TTSDK 服务端接入说明

V 2.1.3

2016 年 6 月 15 日

广州谊游网络科技有限公司

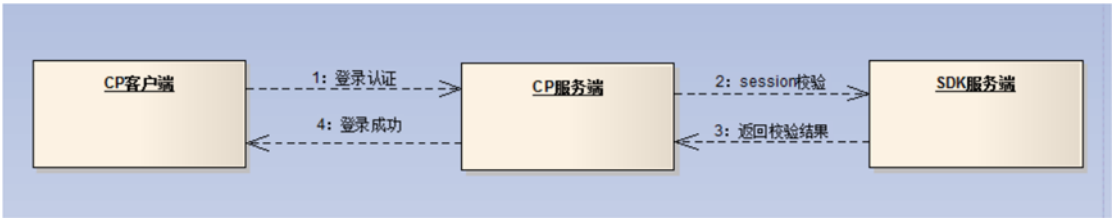
版本历史

版本	日期	作者	说明
V 2.1.3	2016-06-15		修改说明
V 2.1.2	2016-05-26		更改验证地址，修改参数说明
V 2.0.0	2016-01-26		

本文档主要描述谊游游戏平台“SDK 服务器”和游戏合作商“CP 游戏服务器”的交互的接口规范。

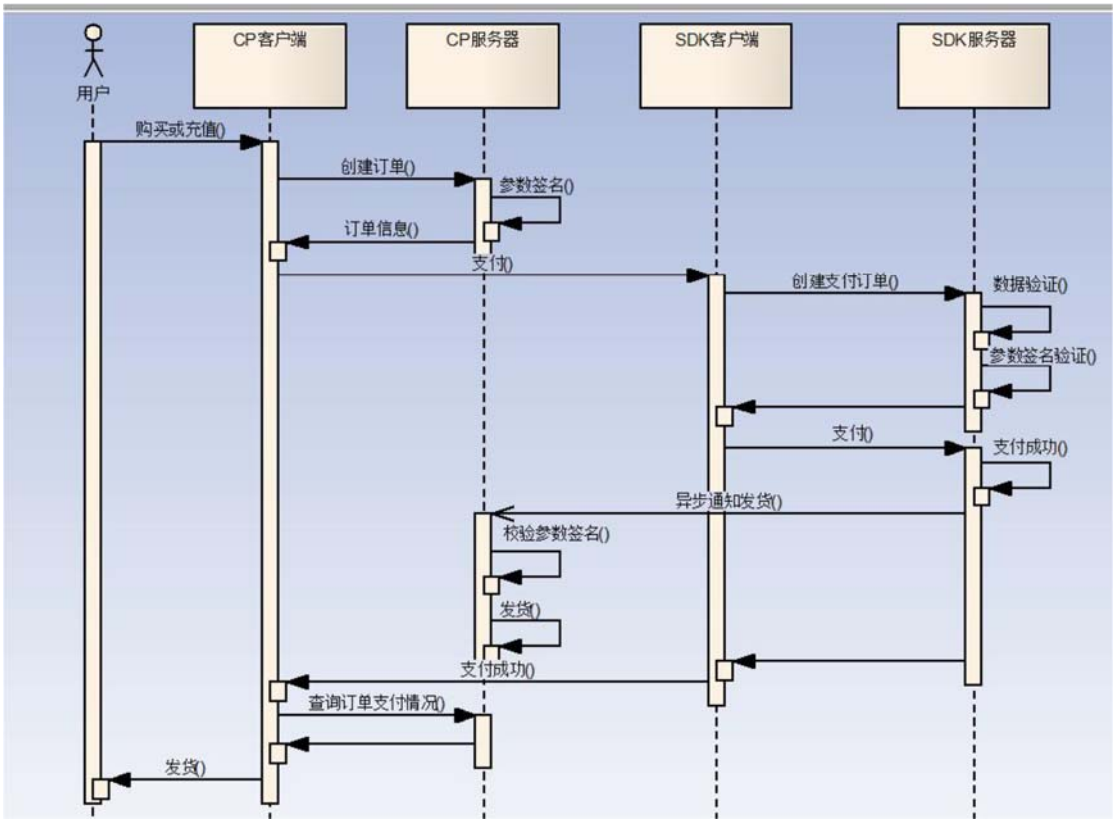
1. 流程

1.1. SESSION 校验流程



SDK 登录成功后，CP 客户端即与 CP 服务器进行登录认证操作。在认证过程中，CP 服务器需要通过 SDK 服务器校验客户端提交的 session_id 是否有效。

1.2. 购买流程



1) 用户发起购买请求后，首先 CP 客户端将订单信息提交到 CP 服务端创建订单，服务端将订单信息返回给客户端

2) 客户端调用 SDK 相关支付方法，SDK 将引导用户完成支付

3) 支付成功后，SDK 服务端会异步通知 CP 服务器进行发货

4) 本流程省略了 SDK 与支付中心的支付流程

2. 接口通用说明

2.1. 通信协议

SDK Server 采用 HTTP 协议作为通信协议，类 REST 风格，客户端通过构造 HTTP 请求，向服务器发起调用。

2.2. 数据格式

请求消息和响应消息的内容都使用 JSON 格式。

通用的数据报文包含 head 和 body 两部分，如：

```
{
    "head":{
        "command":"add",
        "version":"0",
        "ctime":"2015-05-15"
    },
    "body":{
        "key":"value"
    }
}
```

