**SDK for Android集成说明**

|  |  |  |
| --- | --- | --- |
| 文件状态：  [] 草稿  [ ] 讨论稿  [√] 正式发布 | 文件标识： | SDK for Android集成说明 |
| 当前版本： | 3.1 |
| 完成日期： | 2017-12-19 |

|  |  |  |
| --- | --- | --- |
| 版本 | 日期 | 版本说明 |
| 1.0.0 | 2017-07-31 |  |
| 2.0.0 | 2017-09-14 |  |
| 3.0.0 | 2017-11-16 |  |
| 3.1.0 | 2017-12-19 | 最新版本 |

**目录**

一 SDK介绍 3

1. 简介 3

二 SDK清单文件 3

1 文件说明 3

2 开发环境配置 4

2.1 添加jar文件 4

2.2 添加res文件 5  
2.3 修改AndroidMainfest.xml文件 5

2.3.1 添加所需用户权限 5

2.3.2 配置appid 6

3 SDK调用流程 7

三 方法调用 7

3.1 必须调用的方法 7

3.1.1 初始化 7

3.1.2 登录登出 事件监听 8

3.1.3 通知游戏监听 9

3.1.4 退出游戏 9

3.2 使用SDK自带的界面 10

3.2.1 登录 10

3.2.2 选择游戏方式 10

3.2.3 创建房间 10

3.2.4 加入房间 10

3.2.5 显示二维码 10

3.2.6 提交游戏结果 10

3.3 游戏有自定义的界面 11

3.3.1 登录 11

3.3.2 获取创建房间所需的支付金额 11

3.3.3 获取加入房间所需的支付金额 12

3.3.4 创建房间 12

3.3.5 扫描二维码 13

3.3.6 获取加入房间的人的信息 13

3.3.7 加入房间 13

3.3.8 通知SDK游戏结束 14

3.3.9 提交游戏结果 14

3.3.10 注销账号 14

3.3.11 退出游戏 14

3.4 选接方法 14

3.4.1 二维码添加额外信息 14

3.4.2 获取二维码添加的额外信息 14

3.4.3 设置二维码显示的内容 14

# 一、SDK介绍

## 1、简介

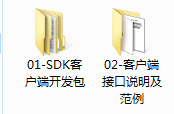
游戏SDK是我们开发的一款专数据统计服务的游戏插件。

不同于一般基于设备的统计，SDK基于游戏内用户游戏账号统计。同一台设备可能同时包含多个游戏账号。基于账号的统计更能精确反应游戏的真实用户情况。

二、SDK清单文件

**1.文件说明**

**解压后文件夹中包括：**



* **01-SDK****客户端开发包Libs文件夹**

android版sdk文件为四个jar文件 都是游戏所需要的jar包

* **02-客户端接口说明及范例**

SDK客户端示例工程，包含文档中所述的示例代码

* **升级说明**

**新版SDK升级内容**

**3、嵌入步骤说明**

嵌入SDK主要分为如下几个步骤：

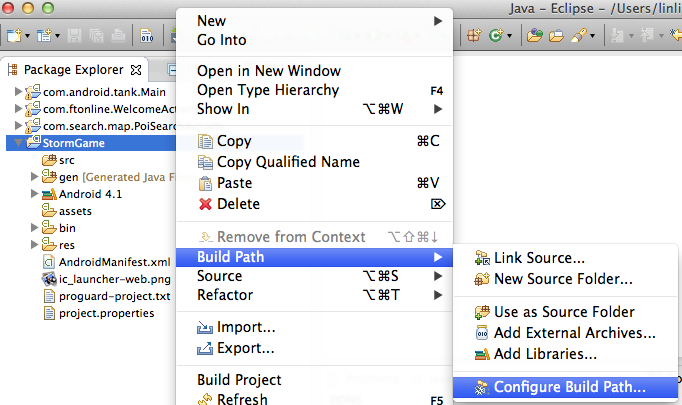
1. 联系商务，获取产品ID和登录验证KEY和支付KEY，每一个游戏对应唯一的ID和KEY。
2. 本地开发环境搭建，针对SDK需要适当调整本地的开发环境，
3. 导入SDK的功能，开发者可参照文档中的示例代码进行功能的对接。

