QuickGame SDK Android 版

接入文档

产品版本：v5.6.22

1.资源导入

前期准备：

由于 sdk 内部使用了 oaid-2.1.0 版本，所以 cp 需要提前到移动安全联盟申请账号以及参数文件；
步骤：

1. 到移动安全联盟官网申请账号并审核 http://www.msa-alliance.cn/

2. 下载 SDK 找到里面的 csv 文件 填写游戏包名等信息 然后发送邮件到 msa@caict.ac.cn

3. 得到包名.cert.pem 文件 并且复制到游戏工程 assets 目录下

4.


3、将证书文件（应用包名.cert.pem）拷贝到项目 assets 目录下。

证书需要填写 example_batch.csv 后发送到 msa@caict.ac.cn 申请，注意每个包名对应一个签名，申请时需要将所需申请的全部包名填写到表格中。

证书有效期一年，为避免证书过期影响 APP 获取 ID，建议证书信息可以从后台服务器获取，或者当调用 oaid SDK 接口提示证书无效时，调用后台接口及时更新证书信息，并且快到期时及时提前重新申请证书。

4、设置依赖

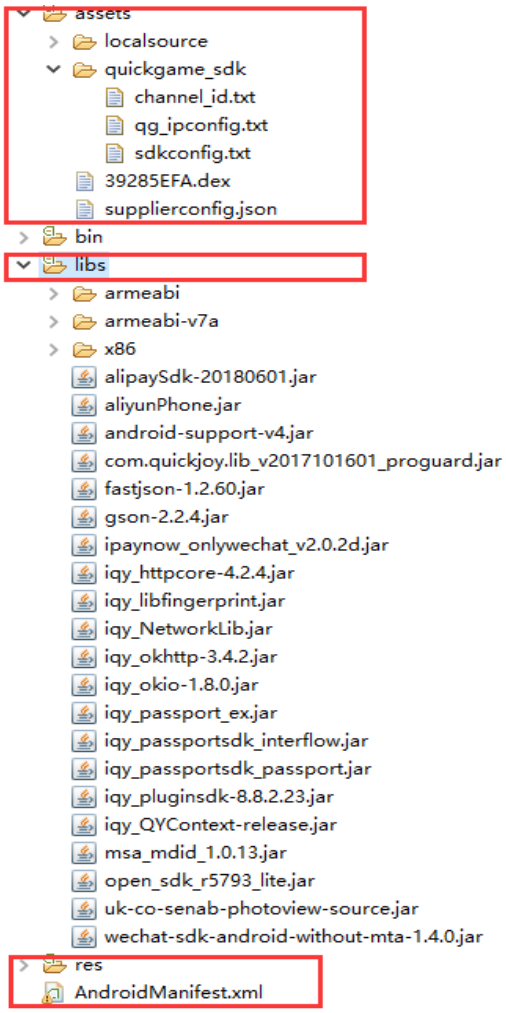
implementation files('libs/oaid_sdk_x.x.x.aar')

5、设置 gradle 编译选项，开发者可以根据自己对平台的选择进行合理配置

ndk {

依次复制以下资源到您的项目中：

***包含 assets libs res manifest.xml 里面的资源**



资源复制进去后，切记修改 manifest 里面部分包名占位符，以及预设授权登录方式的参数，如不使用授权登录，则保留默认参数 QK0

在项目或 module 的 build.gradle 添加以下依赖（版本只能高不能低）：

```
implementation fileTree(dir: 'libs', include: ['*.jar','*.aar'])
implementation 'org.jetbrains.kotlin:kotlin-stdlib-jdk7:1.3.61'
implementation 'androidx.appcompat:appcompat:1.2.0'

//taptap
implementation 'cn.leancloud:storage-android:8.2.18'
```

```
implementation 'cn.leancloud:realtime-android:8.2.18'
implementation 'io.reactivex.rxjava2:rxandroid:2.1.1'
//api project(path: ':oaid_1_0_25')
api 'com.tencent.vasdolly:helper:3.0.4'
```

2.接入描述

1. 调用隐私协议接口

```
QGManager.showPrivace(MainActivity.this, new QGCallBack() {
    @Override
    public void onSuccess() {
        //下一步游戏申请读写内存和读取设备信息权限，申请完成之后初始化sdk
    }

    @Override
    public void onFailed(String msg) {
        //    玩家拒绝协议，游戏执行退出游戏操作
    }
});
```

