U9 SDK 接入文档

版本内容详解			
版本号	更新时间	更新内容	更新人员
v1.0	2015-12-10	初始制作	smile
V1.0	2015-2-22	添加注意事项	Daniel

1、概述

此文档为 Android 接入文档

注意事项:

1、请不要使用 Toast 消息，使用自定义提示消息

2、渠道闪屏预留接口是后续提供使用渠道闪屏，游戏跟发行闪屏 Log 请使用自己的闪屏播放

3、游戏的退出方式请不要使用 ”按两次退出游戏”，请使用提示退出框，退出游戏

4、游戏登录按钮、支付 做按钮限制 避免造成多次调用 (请不要根据登录回调来做判断,不一定所有渠道每一个操作都一定会有回调)

5、游戏内收到注销账号成功后，请回到 “游戏登录场景” 并注销玩家的登录状态 与 登录信息(避免再次点击登录直接进入游戏)

6、注意 路径名 请不要出现 ”com.hy.game”

7、注意 路径名 请不要出现 与 AndroidManifest.xml 中 package 一致

8、数据渠道区分

方法 1:

——— 不区分登录 , 只区分支付 ———

玩家支付完成 , U9 服务器通 知 游戏服务器 时

会传入 玩家的

UserId(用户 Id) ChannelUserId(渠道用户 Id)

ChannelId(渠道 Id) 即 客户端的 HY_CHANNEL_CODE

方法 2 :

——— 区分登录 , 区分支付 ———

通过 AndroidManifest.xml 中 meta-data 信息

HY_GAME_ID 游戏号

HY_CHANNEL_CODE 渠道 code

HY_CHANNEL_TYPE 渠道类型 来区分渠道

通过获取到的渠道 Code 与 Type 区分数据

例如 :

```
<meta-data
          android:name="HY_GAME_ID"
          android:value="1000" />
      <meta-data
          android:name="HY_CHANNEL_CODE"
          android:value="101" />
      <meta-data
          android:name="HY_CHANNEL_TYPE"
          android:value="DangLe" />
```

获取方法 :

```
//获取游戏号
HY_Utils.getHYGameId(mActivity);
//获取渠道号
HY_Utils.getHYChannelCode(mActivity);
//渠道类型(渠道名-英文或拼音)
HY_Utils.getHYChannelType(mActivity);
```

9、游戏分渠道更新

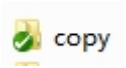
—— 目前 SDK 未提供强更下载功能 ——

如果更新机制 是 强更，需要游戏 来 区分渠道 更新游戏包

如何区分渠道，参照上条

2、环境搭建

2.1、资源导入



将 copy 文件夹下面的文件 拷贝至 工程中

2.2、配置 AndroidManifest.xml 文件

在<application>标签内加入如下权限声明：

```
<uses-permission android:name="android.permission.INTERNET" />
```

在<application>标签内加入如下 Activity 声明：

```
<activity
        android:name="com.hy.game.Game_SplashActivity"
        android:configChanges="orientation|keyboardHidden|screenSize" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

<activity
    android:name="com.hy.game.demo.HyGameDemo"
    android:configChanges="orientation|keyboardHidden|screenSize" />

<activity
    android:name="com.hy.game.demo.FloatActivity"
    android:configChanges="orientation|keyboardHidden|screenSize" />
```

注意：

注意添加闪屏 Activity ,其路径可自由更改，但不要与游戏包名一致

添加 meta-data 参数：

```
        <!-- 游戏号必填 -->
        <meta-data
            android:name="HY_GAME_ID"
            android:value="1000" />
        <!-- 渠道号 打包时自动写入  可以不填 -->
        <meta-data
            android:name="HY_CHANNEL_CODE"
            android:value="100" />
        <!-- 渠道标识 打包时自动写入  可以不填 -->
        <meta-data
            android:name="HY_CHANNEL_TYPE"
            android:value="Test" />
```