**2.开发环境配置**

## 2.1 添加jar文件

将解压后得到的sdk 客户端开发包里面的libs文件夹，复制到您的工程目录下，然后刷新工程并导入jar文件。具体步骤：

在eclipse中右键您的工程， 选择Build Path->Config Build Path，在弹出的面板中选择Add JARS，将libs文件夹下新添加的jar文件全部添加到工程中。点“ok”导入jar文件。



## 2.2 添加res资源文件

将解压得到的sdk 客户端开发包里面的res文件夹直接拷贝覆盖至您工程中的res文件夹。

## 2.3修改AndroidMainfest.xml文件

在您的游戏工程的AndroidMainfest.xml文件中添加SDK工程中需要的权限及service声明等。

### 2.3.1 添加所需用户权限

**所需用户权限：**

<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />  
<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />  
<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  
<uses-permission android:name="android.permission.Man" />  
<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />  
<uses-permission android:name="android.permission.MOUNT\_UNMOUNT\_FILESYSTEMS" />  
<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />  
<uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW" />  
<uses-permission android:name="android.permission.GET\_TASKS" />  
  
<uses-permission android:name="android.permission.VIBRATE" />  
<uses-permission android:name="android.permission.CAMERA" />  
  
<uses-feature android:name="android.hardware.camera.autofocus" />

**添加所需的activity：**

<activity  
 android:name="com.yyjia.sdk.ScanQRCodeActivity"  
 android:configChanges="orientation|keyboardHidden|navigation"  
 android:theme="@android:style/Theme.Light.NoTitleBar" />  
<activity  
 android:name="com.yyjia.sdk.CaptureActivity"  
 android:configChanges="orientation|keyboardHidden|navigation"  
 android:theme="@android:style/Theme.Light.NoTitleBar" />

**添加所需的service：**

<service android:name="com.yyjia.sdk.util.FloatViewService" />  
<service  
 android:name="com.yyjia.sdk.util.CxAccessService"  
 android:enabled="true"  
 android:exported="true"  
 android:permission="android.permission.BIND\_ACCESSIBILITY\_SERVICE">  
 <intent-filter>  
 <action android:name="android.accessibilityservice.AccessibilityService" />  
 </intent-filter>  
  
 <meta-data  
 android:name="android.accessibilityservice"  
 android:resource="@xml/game\_sdk\_accessible\_service\_config" />  
</service>  
<service android:name="com.yyjia.sdk.util.GameResultService" />

### 2.3.2 配置渠道号和appid

<meta-data

android:name="TOKYOGAME\_APPID"

android:value="20002" />

**参数说明：**

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **默认值** |
| TOKYOGAME\_APPID | 分配给合作游戏产品的APPID，作为唯一标示符(如果是多个游戏集成的，只需配置其中一款游戏的APPID即可) | N/A |

**3.SDK调用流程**

**初始化——>登录——>支付界面——开始游戏——>上报游戏结果——>获取游戏最终结果**

# 三 方法调用

# 3.1 必须调用的方法

## 3.1.1 初始化

**方法：**

mCenter = GMcenter.*getInstance*(mActivity);

mCenter.onCreate(mActivity);

**在游戏的启动类中，游戏初始化时调用此方法，激活SDK；**

**3.1.2登录、登出、关闭登录窗口、SDK初始化 事件监听**

// 登录登出监听器  
mCenter.setLoginListener(new LoginListener() {  
 // 登录监听方法  
 @Override  
 public void loginSuccessed(String code) {  
 if (code == Information.*LOGIN\_SUSECCEDS*) {  
 // SDK登录成功 游戏(服务端)请求 登录验证  
 ToastUtil.*showShortToast*(mActivity, "LOGIN SUCCESS");  
 } else {  
 ToastUtil.*showShortToast*(mActivity, "LOGIN FAIL");  
 }  
 }  
  