使用这个接口所展示的协议弹窗，需要在 manifest 内 配置隐私协议地址和用户协议地址，否则协议标志点击无效果

```
<meta-data
    android:name="QG_PRIVACY_URL"
    android:value="QK[redacted]" />

<meta-data
    android:name="QG_USER_URL"
    android:value="QKhttps://www.baidu.com" />
```

注意协议前面保留QK标志

1. 1 设置注销账号回调

```
QGManager.setLogoutCallback(new QGCallBack() {
    @Override
```

```
        public void onSuccess() {
            Toast.makeText(mActivity, "注销成功", Toast.LENGTH_SHORT).show();
            //隐藏浮窗
            QGManager.hideFloat();
        }

        @Override
        public void onFailed(String msg) {
            Toast.makeText(mActivity, "注销失败", Toast.LENGTH_SHORT).show();

        }
    });
```

2. 初始化 （2 个接口）

QGManager .initMsa（Application application）;// 此接口需要在 Application 的 onCreate 之中调用，以便于获取 OAID;

//init 接口在主 Activty 的 onCreate 执行或申请权限的回调内执行

```
QGManager.init(MainActivity.this, "产品号（product_code）", new QGCallBack() {
        @Override
        public void onSuccess() {

        }


        @Override
        public void onFailed(String msg) {
        }
    });
```

3. 登录

```
QGManager.login(MainActivity.this, new QGCallBack() {
        @Override
```

```
        public void onSuccess() {

            QGManager.getUID();//获取 UID
            QGManager.getUserName();//获取 UserName
            QGManager.getLoginToken();//获取 Token


        }

        @Override
        public void onFailed(String msg) {
        }
    });
```

3.1 静默登录 (不显示界面，默认生成账号)

```
QGManager. slienceLogin (MainActivity.this, new QGCallBack() {
        @Override
        public void onSuccess() {

            QGManager.getUID();//获取 UID
            QGManager.getUserName();//获取 UserName
            QGManager.getLoginToken();//获取 Token


        }

        @Override
        public void onFailed(String msg) {
        }
    });
```

3.2 获取登录方式

```
4. /**
    * 获取登录方式  仅在第一次选择时生效  建议游戏做 uid 绑定 防止后续获取到的都是
   **自动登录
    *自动登录 0
    * 账号登录 1
    * 验证码登录 2
```

```
* qq 4
* 微信 5
* 阿里云一键授权 6
* taptap 7
*/
```

QGManager.getLoginType();

4.设置角色信息

```
QGRoleInfo roleInfo = new QGRoleInfo();
                roleInfo.setBalance("9999");
                roleInfo.setPartyName("天地");
                roleInfo.setRoleId("10086");
                roleInfo.setRoleName("白富美");
                roleInfo.setServerName("888");
                roleInfo.setVipLevel("999");
                roleInfo.setRoleLevel("999");//等级
                roleInfo. setRolePower ("999");//战斗力
                QGManager.setGameRoleInfo(MainActivity.this, roleInfo);
```

5.支付

```
QGOrderInfo mOrderInfo = new QGOrderInfo();
QGRoleInfo mRoleInfo = new QGRoleInfo();
//角色信息
mRoleInfo.setRoleId("123546421321");
mRoleInfo.setRoleLevel("1");
mRoleInfo.setRoleName("hhaha");
mRoleInfo.setServerName("zzz");
mRoleInfo.setVipLevel("666");
// 订单信息
mOrderInfo.setAmount("0.01");
mOrderInfo.setCount(1);   //数量
mOrderInfo.setExtrasParams("2017110403"); //订单号
mOrderInfo.setPayParam("eedwd");//   商品描述
mOrderInfo.setOrderSubject("钻石"); //  商品名称
mOrderInfo.setProductOrderId("2017110403");//cp 订单号

QGManager.pay(MainActivity.this, mRoleInfo, mOrderInfo, new QGCallBack() {
```

```
                @Override
                public void onSuccess() {

                }

                @Override
                public void onFailed(String msg) {

                }
        });
```

