快看游戏 SDK 服务端文档

> 文档版本：v1.0.2
> 创建日期：2017.1.5

日期	修订内容	备注
1.22	1.sign 签名须进行 urlencode 2.支付通知地址参数更正 out_notify_url	
2.10	1.http 请求头去掉 X-Device	

登录

线上域名：http://api.kkmh.com

1. 验证open_id合法性接口

接口描述

> 接口地址：/v1/game/oauth/check_open_id
> 请求方式：GET

接口参数

参数名称	参数含义	示例	备注
app_id			
open_id			
access_token			
sign	签名		urlencode，签名算法见**支付**部分

参数名称	参数含义	示例	备注

接口返回

```
{
  "code": 200,
  "message": "OK",
  "data": {
    "ret": true //open_id正确此值为true，否则为false
  }
}
```

2. 获取用户信息

接口描述

> 接口地址：/v1/game/oauth/user_info
> 请求方式：GET

游戏方调用该 API 时，需要区分授权用户和游客，游客不要调用该 API，否则会报错。

接口参数

参数名称	参数含义	示例	备注
access_token			

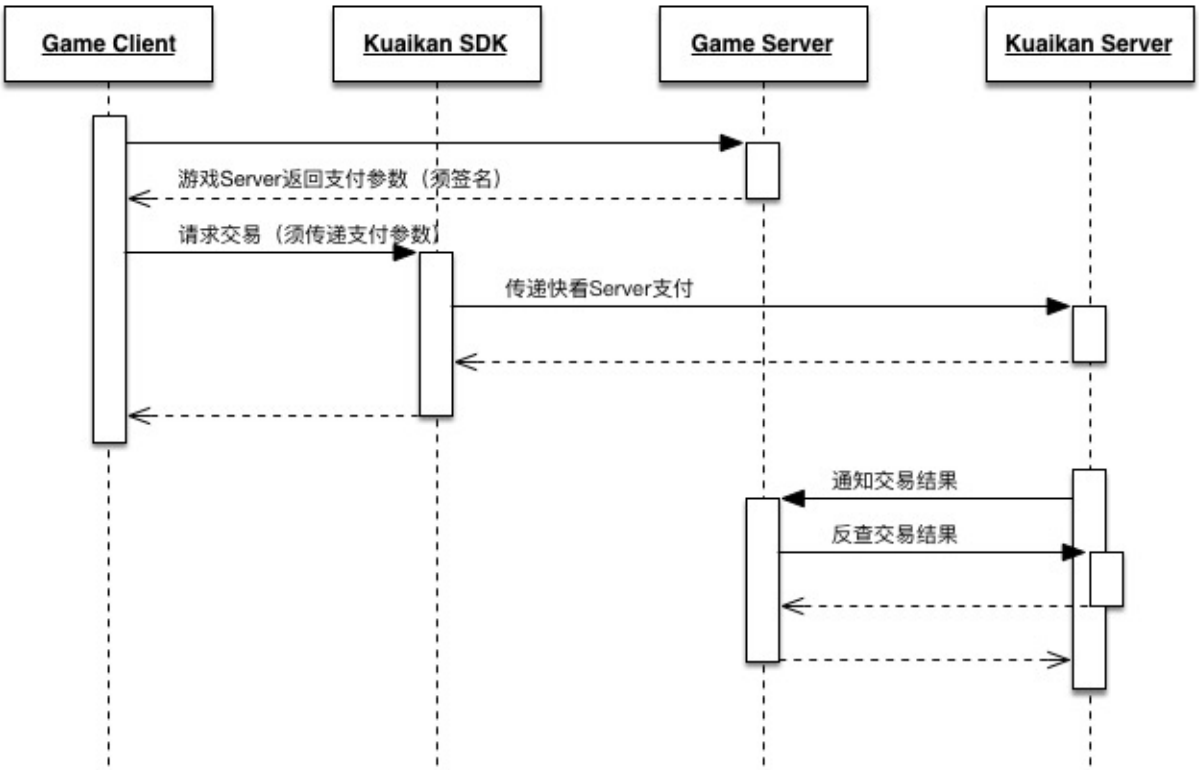
接口返回

```
{
  "code": 200,
  "message": "OK",
  "data": {
    "open_id": "120001",
    "nickname": "nick",
    "avatar_url": "touxiang",
  }
}
```

支付

> 游戏申请接入后，快看会为游戏分配 **app_id**；游戏方须向快看提供游戏内商品列表，以便分配对应的 **wares_id** 给游戏方。

1. 时序图



2. 签名规则

签名算法

签名生成的通用步骤如下：

step1：设所有发送或者接收到的数据为集合 M，将集合 M 内非空参数值的参数按照参数名 ASCII 码从小到大排序（字典序），使用 URL 键值对的格式（即key1=value1&key2=value2…）拼接成字符串 stringA。