 // 登出监听方法  
 public void logoutSuccessed(String code) {  
 if (code == Information.*LOGOUT\_SUSECCED*) {  
 // 账号 退出 游戏需要重启到 登录界面  
 ToastUtil.*showShortToast*(mActivity, "EXIT SUCCESS");  
 } else {  
 ToastUtil.*showShortToast*(mActivity, "EXIT FAIL");  
 }  
 }  
  
 @Override  
 public void logcancelSuccessed(String code) {  
 if (code == Information.*LOGCANCEL\_SUSECCED*) {  
 ToastUtil.*showShortToast*(mActivity, "CANCEL LOGIN");  
 }  
 }  
});

// 初始化监听方法

mCenter.setInitListener(new InitListener() {  
  
 @Override  
 public void initSuccessed(String code) {  
 // 监听SDK初始化 如果初始化成功 可以选择适合时间登陆  
 if (code == Information.*INITSUCCESSEDS*) {  
 mCenter.checkLogin();  
 } else {  
 ToastUtil.*showShortToast*(mActivity, "INIT FAIL");  
 }  
 }  
});

## 3.1.3 通知游戏监听

mCenter.setNoticeGameListener(new NoticeGameListener() {  
  
 // 通知游戏开始  
 @Override  
 public void startGame(int gameId) {  
 ToastUtil.*showShortToast*(mActivity, "Start Game");  
 }  
  
 // 通知游戏结束  
 @Override  
 public void gameOver(int gameId) {  
 ToastUtil.*showShortToast*(mActivity, "Game Over");  
 }  
   
 // 单人模式（试玩）  
 @Override  
 public void startSingleGame(int gameId) {  
  
 }  
});

## 3.1.4 退出游戏

mCenter.exitSDK();

// 退出游戏监听方法

mCenter.setExitSDKListener(new ExitSDKListener() {  
 @Override  
 public void onSuccess() {  
 mCenter = null;  
 System.*exit*(0);  
 }  
  
 @Override  
 public void onFailure() {  
 ToastUtil.*showShortToast*(mActivity, "Exit Failure");  
 }  
});

**3.2 使用SDK自带的界面（游戏视自身情况接入）**

**3.2.1登录**

mCenter.checkLogin();

**3.2.2 选择游戏方式(一般在只用一个按钮的时候需要，由游戏自身决定需不需要调用)**

mCenter.selectGameWay();

**3.2.3 创建房间(显示SDK的支付界面)**

mCenter.createRoom(Context context);

// 不设置创建房间的监听setCreateRoomListener

**3.2.4 加入房间(显示SDK的支付界面)**

mCenter.joinRoom(Context context, String roomId);

// 不设置加入房间的监听setJoinRoomUserListListener

**3.2.5 显示二维码**

mCenter.showQrCode();

// 不设置这两个监听 setJoinRoomUserListListener setScanQrCodeListener

**3.2.6 提交游戏结果**

*/\*\*  
 \* 提交游戏结果  
 \*  
 \** ***@param*** *score 游戏结果（分数）  
 \** ***@param*** *isCustomSubmitResultInterface 是否使用SDK自带的界面 true 不使用SDK自带的界面(包括显示游戏分数界面和显示游戏最终结果的界面)  
 \*/*

mCenter.submitGameResult(int Score, false);

// 如果需要显示游戏最终结果的界面，不需要设置游戏结果的监听 setSubmitGameResultListener

# 3.3 游戏有自己自定义的界面（**游戏视自身情况接入**）

**3.3.1 登录**

// 是否自定义登录界面(有则传入true，否则传入false)  
mCenter.setCustomLoginInterface(true);

*/\*\*  
 \* 登录（使用游戏自己的登录界面）  
 \*  
 \** ***@param*** *userName 账号  
 \** ***@param*** *password 密码  
 \*/*

mCenter.login(String userName, String password);

**3.3.2 获游创建房间所需的支付金额**

*/\*\*  
 \* 获取创建房间所需支付的金额  
 \*  
 \** ***@param*** *context  
 \** ***@param*** *gameId 游戏ID  
 \*/*

mCenter.getCreateRoomCost(Context context, int gameId);

// 获取创建房间所需支付的金额  
mCenter.setCreateRoomCostListener(new CreateRoomCostListener() {  
 @Override  
 public void onSuccess(int gameId, String money) {  
 Utils.*E*("money = " + money);  
 }  
  