5.1 统计支付信息（非必接）

```
    //游戏确认到账发货时调用
QGManager.adPayStatistics(QGManager.getUID(),
                QGManager.getUserName(), "角色 id", "订单号", "商品 id", "商品名称", 总价, 货币);
```

6.注销

```
QGManager.logout(MainActivity.this);
```

7.生命周期

```
    @Override
    protected void onPause() {
        super.onPause();
        QGManager.hideFloat();
    }

    @Override
    protected void onResume() {
        super.onResume();
        if (!TextUtils.isEmpty(QGManager.getUID())) {
            QGManager.showFloat();
        }

    }
```

8.退出

```
QGManager.exit(MainActivity .this, new QGCallBack() {
                    @Override
                    public void onSuccess() {
                        finish();
                    }

                    @Override
                    public void onFailed(String msg) {
                    }
                });
```

9.显示/隐藏浮窗

QGManager.showFloat(true/false); //true: 浮窗默认倚靠屏幕左边
QGManager.hideFloat();

10.拓展登录方式授权相关

1.开启 QQ 授权登录

请修改 manifest 配置中的以下 2 处参数：

```
<activity
    android:name="com.tencent.tauth.AuthActivity"
    android:launchMode="singleTask"
    android:noHistory="true" >
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />

        <category android:name="android.intent.category.DEFAULT" />
        <category android:name="android.intent.category.BROWSABLE" />
        <!-- 替换为腾讯的appid    需要cp申请 -->
        <data android:scheme="tencent101936006" />
    </intent-filter>
</activity>
<!-- 0表示不支持QQ登录   需要cp申请 -->
<meta-data
    android:name="QQ_APP_ID"
    android:value="QK101936006" />
```

注意：meta-data 的 value 格式为 android:value="QK+你的 appid"

2.开启微信授权登录

如果游戏需要使用微信登录，需要将参数配置到manifest文件中的指定位置，并且新建一个WXEntryActivity类，继承QGWXEntryActivity类，放到包名路径下，并且在manifest中注册该Activity；此处可参照demo处理；如果不使用，则保持默认的0；

```
<!-- 0表示不支持微信登录   需要cp申请 wxce12bb34938c402a-->

<meta-data
    android:name="WX_APP_ID"
    android:value="QKwx36e63475538760fd" />
<meta-data
    android:name="WX_APP_SECRET"
    android:value="QKb2fb9c3d40af9e57be79cc278666f09b" />
```

```
package com.htjy.xycs.android.wxapi;
import com.quickgamesdk.activity.QGWXEntryActivity;

public class WXEntryActivity extends QGWXEntryActivity {

}
```

注意：meta-data 的 value 格式为 android:value="QK+你的 appid 或 appkey"

3.获取微信/QQ 的 openID：

```
public void getExtUserInfo() {
    String openId = QGManager.getExtInfo().getOauthId(); // 获取微信/qq 的 openId
    int openType = QGManager.getExtInfo().getOauthType(); // openType: 4 是微信
                                                          // 5 是 qq
}
```

5. 阿里云手机号一键授权登录

需要修改 manifest 的一处参数

```
<meta-data
    android:name="ALP_KEY"
    android:value="QKPHT5jFA9eDgfMrQ0Ejbg6BCWmEVI0ZnQ9kK5zhxtl7JRDP+L1zk/R2TKirUnj6LUnUACqqSEd2
```

注意：meta-data 的 value 格式为 android:value="QK+你的阿里云参数"（ps 阿里云的此处的 key 长度非常长，最少 128 位以上），并且 游戏包名 签名 要和 阿里云后台申请产品时所填入的信息保持一致。 测试时请保证插入了 sim 卡并且打开了移动网络

此处设置之后 还需要将阿里云后台的 appid 配置到部署版 SDK 后台的拓展配置那里

6.Taptap 授权登录

```
<meta-data
        android:name="TAP_ID"
        android:value="QKoNtf0bprDoiyjOzoS4" />

    <meta-data
        android:name="TAP_KEY"
        android:value="QKzsL8aSPySaHrpdOmJDYCEL21gMuYJHnm" />
```

注意：meta-data 的 value 格式为 android:value="QK+你的 appid 或 appkey"