3、接口接入

3.1、渠道闪屏预留接口（必接接口）

接口范例：

```
public class Game_SplashActivity extends HY_SplashActivity
{
    private String splash_color = "";
    @Override
    public int getBackgroundColor()
    {
    // 当闪屏 PNG 图片无法铺满部分机型的屏幕时，设置与闪屏颜色配合的背景色会给用户更好的体验，设置背景颜色(这里预留接口，如果需要使用，在打包后台提供)
        splash_color = HY_Utils.getManifestMeta(this,"HY_GAME_COLOR");
        if(!TextUtils.isEmpty(splash_color)){
            return Color.parseColor(splash_color);
        }else{
            //默认背景颜色-白色
             return Color.WHITE;
        }
    }
    @Override
    public void onSplashStop()
    {
        Intent intent = new Intent(this, MainActivity.class);
        startActivity(intent);
        this.finish();
    }
}
```

接口说明：

接入该接口后，开发者可以在打包后台灵活配置闪屏内容，不配置闪屏图片则不出现闪屏,该接口不会影响程序原有闪屏。(请作为第一启动 Activity)

3.2、生命周期接口（必接接口）

```
@Override
    public void onStop()
    {
        super.onStop();
        HY_GameProxy.getInstance().onStop(this);
    }
    @Override
    public void onDestroy()
    {
        super.onDestroy();
        HY_GameProxy.getInstance().onDestroy(this);
    }
    @Override
    public void onResume()
    {
        super.onResume();
        HY_GameProxy.getInstance().onResume(this);
    }
    @Override
    public void onPause()
    {
        super.onPause();
        HY_GameProxy.getInstance().onPause(this);
```

```
    }
    @Override
    public void onRestart()
    {
        super.onRestart();
        HY_GameProxy.getInstance().onRestart(this);
    }
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent
data)
    {
        super.onActivityResult(requestCode, resultCode, data);
        HY_GameProxy.getInstance().onActivityResult(this,
     requestCode,resultCode, data);
    }
```

3.3、初始化接口（必接接口）

参数说明：

参数名	参数类型	说明
mIsLandscape	Boolean	横竖屏控制 true:横屏　false:竖屏

接口范例：

```
// 初始化 app
HY_GameProxy.getInstance().applicationInit(this,mIsLandscape);
//设置 onCreate()
HY_GameProxy.getInstance().onCreate(this);
//设置用户监听(具体详细请看如下 3.4 )
HY_GameProxy.getInstance().setUserListener(this,HY_UserListener())
```

接口说明：

初始化参数 与 传入游戏横竖屏(便于部分渠道设置 SDK 横竖屏)

3.4、设置用户监听（必接接口）

参数说明：

HY_User		
参数名	参数类型	说明
userId	String	用户 id
channelUserId	String	渠道用户 id
channelUserName	String	渠道用户名
Token	String	登录验证令牌

接口范例：

```
//设置用户监听器(切换账号/注销账号)
HY_GameProxy.getInstance().setUserListener(Activity,new HY_UserListener() {
                @Override
                public void onSwitchUser(HY_User user,int resultCode){
                    switch (resultCode) {
                    case HY_SdkResult.SUCCESS:
                        mUser = user;
                        //切换账号成功,回到游戏登录场景,并重置角色信息
                       //如果切换同一角色,请不做任何处理
                        String Token = mUser.getToken();
                        String userId = mUser.getUserId();
                        String channelUserId = mUser.getChannelUserId();
                        String channelUserName = mUser.getChannelUserName();
                        Toast.makeText(MainActivity.this, "切换账号成功",
             Toast.LENGTH_SHORT).show();
                        break;
                    case HY_SdkResult.CANCEL:
                        //切换账号取消不做任何处理
                        Toast.makeText(MainActivity.this, "切换账号取消",
           Toast.LENGTH_SHORT).show();
                        break;
                    default:
                        //切换账号失败不做任何处理
                        Toast.makeText(MainActivity.this, "切换账号失败",
            Toast.LENGTH_SHORT).show();
                        break;
                    }
                }
                @Override
```

```
        public void onLogout(int resultCode,Object paramObject) {
             switch (resultCode) {
             case HY_SdkResult.SUCCESS  :
                    mUser = null;
                    Toast.makeText(MainActivity.this, "注销成功",
      Toast.LENGTH_LONG).show();
                    break;
                default:
                    Toast.makeText(MainActivity.this, "注销失败",
      Toast.LENGTH_LONG).show();
                    break;
             }
          }
      });
```