特别注意以下重要规则：

- 参数名 ASCII 码从小到大排序（字典序）；
- 如果参数的值为空不参与签名；
- 参数名区分大小写；
- 验证调用返回或快看主动通知签名时，传送的 sign 参数不参与签名，将生成的签名与该 sign 值作校验。

step2：在 stringA 最后拼接上 key 得到 stringSignTemp 字符串，并对 stringSignTemp 进行 MD5 运算，在进行 base64，得到 sign 值 signValue。

假设传送的参数如下：

app_id: 1024
out_order_id: 110
wares_id: 1
open_uid: 88881024
out_notify_url: http://staging.kuaikanmanhua.com/v2/game_pay/cp_notify

分配的快看秘钥：`QRAd2rAZO7RQIDAQAB`

1、对参数按照 key=value 的格式，并按照参数名 ASCII 字典序排序如下：

```
String stringA="app_id=1024&open_uid=88881024&out_notify_url=http://stagi
ng.kuaikanmanhua.com/v2/game_pay/cp_notify&out_order_id=1104&wares_id=1";
```

2、拼接 API 密钥：

```
String stringSignTemp="stringA&key=QRAd2rAZO7RQIDAQAB"
MessageDigest md5 = MessageDigest.getInstance("MD5");
BASE64Encoder base64en = new BASE64Encoder();
//加密后的字符串
String sign=base64en.encode(md5.digest(stringSignTemp.getBytes("utf-
8")));
```

最终得到最终发送的数据：

```
gldiA2tbDVBKM/OfRQo63Q==
```

java代码举例：

```java
package com.kuaikan.game.pay.sign;

import lombok.extern.slf4j.Slf4j;
import org.apache.commons.lang.StringUtils;
import org.springframework.util.CollectionUtils;
import sun.misc.BASE64Encoder;

import java.security.MessageDigest;
import java.util.Iterator;
import java.util.Map;
import java.util.Set;
import java.util.TreeMap;

/**
 * Created by weizhiyu on 16/12/19.
 */
@Slf4j
public class KuaikanSignUtil {
    public static String getSign(String key, Map<String, Object> paramMa
p) {
        try {
            if (StringUtils.isBlank(key) || CollectionUtils.isEmpty(param
Map)) {
                return null;
            }
            TreeMap<String, Object> signMap = new TreeMap<>(paramMap);
            StringBuffer sb = new StringBuffer();
            Set es = signMap.entrySet();
            Iterator it = es.iterator();
            while (it.hasNext()) {
                Map.Entry entry = (Map.Entry) it.next();
                String k = (String) entry.getKey();
                Object v = entry.getValue();
                if (v != null && !"".equals(v) && !"sign".equals(k) &&
!"key".equals(k)) {
                    sb.append(k + "=" + String.valueOf(v) + "&");
                }
            }
            sb.append("key=" + key);
            String md5Str = sb.toString();
            MessageDigest md5 = MessageDigest.getInstance("MD5");
            BASE64Encoder base64en = new BASE64Encoder();
            //加密后的字符串
            String sign = base64en.encode(md5.digest(md5Str.getBytes("utf-
8")));
         return sign;
      }
      catch (Exception e) {
          String errorMessage = "KuaikanSignUtil getServerSign error";
          log.error(errorMessage, e);
          }
```

```
            return null;
        }
        public static boolean checkSign(String key,String sign, Map<String,
    Object> paramMap) {
            if (CollectionUtils.isEmpty(paramMap)) {
                return false;
            }
            if (StringUtils.isBlank(sign)) {
                return false;
            }
        String checkSign = getSign(key, paramMap);
            if (StringUtils.isBlank(checkSign)) {
                return false;
            }
            if (!sign.equals(checkSign)) {
                return false;
            }
        return true;
       }
  }
```

3. API接口

线上域名：http://api.kkmh.com

注意：

- 所有接口都需要签名验证
- 下单接口有用户OAuth验证，用户必须是快看注册用户
- out_order_id必须唯一

code错误提示码:

错误码	解析
200	成功
2005	服务异常
7001	app_id不存在
7002	签名错误
7003	请求参数错误
7004	用户oauth认证失败
7005	app_id下的out_order_id已经存在
7006	查询的订单 不存在
7007	下单失败

1. 下单接口

SDK已封装下单接口，游戏按照要求格式传参即可。

传参格式示例

```
trans_data=
{"app_id":"123","wares_id":1,"out_order_id":"22222","open_uid":"123","out
_notify_url":"http://www.iapppay.com/test"}&sign=xxxxxx
```

