迅雷NIUXSDK接入

修改记录

|  |  |  |
| --- | --- | --- |
| **日起** | **版本号** | **修改描述** |
| 2014-01-08 | 1.0.0 | NIUXSDK接入基本功能 |
| 2014-01-09 | 1.0.1 | 增加资源文件和AndroidManifest.xml描述 |
| 2014-01-20 | 1.0.3 | 1. 增加游戏内牛X助手  2. 增加账号登陆验证接口 |
| 2014-01-20 | 1.0.4 | 1. 扩展了充值接口 |
| 2014-02-13 | 1.0.5 | 1. 修复邮箱账号不能登陆的bug |
| 2014-08-25 | 2.0.5 | 1. 修复自动填入登陆账号密码，修改账号登陆提示参数错误的bug  2. 修复android 4.2以上系统无法调用电话拨号接口及拉起支付宝客户端接口bug |
| 2014-12-12 | 2.0.6 | 1. 接入账号库统一的登陆注册sdk接口  2. 支持用户名，邮箱，手机账号登陆  3. NiuxMobileGame增加appInit和appUnInit接口 |
| 2015-08-01 | 2.0.7 | 支付支持代金券功能 |
| 2015-09-06 | 2.1.0 | 1. 支持悬浮窗影藏、翻转显示功能 2. 悬浮窗点击展示礼包、活动、游戏中心推广入口 |
| 2015-10-13 | 2.2.0 | 1、增加手机注册、登陆功能。  2、增加自动登陆功能。 |
| 2015-11-30 | 2.2.1 | 1、修复手机登陆时数字账号bug |
| 2015-12-9 | 2.3.0 | 1. 增加退弹功能 2. 修改手机登陆、注册成功后的行为 3. 增加代码混淆 |
| 2015-12-15 | 2.3.1 | 1、读取登陆用户账号时不兼容老版本bug |
| 2015-12-22 | 2.3.2 | 1、支付宝升级 |
| 2016-1-13 | 2.3.3 | 1、解决退弹窗的bug |
| 2016-1-21 | 2.4.0 | 1. 解决悬浮窗横屏适配的问题 2. 增加开机广告弹窗接口 |
| 2016-2-29 | 2.4.1 | 1、增加微信支付方式 |
| 2016-4-19 | 2.5.0 | 1、新增绑银支付方式  2、新增开机关机广告转化率上报统计  3、修复部分账号登录提示账号异常的问题 |
| 2016-5-20 | 2.6.0. | 1、增加强更功能 |
| 2016-08-10 | 2.6.1 | 1、解决部分用户切换账号时闪退bug  2、礼包页面改版  3、增加sdk自检工具 |
| 2016-9-26 | 2.6.2. | 1、layout目录下资源文件重命名  2、增加crash日志上报 |
| 2016-10-24 | 2.6.3 | 1、增加登录错误上报  2、增加魅族手机开启悬浮窗提示  3、优化充值页面打开速度 |

# 概述

本文档仅供按照迅雷NIUXSDK标准对接手游的游戏方使用

# 迅雷

在对接之前，迅雷需要提供给游戏方以下资料：

1. 游戏对接需要的jar包在SDK2.x/lib/，（包含xlniuxmobilegame\_sdk-2.x.jar）
2. Niuxsdk资源文件在SDK2.x/res/目录内，SDK2.x/assets/目录下文件放到游戏对应assets目录下
3. 配置参数，列表如下：