11.防沉迷相关

SDK 自带防沉迷系统 会自动弹出防沉迷窗口 包含实名认证 未成年登录限制 和 充值限制 详情请参考功能操作文档

获取年龄&实名认证与否

获取年龄：
QGManager.getAge(); //返回 int 值，如果未实名则返回 0；

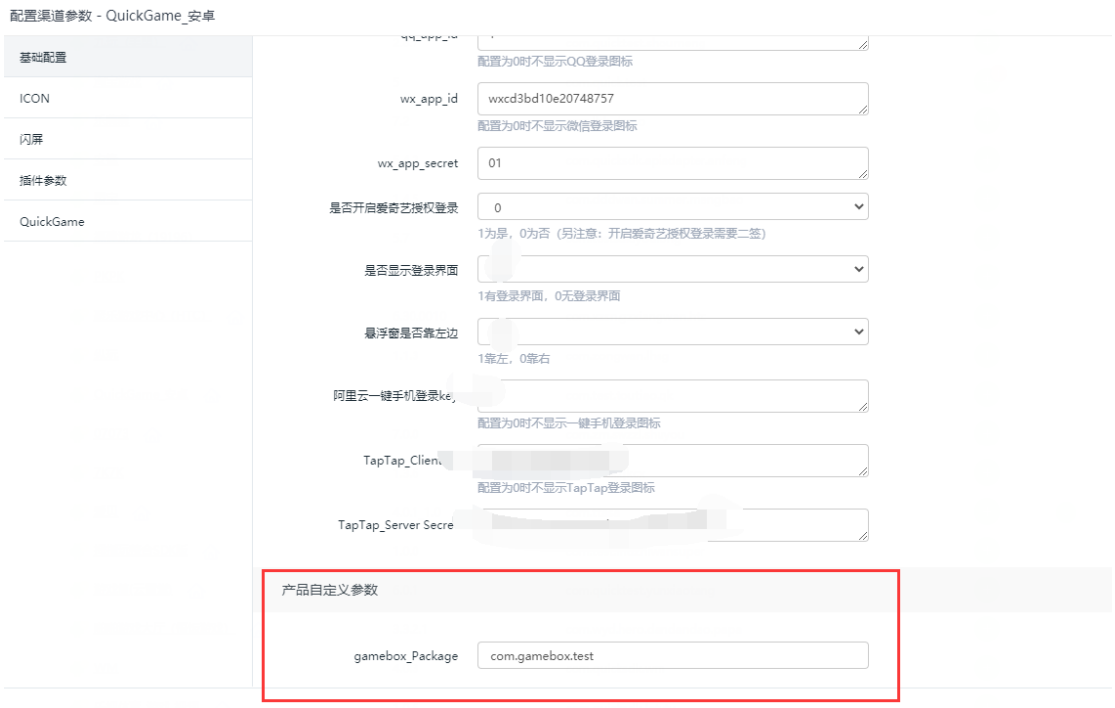
获取实名认证与否：
QGManager.getRealName()；

12.盒子归因

如果想要使用盒子的分包标识，请在打包时进行以下操作：

1. **如果是聚合打包**
   在 quickSDK 后台 添加自定义参数 gamebox_Package 然后填写游戏盒子的包名，此处一定要填写正确的盒子包名，然后保存参数 重新打包

   https://www.quicksdk.com/gameSet/channelList



2. 如果是 cp 单独出包　请在游戏工程内配置如下 meta-data

从 5.6.3->5.6.4　　仅需替换 QuicGame_xxxx.jar 即可

```
<meta-data
    android:name="QGGAMEBOX_PACKAGE"
    android:value="QKdefault" />
```

Value 的值为 QK+游戏盒子的包名

例如 QKcom.test.gamebox

```
<meta-data
    android:name="QGGAMEBOX_PACKAGE"
    android:value="QKcom.test.gamebox" />
```

13 头条巨量与快手买量调试

需要将 QK 后续的字段替换为对应买量 sdk 的参数

```
    <!-- 替换此处巨量参数 -->
    <meta-data
        android:name="tt_appid"
        android:value="QK479115" />
    <meta-data
        android:name="channelName"
        android:value="QKself_jrtt" />

    <!-- 替换此处快手参数 -->
    <meta-data
        android:name="ks_appId"
        android:value="QK0" />
    <meta-data
        android:name="ks_appName"
        android:value="QK0" />
    <meta-data
        android:name="ks_channel"
        android:value="QK0" />
```