接口说明：

因部分渠道，有悬浮小球，可实现切换账号、注销账号，该部分功能必须接入

注意：并非强制要求游戏在游戏内添加按钮，只需根据回调进行相对应处理

3.5、登录接口（必接接口）

参数说明：

HY_User		
参数名	参数类型	说明
userId	String	用户 id
channelUserId	String	渠道用户 id
channelUserName	String	渠道用户名
Token	String	登录验证令牌

接口范例：

```
HY_GameProxy.getInstance().logon(MainActivity.this, new HY_LoginCallBack() {
            @Override
               public void onLoginSuccess(Boolean isLogin,HY_User user)
               {
                   if(isLogin){
                    //登录
                       mUser = user;
                        String Token = mUser.getToken();
                        String userId = mUser.getUserId();
                        String channelUserId = mUser.getChannelUserId();
                        String channelUserName = mUser.getChannelUserName();
                        Toast.makeText(MainActivity.this,”登录成功:
                        Token:”+Token+”,userId:”+userId+”,渠道用户
                        id:”+channelUserId+”,渠道用户名:”+channelUserName,
                        Toast.LENGTH_LONG).show();
```

```
                    }else{//切换账号
                }
            }
            @Override
            public void onLoginFailed(int code, String message)
        {
                Toast.makeText(MainActivity.this,”登录失败：code:
          ”+code+”,message:”+message,Toast.LENGTH_LONG).show();
            }
    });
```

接口说明：

登录成功后，请通知游戏服务器 与 U9 服务器进行登录验证。(具体详情请看服务端接入文档)

3.6、注销接口

若游戏内无 【注销账号】 按钮，可以不接入

接口范例：

```
HY_GameProxy.getInstance().logoff(MainActivity.this, "注销");
```

接口说明：

账号注销成功，请清除玩家登录信息，并注销当前玩家登录状态，并回到游戏登录场景
账号注销失败，不做任何操作(或提示玩家注销失败)

3.7、支付接口（必接接口）

参数说明：

HY_PayParams:支付请求参数			
参数名	参数类型	说明	是否可为空
amount	Int	支付金额/单位：分	NOT_NULL
productId	String	商品 id	NULL
productName	String	商品名称 例如:钻石	NOT_NULL
exchange	String	充值比例 例如: 1:10 = 10	NOT_NULL
gameOrderId	String	游戏订单号	NOT_NULL
callBackUrl	String	支付回传地址	NOT_NULL
appExtInfo	String	拓展信息，支付回传参数	NULL

支付请求返回信息		
参数名	参数类型	说明
retCode	int	返回码 0:成功 1:失败
message	String	返回信息

接口范例：

```
HY_PayParams payParams = new HY_PayParams();
        payParams.setAmount(100);
        payParams.setProductId("");//如果没有传空
        payParams.setProductName("钻石");
        payParams.setCallBackUrl("http://www.baidu.com");
        payParams.setGameOrderId( "game"+time);
        payParams.setAppExtInfo("支付回调拓展字段");
        HY_GameProxy.getInstance().startPay(MainActivity.this, payParams, new
HY_PayCallBack()
        {
            @Override
            public void onPayCallback(int retCode, String message)
            {
                //retCode 状态码： 0 支付成功， 1 支付失败， 2 支付取消， 3 支付
进行中。
               switch (retCode) {
             case 0:
                  Toast.makeText(MainActivity.this, "支付成功",
           Toast.LENGTH_LONG).show();
                 break;
             case 2:
                  Toast.makeText(MainActivity.this, "支付取消",
```

```
            Toast.LENGTH_LONG).show();
                 break;
             case 3:
                 Toast.makeText(MainActivity.this, "支付进行中",
            Toast.LENGTH_LONG).show();
                 break;
             default:
                 Toast.makeText(MainActivity.this, "支付失败",
            Toast.LENGTH_LONG).show();
                 break;
               }
            }
        });
```

