

分发 sdk 安卓接入说明

分发 sdk 安卓接入说明

撰写人	修订日期	版本
溪谷科技	2021-9-1	V1.1.3
邱伟	2022-11-14	V1.1.8
邱伟	2022-11-23	V1.1.9
邱伟	2022-11-26	V1.2.0
邱伟	2022-12-05	v1.2.1
邱伟	2023-03-05	V1.2.2
邱伟	2023-03-21	V1.2.3

更新内容：

1: 新增 GM 中心入口

一、导入 sdk 文件

1. 复制 aar 文件到工程 libs 目录内

在项目 build.gradle 里引入分发 sdk

```
//引入分发 sdk 包
implementation(name:'youweiw_l_core_v1.0.15', ext:'aar')
implementation(name:'youweiw_l_gm_sdk_v1.2.3', ext:'aar')
```

eclipse 环境，解压 aar 文件，取出 jar，导入项目

2. 配置 AndroidManifest.xml

Plain Text

```
<meta-data android:name="YWKJ_GAME_CHANNEL_ID " android:value="ywkj_0"/> #
此值固定，不需要更改
<meta-data android:name="YWKJ_GAME_SCREEN_ORIENTATION"
android:value="1"/> #游戏屏幕方向 1 横屏 2 竖屏

<meta-data android:name="YWKJ_DEMAIN_URL" android:value="
https://gamemeta.net.cn"/>
```

3. 配置 assets/ywgmjhconfig.ini 文件

```
{
    "gameid": "具体参数商务给到",
    "gamename": "具体参数商务给到",
    "gameappid": "具体参数商务给到",
    "access_key": "具体参数商务给到",
    "gameurl": "具体参数商务给到"
}
```

二、生命周期方法接入介绍

1. 接入游戏 Application 生命周期方法

具体使用请参照 demo 演示工程 com.youweiwlgm.distribute.**BaseApplication** 内代码示例

生命周期	是否必接	方法
onCreate()	是	YWWLApi.getInstance().initApplication(context);
attachBaseContext()	是	YWWLApi.getInstance().attachBaseContext(this,context);
onTerminate()	是	YWWLApi.getInstance().onTerminate();
onConfigurationChanged()	是	YWWLApi.getInstance().onConfigurationChanged(this,newConfig);

2. 接入游戏 Activity 生命周期方法

可参照 demo 演示工程 com.youweiwlgm.distribute.**MainActivity** 内代码示例

生命周期	是否必接	方法
onResume()	是	YWWLApi.getInstance().onResume(activity);
onPause()	是	YWWLApi.getInstance().onPause(activity);
onStop()	是	YWWLApi.getInstance().onStop(activity);
onDestroy()	是	YWWLApi.getInstance().onDestroy(activity);
onStart()	是	YWWLApi.getInstance().onStart(activity);
onRestart()	是	YWWLApi.getInstance().onRestart(activity);
onNewIntent()	是	YWWLApi.getInstance().handleIntent(activity,intent);
onActivityResult()	是	YWWLApi.getInstance().onActivityResult(activity, requestCode, resultCode, data);
onRequestPermissionsResult()	是	YWWLApi.getInstance().onRequestPermissionsResult(this,requestCode,permissions,grantResults);
onBackPressed()	是	YWWLApi.getInstance().onBackPressed(this);
onConfigurationChanged()	是	YWWLApi.getInstance().onConfigurationChanged(this, newConfig);

三、主要功能接入

具体使用请参照 demo 演示工程 com.youweiwlgm.distribute.**MainActivity** 内代码示例
(注意：要先设置结果监听，再调用功能方法)

1. 设置隐私授权结果监听、调用隐私授权

```
Kotlin
//设置 sdk 初始化结果监听
//隐私协议授权 回调
YWWLApi.getInstance().setSdkPrivacyAgreementListener(new ISdkPrivacyAgreementListener() {
    @Override
    public void privacyAgreementResult(int result) {
        if (result == SDKContants.RESULT_SUCCESS) {
            Log.w(TAG, "已经授权隐私协议，可以进行权限申请和初始化!");
            YWWLApi.getInstance().init(MainActivity.this, true);
        }
    }
});
//PS:必须要在授权成功后再调用初始化方法
YWWLApi.getInstance().privacyAuthorization(this);
```