gameId:迅雷的游戏Id

gameName:游戏名

appKey:APP的key，用于登陆时加密

exchangeRate:游戏币与人民币的兑换比例

exchangeUnit:游戏币单位

channelId:渠道Id

# 游戏方工作

游戏方APK需要接入以下内容：

1. 用户登陆
2. 用户充值
3. 数据上报，参考4.5.1，4.5.2
4. 悬浮助手
5. 退弹框

游戏方服务器需要对接以下接口：

1. 游戏充值回调接口，参考5.1
2. 玩家信息查询接口，参考5.2 （本接口有就对接）
3. 验证用户登陆是否有效接口，参考6.1

游戏方对接完毕后，可以使用自检工具apkchecker进行检验，该工具可用于检测apk静态文件中是否正确包含sdk中的相关静态文件及配置

# NIUXSDK包功能指南(游戏sdk接入内容)

## NIUXSDK对外接口简介

NIUXSDK向游戏方开放了两个类：（具体使用方法可参考demo）

1. com.xunlei.niux.mobilegame.sdk.listener.NiuxMobileGameListener 监听类。该类需要游戏方实现，包含的方法简介如下：

* void onSDKInitialized(int initialResult) 初始化结果监听，NIUXSDK初始化时会调用该监听方法。
* void onAcitvityFinish() 退出activity的监听；当niuxsdk提供的activity退出时，调用该监听方法。
* void onLoginFinish (int result, User user) 登陆完成监听；
* **v**oid onLogout() 退出登陆时监听
* **v**oid onChargeStart(int money, String serverid, String orderid) 开始支付监听
* **v**oid onChargeFinish(int money, String serverid, String orderid) 支付完成监听
* Void onSDKFinish() 退出游戏时监听

1. com.xunlei.niux.mobilegame.sdk.platform.NiuxMobileGame NIUXSDK向游戏方开放访问类，开放的方法如下：

* void appInit(Context context) 需要在Application的onCreate()中调用此初始化方法
* void appUnInit() 需要再Application的onTerminate()中调用此方法
* **v**oid initMobileGame(String gameID, String gameName, double exchangeRate, String exchangeUnit, String channelID, String appKey, NiuxMobileGameListener listener, Context context) 初始化niuxsdk
* int login(Context context) 如果之前从未登陆过，则进入登陆activity；如果之前登陆过，则自动登陆
* int userCenter(Context context)进入用户中心
* int chargeCenter(Context context, String roleId, String serverId) 不带金额充值
* int chargeCenter(Context context, String roleId, String serverId, int money)带金额充值
* int chargeCenter(Context context, String roleId, String serverId, int money, ,String extParam)带金额和扩展参数充值
* **i**nt changeUser(Context context) 退出并返回登陆界面
* void logout(Context context) 用户账号登出
* void showExitDialog(Context context) 显示退弹框

## 游戏账号

游戏账号为sdk的user对象的customerId

## 账户功能调用

### 账户登陆

Niuxsdk成功初始化后，可以调用用户登陆功能。具体代码如下：

/\*\*

\* 显示登录界面。

\* **@param** context

\* **@return**

\*/

**public** **int** login(Context context)

/\*\*

\* 判断登陆是否成功

\* **@return**

\*/

**public** **boolean** isLogin()

### 账户信息

用户成功登陆后，合作方可以通过getLoginUser方法获取用户信息

/\*\*

\* 获取用户信息

\* **@return** 用户信息

\*/

**public** User getLoginUser()

其中User对象的customerId为游戏账户，游戏充值就是按照customerId到账;cutomerKey为本次登陆验证的key。customerKey可以在用户登陆5分钟内验证customerId的有效性，超出该时间会验证失效。