请求参数

参数名称	参数含义	数据类型	是否可选	参数说明
app_id	应用编号	string(20)	必填	快看平台分配的应用编号
wares_id	商品ID	integer	必填	商品ID
out_order_id	游戏订单号	string(64)	必填	游戏订单号
open_uid	用户在游戏应用的唯一标识	string(20)	必填	
out_notify_url	支付结果通知URL	string(500)	必填	
sign	签名		必填	注意：sign参数不在trans_data中

2. 订单查询接口

接口描述

> 接口地址：/v2/game_pay/query_order
> 请求方式：POST

传参格式示例

```
trans_data={"app_id":"123","order_id":111111,"out_order_id":"22222","noti
fy_url":"http://www.iapppay.com/test"}&sign=xxxxxx
```

请求参数

参数名称	参数含义	数据类型	是否可选	参数说明
app_id	引用编号	string(20)	必填	快看平台分配的应用编号

参数名称	参数含义	数据类型	是否可选	参数说明
order_id	快看订单ID	string(64)	选填(order_id,out_order_id二选一)	快看订单ID
out_order_id	游戏订单号	string(64)	选填(order_id,out_order_id二选一)	游戏订单号
sign	签名		必填	注意：sign参数不在trans_data中

返回格式示例

```
{  "code": 200,
   "data": {
     "trans_data": "{\"wares_id\":1,\"pay_status\":2,\"out_order_id\":\"1104\",\"trans_money\":0.0,\"trans_id\":\"32461612231438102462\",\"currency\":\"RMB\",\"pay_type\":402,\"trans_time\":1482475126000,\"open_uid\":\"1104\",\"order_id\":\"75010856699650048888881024\",\"app_id\":1}",
     "sign": "DhA6FwTq/2omXra9EMTJ4Q=="
    },
    "message": "ok"
}
```

返回参数

参数名称	参数含义	数据类型	参数说明
code	错误码	integer	错误提示码200成功；其他错误码时不会返回data{}结构
msg	错误信息	string	快看订单ID
data	数据信息		data{}结构

data结构

参数名称	参数含义	数据类型	参数说明
trans_data	订单数据信息		trans_data{}数据结构
sign	签名		trans_data{}结构中的数据的签名验证

trans_data结构

参数名称	参数含义	数据类型	参数说明
app_id	应用编号	string(20)	
order_id	快看订单ID	string(64)	

参数名称	参数含义	数据类型	参数说明
out_order_id	游戏订单号	string(64)	
open_uid	用户在游戏应用的唯一标识	string(20)	
wares_id	商品ID	integer	
trans_id	交易流水号	string(64)	
trans_money	交易金额	float(6,2)	
currency	币种	string	现在只支持RMB
pay_type	支付类型	integer	支付方式
pay_status	支付状态	integer	支付状态：1-WAIT_BUYER_PAY 2-SUCCESS 3-CLOSED
trans_time	成交时间	long	时间戳

3. 订单支付成功通知

订单成功支付时，快看服务器向游戏服务器发起 **POST** 请求，游戏方可通过**“ 2. 订单查询接口”** 确认订单有效性，游戏方在确认订单处理成功后返回给快看 **SUCCESS** 字符串（必须大写），否则快看认为游戏方没有成功接收支付通知。支付联调前，此接口务必设好返回值。

传参格式示例

```
trans_data={\"wares_id\":1,\"pay_status\":2,\"out_order_id
\":\"1104\",\"trans_money\":0.0,\"trans_id\":\"32461612231438102462\",\"c
urrency\":\"RMB\",\"pay_type\":402,\"trans_result\":1,\"trans_time\":1482
475126000,\"open_uid\":\"1104\",\"order_id\":\"75010856699650048888881024
\",\"app_id\":1} &sign=xxxxxx
```

trans_data结构

参数名称	参数含义	数据类型	参数说明
app_id	应用编号	string(20)	
order_id	快看订单ID	string(64)	
out_order_id	游戏订单号	string(64)	
open_uid	用户在游戏应用的唯一标识	string(20)	
wares_id	商品ID	integer	

参数名称	参数含义	数据类型	参数说明
trans_id	交易流水号	string(64)	
trans_money	交易金额	float(6,2)	
currency	币种	string	现在只支持RMB
pay_type	支付类型	integer	支付方式
pay_status	支付状态	integer	支付状态：1-WAIT_BUYER_PAY 2-SUCCESS 3-CLOSED
trans_time	成交时间	long	时间戳
trans_result			本字段只参与签名，无实际意义