2. 设置初始化结果监听、初始化 sdk

```
Kotlin
//设置 sdk 初始化结果监听
YWWLApi.getInstance().setSdkInitListener(new ISdkInitListener() {
    @Override
    public void initResult(SdkInitResult initResult) {
        if (initResult.getErrorCode() == SDKContants.RESULT_SUCCESS) {
            btnlogin.setVisibility(View.VISIBLE);
            Toast.makeText(MainActivity.this, "初始化成功", Toast.LENGTH_SHORT).show();

        } else {
            Toast.makeText(MainActivity.this, "初始化失败: " + initResult.getErrorMesage(),
Toast.LENGTH_SHORT).show();
        }
    }
});

//初始化 sdk（建议写在游戏 Activity 页面的 onCreate()方法中）
// true 开启日志，建议开发时开启，上线时关闭
YWWLApi.getInstance().init(MainActivity.this, true);
```

2. 设置用户登录结果监听、发起登录

登录成功时游戏服务器需要拿 **userId** 和 **token** 访问 **sdk** 服务器，进行登录验证。详见《分发 **sdk** 服务端接口文档》

```
Plain Text
//设置 sdk 用户登录结果监听
YWWLApi.getInstance().setSdkLoginListener(new ISdkLoginListener() {
    @Override
    public void loginResult(SdkLoginResult loginResult) {
        if (loginResult.getErrorCode() == SDKContants.RESULT_SUCCESS) {
            String userId = loginResult.getUserId(); //user_id(用户唯一标识)
            String token = loginResult.getToken(); //user_token
            Boolean userCertification = loginResult.isUserCertification(); //用户实名状态，
true: 已实名认证 false:未实名认证
            String userBirthday = loginResult.getUserBirthday(); //用户生日信息
            (例:"19950712")
            Log.w(TAG, "登录成功: " + "userId:" + userId + ", token:" + token);
            if (userCertification){
                Log.w(TAG, "该用户已经实名认证，生日为: " + userBirthday);
            }
        } else {
            Toast.makeText(MainActivity.this, "登录失败: " + loginResult.getErrorMesage(),
Toast.LENGTH_SHORT).show();
        }
    }
});
```

```
});  
  
//发起 sdk 登录  
YWWLApi.getInstance().login(MainActivity.this);
```

3. 设置支付结果监听、设置游戏道具参数发起支付

支付结果应以 sdk 服务端通知为准，服务端通知规则详见《分发 sdk 服务端接口文档》

```
Plain Text  
//设置 sdk 支付结果监听  
YWWLApi.getInstance().setSdkPayListener(new ISdkPayListener() {  
    @Override  
    public void payResult(SdkPayResult payResult) {  
        if (payResult.getErrorCode() == SDKConstants.RESULT_SUCCESS) {  
            Toast.makeText(MainActivity.this, "支付成功", Toast.LENGTH_SHORT).show();  
        } else {  
            Toast.makeText(MainActivity.this, "支付失败: " + payResult.getErrorMessage(),  
                Toast.LENGTH_SHORT).show();  
        }  
    }  
});
```

设置道具信息发起 sdk 支付,所有参数必填，游戏没有的字段传一个默认值,否则可能出现错误。**String 类型默认值为: "default",int 类型默认值为: 0。**

```
GamePropsInfo gamePropsInfo = new GamePropsInfo();  
gamePropsInfo.setPropsId(propsId);//渠道道具 id  
gamePropsInfo.setPropsName("测试道具");//道具名称  
gamePropsInfo.setPropsDesc("这是道具描述");//道具描述  
gamePropsInfo.setPropsPrice(price);//道具价格（单位：分）  
gamePropsInfo.setPropsNumber(1);//购买数量  
gamePropsInfo.setRoleId("1");//游戏角色 id  
gamePropsInfo.setRoleName("测试角色");//游戏角色名  
gamePropsInfo.setRoleLevel("1");//游戏角色等级  
gamePropsInfo.setServerId("1");//游戏区服 id  
gamePropsInfo.setServerName("测试 1 区");//游戏区服名  
gamePropsInfo.setGameOrderId(System.currentTimeMillis()+"");//游戏订单号（每次支付  
订单号不可重复）  
gamePropsInfo.setExtend("测试透传数据");//游戏透传信息  
gamePropsInfo.setCount(1);//玩家获取的游戏币/道具数量  
gamePropsInfo.setRoleVipLevel("1");//玩家 VIP 等级  
gamePropsInfo.setPartyName("荣耀圣殿");//工会 帮会名称  
gamePropsInfo.setRoleBalance("5000");//玩家战力  
gamePropsInfo.setPayCallbackUrl("https://sytp5.ceshi.xghy.com/pay/callback");//支付通知  
地址，测试使用，实际以平台设置为准  
YWWLApi.getInstance().pay(MainActivity.this, gamePropsInfo);
```