接口说明：

注意：客户端的回调仅代表玩家操作，可以用作提示玩家，支付以结果请以服务端收到通知为准

3.8、角色信息提交接口（必接接口）

HY_GameRoleInfo :角色信息提交参数			
参数名	参数类型	说明	是否可为空
typeId	Int	上传类型： 0:登录服务器 1:创角 2:升级	NOT_NULL
roleId	String	角色 id:如果没有请传 userid	NOT_NULL
roleName	String	角色名	NOT_NULL
roleLevel	String	角色等级:若无请传 1	NOT_NULL
zoneId	String	区服 id:若无请传 1	NOT_NULL
zoneName	String	区服名:若无，游戏名+1 区	NOT_NULL
balance	Int	道具余额:例如剩余 300 钻石 传值:300,若无请传 0	NOT_NULL
vip	String	Vip 等级:若无请传 0	NOT_NULL
partyName	String	帮派名称:若无请传”无帮派”	NOT_NULL

接口范例：

```
HY_GameRoleInfo gameRoleInfo =new HY_GameRoleInfo();
    gameRoleInfo.setTypeId(HY_Constants.ENTER_SERVER);
    gameRoleInfo.setRoleId("1235698465");
    gameRoleInfo.setRoleName("欧阳锋");
    gameRoleInfo.setRoleLevel("11");
    gameRoleInfo.setZoneId("1358");
    gameRoleInfo.setZoneName("狂战一区");
    gameRoleInfo.setBalance(10);
    gameRoleInfo.setVip("3");
    gameRoleInfo.setPartyName("丐帮");
 HY_GameProxy.getInstance().setRoleData(this,gameRoleInfo);
```

接口说明：

注意： 角色信息不可为空，如果没有，请对应传 0 或 无

3.9、退出接口（必接接口）

接口范例：

```
HY_GameProxy.getInstance().exit(this, new HY_ExitCallback()
        {
            @Override
            public void onGameExit()
            {
               //如果渠道无提供退出窗口
                 退出确认窗口需要游戏自定义，并且实现资源回收，杀死进程等退出逻辑
                AlertDialog.Builder builder = new Builder(MainActivity.this);
                builder.setTitle("游戏退出界面");
                builder.setPositiveButton("退出",
                        new DialogInterface.OnClickListener()
                        {
                            @Override
                            public void onClick(DialogInterface dialog,
                                    int which)
                            { //销毁接口,必接
                                HY_GameProxy.getInstance().applicationDestroy(
                                        MainActivity.this);
                                MainActivity.this.finish();
                            }
                        });
                builder.show();
            }
            @Override
            public void onChannelExit()
            {
```

```
                // 渠道存在退出界面
                Toast.makeText(MainActivity.this, "渠道退出",
                        Toast.LENGTH_LONG).show();
                 //销毁接口,必接
                HY_GameProxy.getInstance()
                        .applicationDestroy(MainActivity.this);
                 MainActivity.this.finish();
            }
        });
```

接口说明：

onGameExit() 游戏退出接口
中请调用游戏退出界面,执行游戏退出销毁，游戏退出销毁之前请执行 销毁接口。
onChannelExit() 渠道退出接口
无需调起游戏退出界面，执行游戏退出销毁，游戏销毁退出之前请执行 销毁接口。

3.10、销毁接口（必接接口）

接口范例：

```
HY_GameProxy.getInstance().applicationDestroy(MainActivity.this);
```

接口说明：

请在游戏游戏退出时，销毁之前调用次接口