品宣回调接口
3. 在回调接口里 finish()掉自身，也可选择 kill 掉游戏进程

4.5.4 实例代码

```
@Override
public void onBackPressed() {
    /*
     * 展示退弹应用推荐
     * context 不得传空，必须传Activity的context
     * callback 不得传空，当用户点击确认离开游戏会触发该回调
     */
    PPSingleSDK.getInstance().showExitGameWindow(this, new PPBrandPubCallback() {
        @Override
        public void onBrandPubCompleted() {
            // 在回调接口里finish()掉自身，也可选择kill掉游戏进程
            finish();
            Process.killProcess(Process.myPid());
        }
    });
}
```

4.6. 用户登录接口

4.6.1 方法定义

```
/**
* 登录接口
* @param activity 传入当前界面的activity
* @param callback 登录回调
*/
public void login(final Activity activity, final PPOnLoginCallback callback)
```

4.6.2 参数说明

参数	说明
activity	不得传空，传入当前界面的 Activity
callback	不得传空，会回调用户的登录结果信息，包括 onSuccess，onFailed，onCancel 三种情况

回调的 PPLoginUser 对象包含以下数据

数据属性	获取方法	是否非空
用户 id	getUid	不为空
用户 token	getUserToken	不为空
用户头像 url	getAvatar	可能为空
用户昵称	getNickName	可能为空
用户性别	getGender	可能为空

4.6.3 使用方式

1. 在 Activity 中，需要用户信息的地方调用登录接口，传入回调接口；

2. 在回调接口的结果中处理回调情况，注意先判断 Activity.isFinishing()，如果界面不存在则不再处理 UI 相关事务，以免回调回来界面已被销毁，导致崩溃；

4.6.4 实例代码

```
PPSingleSDK.getInstance().login(this, new PPOnLoginCallback() {
```

```
    @Override
    public void onSuccess(PPLoginUser user) {
        Log.i(TAG, "login onSuccess -> " + user);
        showToastMessage(MainActivity.this, "登录成功");
    }

    @Override
    public void onFailure(int errorCode, String errorMessage) {
        showToastMessage(MainActivity.this, "登录失败 -> " + errorMessage);
    }

    @Override
    public void onCancel() {
        showToastMessage(MainActivity.this, "用户取消登录");
    }
});
```

5. 代码混淆

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。

```
#不对类库进行压缩和优化
-dontshrink
-dontoptimize

#不警告找不到该类，因为该类被系统隐藏
-dontwarn android.os.SystemProperties

#android 基础组件相关不混淆
-keep public class android.** { *; }
-keep public class * extends android.app.Activity
-keep public class * extends android.app.Application
-keep public class * extends android.app.Service
-keep public class * extends android.content.BroadcastReceiver
-keep public class * extends android.content.ContentProvider
-keep public class * extends android.app.backup.BackupAgentHelper
-keep public class * extends android.preference.Preference
```

```
-keep public class com.android.vending.licensing.ILicensingService

#android 序列化类不混淆
-keep class * implements android.os.Parcelable {
    public static final android.os.Parcelable$Creator *;
}

-keepclassmembers class * implements java.io.Serializable {
    static final long serialVersionUID;
    static final java.io.ObjectStreamField[] serialPersistentFields;
    private void writeObject(java.io.ObjectOutputStream);
    private void readObject(java.io.ObjectInputStream);
    java.lang.Object writeReplace();
    java.lang.Object readResolve();
}

#枚举类及其子类不混淆
-keepclassmembers enum * {
   public static **[] values();
   public static ** valueOf(java.lang.String);
}

-keepclassmembers class * extends java.lang.Enum {
    public static **[] values();
    public static ** valueOf(java.lang.String);
}

#包含 native 方法的类不混淆
-keepclasseswithmembernames class * {
   native <methods>;
}

#sdk 相关组件不混淆
-keep class api.** { *; }
-keep class com.pp.** { *; }
-keep class com.lib.commom.tool.** { *; }
-keep class com.single.sdk.** { *; }
-keep class cn.uc.paysdk.** { *; }
```

```
-keep class com.alipay.** { *; }
-keep class com.ut.*

#微信支付混淆添加开始
-keep public class com.ipaynow.plugin.utils.MerchantTools{ <fields>; <methods>; }
-keep public class com.ipaynow.plugin.utils.PreSignMessageUtil{ <fields>; <methods>; }
-keep public class com.ipaynow.plugin.api.IpaynowPlugin{ <fields>; <methods>; }
#微信支付混淆添加结束

#中国移动 MM 混淆添加开始（无需运营商支付方式可忽略以下混淆代码）
-keep class com.ccit.** {*; }
-keep class ccit.** { *; }
-keep class mm.purchasesdk.**
-keep class mm.purchasesdk.** {*;}
-keep class com.chinaMobile.**
-keep class com.chinaMobile.** {*;}
#中国移动 MM 混淆添加开始结束

#中国移动南京游戏基地混淆添加开始（无需运营商支付方式可忽略以下混淆代码）
-keep public interface cn.cmgame.billing.api.GameInterface$IPayCallback {*;}
-keep class cn.cmgame.billing.api.GameInterface.** {*;}
-keep class cn.cmgame.**
-keep class cn.cmgame.**{*; }
#中国移动南京游戏基地混淆添加结束

#中国联通混淆添加开始（无需运营商支付方式可忽略以下混淆代码）
-libraryjars  libs/Multimode_UniPay_base.jar
-keep class com.unipay.account.**{*;}
-keep class com.unicom.dcLoader.**{*;}
-keep class com.unicom.wostore.unipay.paysecurity.**{*;}
-keep class com.unipay.secret.jni.**{*;}
-keep class com.wow.shell.**{*;}
-keep class cn.egame.terminal.miniapy.**{*;}
-keep class egame.terminal.feesmslib.jni.**{*;}
#中国联通混淆添加结束

#中国电信混淆添加开始（无需运营商支付方式可忽略以下混淆代码）
-dontwarn cn.play.**
```

```
-dontwarn cn.play.dserv.SdkServ.**
-dontwarn cn.egame.**
-keep class cn.play.**{ *; }
-keep class cn.egame.**{ *; }
#中国电信混淆添加结束
```

附 1.API 类具体使用 demo 工程

请解压《PPSingleSDKDemo.zip》导入 eclpse 查看或运行。