### 账户中心

登陆成功后，用户可以进入账户中心，进入方法如下：

/\*\*

\* 进入用户中心

\* **@param** context

\* **@return**

\*/

**public** **int** userCenter(Context context)

### 账户退出

有两个方法可以退出，**changeUser(context)** 退出并返回登陆界面；**logout(context)直接退出。具体如下：**

/\*\*

\* 退出登陆，并进入登陆界面

\* **@param** context

\* **@return**

\*/

**public** **int** changeUser(Context context)

/\*\*

\* 直接退出

\* **@param** context

\* **@return**

\*/

**public** **int** logout(Context context)

## 充值

### 不带金额充值

/\*\*

\* 不带金额的充值中心

\*

\* **@param** context

\* **@param** roleId

\* 充值发生的游戏角色ID

\* **@param** serverId

\* 充值发生的服务器ID

\* **@return**

\*/

**public** **int** chargeCenter(Context context, String roleId, String serverId)

### 带金额充值

/\*\*

\* 带金额的充值中心

\*

\* **@param** context

\* **@param** roleId

\* 充值发生的游戏角色ID

\* **@param** serverId

\* 充值发生的服务器ID

\* **@param** money

\* 充值金额

\* **@return**

\*/

**public** **int** chargeCenter(Context context, String roleId, String serverId,

**int** money)

### 待扩展参数的充值

通过该方法游戏方可以在下单时传入需要的参数下单充值。用户支付成功后，迅雷会通过充值回调接口将该参数回传给游戏方。回调接口参考4.1

/\*\*

\* 待扩张字段的充值

\* **@param** context

\* **@param** roleId 充值发生的游戏角色ID

\* **@param** serverId 充值发生的服务器ID

\* **@param** money 充值金额

\* **@param** extParam 扩张字段，充值成功后该字段将传给游戏方

\* **@return**

\*/

**public** **int** chargeCenter(Context context,String roleId,String serverId,**int** money,String extParam)

## 数据上报

### 进入游戏上报

/\*\*

\* 进入游戏上报

\* **@param** context

\* **@param** roleID 角色Id

\* **@param** roleName 角色名字

\* **@param** serverID 区服id

\* **@return**

\*/

**public** **int** enterGame(Context context, String roleID, String roleName, String serverID)

（用op=enter\_game关键字查看日志确认调用）

### 创建角色上报

/\*\*

\* 创建角色的统计上报

\* **@param** context

\* **@param** roleID 角色ID

\* **@param** roleName 角色名称

\* **@param** serverID 服务器ID

\* **@return**

\*/

**public** **int** createRole(Context context, String roleID, String roleName, String serverID)

（用op=create\_role关键字查看日志确认调用）

## AndroidManifest.xml配置

### 开放权限

以AndroidManifest.xml为准

### 注册activity

以AndroidManifest.xml为准

## 游戏内牛X助手

在游戏页Activity的resume方法中调用NiuxMobileGame.getInstance().createFloatView方法创建牛X助手悬浮窗口；在activity的stop()方法中调用NiuxMobileGame.getInstance()

.destoryFloatView(activity)销毁牛X助手悬浮窗口。具体接口如下：

/\*\*

\* 创建牛X悬浮助手

\* **@param** activity

\*/

**public** **void** createFloatView(Activity activity)

/\*\*

\* 销毁牛X悬浮助手

\* **@param** activity

\*/

**public** **void** destoryFloatView(Activity activity)

## 退弹框

在游戏页Activity的onBackPressed方法中调用NiuxMobileGame.getInstance().showExitDialog方法显示退弹框；如果用户点击“退出游戏”按钮，则会调用NiuxMobileGameListener的onSDKFinish方法，游戏方可以在此方法中加入退出游戏的相应逻辑。具体接口如下：

/\*\*

\* 显示退弹框

\* **@param** activity

\*/

**public** **void** showExitDialog(Activity activity)

## 开机广告弹窗

在游戏监听类中，覆盖父类中onActivityFinish(boolean needShowAd)方法，加入相应逻辑，具体接口如下：

/\*\*

\* 显示开机广告弹窗

\* **@param** activity

\*/

**public** **void** showLoginAdDialog(boolean needShowAd)

另外提供了弹窗关闭的回调，如果有需要，可以监听该回调，具体接口如下：

**public** **void** onLoginAdDialogClosed()

## Crash日志上报

在全局Application类的onCreate方法中，调用以下代码：

NiuxCrashHandler.getInstance().init(this);