 @Override  
 public void onFailure(int gameId, String errMsg) {  
  
 }  
});

**3.3.3 获游加入房间所需支付的金额**

*/\*\*  
 \* 获取加入房间所需支付的金额  
 \*  
 \** ***@param*** *context  
 \** ***@param*** *gameId 游戏ID  
 \*/*

mCenter.getJoinRoomCost(Context context, int gameId);

// 加入房间所需支付金额的监听  
mCenter.setJoinRoomCostListener(new JoinRoomCostListener() {  
 @Override  
 public void onSuccess(int gameId, String money) {  
  
 }  
  
 @Override  
 public void onFailure(int gameId, String errMsg) {  
  
 }  
});

**3.3.4 创建房间**

mCenter.createRoom(mActivity);

/// 创建房间监听  
mCenter.setCreateRoomListener(new CreateRoomListener() {  
 @Override  
 public void onSuccess(int gameId, String result) {  
 ToastUtil.*showShortToast*(mActivity, "Create Room Success");  
 }  
  
 @Override  
 public void onFailure(int gameId, String errMsg) {  
 ToastUtil.*showShortToast*(mActivity, errMsg);  
 }  
});

**3.3.5 扫描二维码**

mCenter.scanQrCode(mActivity);

mCenter.setScanQrCodeListener(new ScanQrCodeListener() {  
 @Override  
 public void showResult(String result) {  
 Utils.*E*("result = " + result);  
 // 返回二维码扫描数据，需游戏自己解析数据  
 mCenter.joinRoom(mActivity, result);  
 }  
});

**3.3.6 获取加入房间的人的信息（如果游戏不需要获取加入房间的人的信息，也需要接入此方法）**

mCenter.setJoinRoomUserListListener(new JoinRoomUserList() {  
 @Override  
 public void result(String result) {  
 Utils.*E*("setJoinRoomUserListListener result =" + result);  
 }  
});

**3.3.7 加入房间**

// 第一个参数：游戏ID 第二个参数：房间ID

mCenter.joinRoom(mActivity, gameId, roomId);

// 加入房间的监听  
mCenter.setJoinRoomListener(new JoinRoomListener() {  
 @Override  
 public void onSuccess(int gameId, String result) {  
 Utils.*E*("JoinRoomListener onSuccess = " + result);  
 ToastUtil.*showShortToast*(mActivity, "Join Room Success");  
 }  
  
 @Override  
 public void onFailure(int gameId, String errMsg) {  
 ToastUtil.*showShortToast*(mActivity, errMsg);  
 }  
});

**3.3.8 通知SDK游戏开始（如果游戏自定义了加入房间的界面或者创建房间的界面）**

mCenter.noticeSdkStartGame();

**3.3.9 提交游戏结果**

*/\*\*  
 \* 提交游戏结果  
 \*  
 \** ***@param*** *score 游戏结果（分数）  
 \** ***@param*** *isCustomSubmitResultInterface 是否使用SDK自带的界面 true 不使用SDK自带的提交结果界面  
 \*/*

mCenter.submitGameResult(int Score, true);

// 游戏需要实现自己最终显示游戏结果的界面，需要实现此方法

mCenter.setSubmitGameResultListener(new SubmitGameResultListener() {  
 @Override  
 public void showGameResult(String result) {  
 Utils.*E*("showGameResult = " + result);  
 }  
});

**3.3.10 注销账号**

mCenter.logout();

**3.3.11 退出游戏（App由多个游戏组成，要切换其它游戏需调用这个方法）**

mCenter.exitGame();

**3.4 选接方法**

**3.4.1 二维码添加额外信息**

mCenter.setQrCodeExtraInfo("Extra Info");

**3.4.2 获取二维码添加的额外信息（扫描二维码成功之后调用，实际调用时机，游戏视自身情况调用）**

**3.4.3 设置二维码显示的内容（如果调用了此方法，需设置扫描二维码的监听，即调用setScanQrodeListener()）**

mCenter.setQrCodeContent("QrCode Content");