4. 设置上传游戏角色信息结果监听、上传游戏角色信息

```
Plain Text
//设置 sdk 上传角色结果监听
YWWLApi.getInstance().setSdkRoleListener(new ISdkRoleListener() {
    @Override
    public void submitResult(int result) {
        if (result == SDKConstants.RESULT_SUCCESS){
            Toast.makeText(MainActivity.this, "上传角色信息成功",
            Toast.LENGTH_SHORT).show();
        }else {
            Toast.makeText(MainActivity.this, "上传角色信息失败",
            Toast.LENGTH_SHORT).show();
        }
    }
});
```

设置游戏角色信息,上传游戏角色,所有参数必填,游戏没有的字段传一个默认值,否则可能出现错误。**String 类型默认值为: "default",int 类型默认值为: 0。**

```
Plain Text
RolesInfo rolesInfo = new RolesInfo();
rolesInfo.setRoleId("1");//角色 Id
rolesInfo.setRoleName("测试角色");//角色名
rolesInfo.setLeval("1");//角色等级
rolesInfo.setBalance("1");//角色余额
rolesInfo.setRoleCombat("9999");//角色战力值
rolesInfo.setVipLevel("1");//角色 vip 等级
rolesInfo.setZoneId("1");//区服 Id
rolesInfo.setZoneName("测试 1 区");//区服名
rolesInfo.setGuildName("测试公会");//工会/帮会名
rolesInfo.setRoleGender("男");//角色性别
rolesInfo.setPartyId("123");//工会/帮会 id
rolesInfo.setPartyRoleName("帮主");//角色在帮派中的名称
rolesInfo.setPartyRoleId("1");//角色在帮派中的 id
rolesInfo.setProfessionId("1");//职业 id
rolesInfo.setProfession("法师");//职业
rolesInfo.setFriendList("");//设置好友关系列表
rolesInfo.setRoleCreateTime("1101201511");//角色创建时间 值为 10 位数时间戳
rolesInfo.setSubmitType("1");//上传场景类型: 1:创建角色 2:角色登录 3:角色升级 4:角色退出 0:其他场景
YWWLApi.getInstance().submitRoleInfo(MainActivity.this, rolesInfo);
```

5. 设置注销登录结果回调、注销登录

```
Plain Text
//设置 sdk 注销登录结果监听
YWWLApi.getInstance().setSdkLogoutListener(new ISdkLogoutListener() {
```

```

@Override
public void logoutResult(int logoutResult) {
    if (logoutResult == SDKContants.RESULT_SUCCESS){
        Toast.makeText(MainActivity.this, "注销登录成功",
Toast.LENGTH_SHORT).show();
    }else {
        Toast.makeText(MainActivity.this, "注销登录失败",
Toast.LENGTH_SHORT).show();
    }
}
});

//注销登录
YWWLApi.getInstance().logout(MainActivity.this);

```

6. 设置退出程序结果回调、退出游戏程序

```

Plain Text
//退出游戏程序结果监听
YWWLApi.getInstance().setSdkExitListener(new ISdkExitListener() {
    @Override
    public void exitResult(int SdkExitresult) {
        if (SdkExitresult == SDKContants.RESULT_SUCCESS){
            Log.w(TAG,"玩家执行了退出程序操作");
            //这里设置游戏自己的关闭进程、退出程序等操作
            Process.killProcess(Process.myPid());
        }
    }
});

//调用 sdk 确认退出程序弹窗
YWWLApi.getInstance().exit(MainActivity.this);

```

7. 设置实名认证和查询实名信息结果回调、查询实名结果

```

//设置实名认证结果回调
YWWLApi.getInstance().setRealnameListener(new ISdkRealNameListener() {
    @Override
    public void submitResult(RealNameInfo realNameInfo) {
        //0 已成年 1 未成年 2 等待中
        int status = realNameInfo.stauts;
        //生日
        String birthday = realNameInfo.birthday;
        //身份证
        String idCard = realNameInfo.idcard;
        //姓名
        String realName = realNameInfo.realname;
    }
});

```

```
}  
});  
YWWLApi.getInstance().queryRealname(MainActivity.this)
```

四、注意事项（必读）

如果游戏工程是 **AndrodX** 项目，为防止分发打包出错，请**禁止依赖包迁移至 AndroidX**
gradle.properties 中添加设置：

```
JavaScript  
#依赖包是否迁移至 AndroidX  
android.enableJetifier=false
```