如果游戏方已实现UncaughtExceptionHandler，则在其逻辑中直接调用上报我方平台的方法:

CrashReportService.reportCrash(errContent);

# 游戏方提供的接口

## 游戏充值接口

### 传入参数

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **备注** |
| **orderid** | 充值订单号 | 由平台生成，D+18位数字，总共19位  例如：D121010984734196997 |
| **user** | 充值用户 | 该值是urlencode之后的。  用户名格式为6到20个字符，使用字母、数字、中文。 |
| **gold** | 游戏币数量 | 整数，通过金额和兑换比例得到 |
| **money** | 支付金额 | 整数，单位：元 |
| **time** | UNIX时间戳 | 用于超时验证，同时加密验证里面要该参数 |
| **sign** | 加密串 | md5(orderid + user + gold + money + time + 密钥)  密钥请准备好回调地址之后联系我们来配置！此处user是urlencode前的值！ |
| **server** | 服务器编号 | 整数，如1服：server=1 |
| **ip** | 用户ip |  |
| **ext** | 扩展字段 | 支付扩展字段。 |
| **roleid** 角色编号 不需要时，不传这个值。如果同一个账号同一个区服（包括合服）存在多个角色的情况， 并且充值需要选择充值角色时传入，这个参数。并且需要提供多角色查询接口 | | |

备注：参数中user为niuxsdk中的customerId

服务器ip：

123.151.31.232

125.39.36.232

123.151.31.233

125.39.36.233

123.151.31.141

125.39.36.141

123.151.31.210

125.39.36.210

123.151.31.96  
111.161.125.253

### 返回值

|  |  |
| --- | --- |
| **返回值** | **备注** |
| **1** | 成功 |
| **2** | 订单重复 |
| **-1** | 提交参数不全 |
| **-2** | 验证失败 |
| **-3** | 用户不存在 |
| **-4** | 请求超时 |
| **-5** | 服务器编号错误 |
| **-6** | Ip限制 |

建议合作方能够添加更加完善的安全措施，比如ip限制。

## 玩家信息查询

### 传入参数

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **备注** |
| **customerid** | 充值用户 | 该值是urlencode之后的。 |
| **time** | UNIX时间戳 | 用于超时验证，同时加密验证里面要该参数 |
| **sign** | 加密串 | md5(user + time + 密钥)  密钥由游戏方指定！此处user是urlencode前的值！ |
| **server** | 服务器编号 | 整数，如1服：server=1 |

### 返回

返回Json数据对应Array格式：

返回信息：用户账号，游戏角色名称，角色id，必填。

用户账号user 、游戏角色名称nickname、等角色id 必填，其他值可以为空，中文返回unicode编码 ，例如角色名等 中文

例子：{"status":0,"data":[{"user":"xlmxqy0000000001","roleid":"1","nickname":"\u7389\u7af9\u73ca\u4e39","level":"43"},{"user":"xlmxqy0000000001","roleid":"2","nickname":"\u6da6\u601c\u82b1\u6d60","level":"54"}]}

　　 Status 状态码 说明，只有在查询成功时才返回data数据

|  |  |
| --- | --- |
| **返回值** | **备注** |
| **0** | 查询成功 |
| **-1** | 提交参数不全 |
| **-2** | 验证失败 |
| **-3** | 用户尚未创建角色 |

# 迅雷提供可调用接口

## 验证用户登陆的customerId是否有效的接口

接口：

<http://websvr.niu.xunlei.com/checkAppUser.gameUserInfo?gameid=000146&customerid=xlmxhy0000000017&customerKey=1390233954db09090c6d66da10ecb3c93dc23d85ef>

返回值：{"code":0,"msg":"验证成功"}

注：customerKey可以在用户登陆5分钟内从getLoginUser方法获取的user对象中获取

code :0 验证成功

1 参数为空

2 签名错误

3 该账号没有登陆过游戏

4 没有配置该游戏