一般情况下，如清晰明了，报文也可以简化，如：

```
{
    "result":"0",
    "message":"成功"
}
```

传输时使用紧凑模式，即删除所有不必要的空白，以减少传输量。

2.3. 字符编码

数据报文采用 utf-8 字符编码。

2.4. 签名

签名说明：

签名用于跟 SDK 服务器做交互验证。

1. **登录验证： CP 服务器 ->SDK 服务器**

客户端登录后，在 CP 服务器中， 使用 gameId、uid 以 json 的格式加上交互密钥进行签名,如:

{"gameId":"20150812","uid":"3459079"} + key

签名后，放在请求的 Header 中。

2. **充值验证： SDK 服务器 -> CP 服务器**

支付完成后 SDK 服务器会以签名+报文的形式通过 POST 方式通知 CP 服务器，CP 服务器收到后，将报文进行 UrlDecode 后，再加上密钥（充值交互 key）进

行签名。验证跟 SDK 服务器返回的签名是否一致（SDK 服务器返回的签名放在请求的 Header 中），验证成功后，再进行响应（格式参照 4.1.6）。

签名的方法是否一致：

接口请求时，对请求报文（如：{"command":"add"}）+ 密钥（如：1234567890）所得字串（如：{"command":"add"}1234567890）进行 MD5 摘要（一个 byte 数组，不需要转为 HEX String），并对摘要作 BASE64 编码（如：4PIfnb1IQ+6gsbQoSKmD6w==）

JAVA 签名样例代码：

```
import java.security.MessageDigest;
import sun.misc.BASE64Encoder;

public class Sign {
    public static String encodeBASE64(byte[] key) throws Exception {
        return (new BASE64Encoder()).encodeBuffer(key);
    }

    public static byte[] digestMD5(byte[] data) throws Exception {
        MessageDigest md5 = MessageDigest.getInstance("MD5");
        md5.update(data);
        return md5.digest();
    }

    public static void main(String[] args) throws Exception {
        // 需要加密的数据, UrlDecode之后的报文
        String data =
"{\"cpOrderId\":\"1291012688050000000\",\"gameId\":201500000,\"payDate
\":\"2016-04-21
12:05:35\",\"payFee\":\"1.00\",\"payResult\":\"1\",\"sdkOrderId\":\"20160421
120535101\",\"uid\":123456}";
        String key = "1234568997ab"; // 加密用的秘钥
        String src = data + key; // 密钥+数据组合源串
        // 加密后,生成的bytes字节数组
        byte[] enBytes = digestMD5(src.getBytes("utf-8"));
        String sign = encodeBASE64(enBytes);
        // 将byte数组 通过base64 encode成String类型.
        System.out.println(sign);
    }
}
```

PHP 签名方式

```
<?php
//需要加密的数据, UrlDecode 之后的报文
$var
="{\"cpOrderId\":\"1291012688050000000\",\"gameId\":201500000,\"payDa
te\":\"2016-04-21
12:05:35\",\"payFee\":\"1.00\",\"payResult\":\"1\",\"sdkOrderId\":\"20160421
120535101\",\"uid\":123456}";
$key = "1234568997ab"; // 加密用的秘钥 ↳
$sign = base64_encode(md5($var . $key, true));
echo $sign
```

3. 用户类接口

3.1. 登录状态查询

3.1.1. 描述

用户登录后，服务器返回 sid（会话 token），sid 在客户端获取。

之后每请求时，在自定义的 HTTP Header 一并发送 sid，Header 中（如：sid：TT3RDTK_TT_aaakSXnttJyPJgC8MXvrv），sid 检验失败将返回用户未登录或会话已超时错误。用户登出后，sid 失效。

登录验证流程：

1. 在 CP 客户端登陆时，可以获取到 sid：TTSDKV2.getInstance().getSession();
2. 将获取到的 sid 传到 CP 服务端。
3. CP 服务端将客户端传过来的 sid 和签名放在 POST 的 Header 中，和请求报文一并发送到 SDK 服务器。（POST 格式参照 3.1.3）
4. SDK 服务器收到请求后，会响应内容。（格式参照 3.1.5）

3.1.2. 请求地址

ContextPath：

TTSDK 测试环境：http://123.59.85.209:8080/server

TTSDK 正式环境：http://123.59.74.32/server

[ContextPath]/rest/user/loginstatus.view

注: CP 拿到的参数，若为正式参数,则请求到 TTSDK 的正式环境地址,否则请求到测试环境.

3.1.3. 请求内容

HTTP Header 中放入 sid, sign 签名.

```
"Content-Type":"application/json",
"sign":"wosTJy39ftJi0VOSJ4jvjg==",
"sid":"TT3RDTK_TT_aaakSXnttJyPJgC8MXvrv"
```

报文中使用以下参数, json 格式.

名称	类型	父节点	必填	说明
uid	long	-	Y	用户 ID
gameId	Int	-	Y	游戏 ID

样例：

```
{
        "gameId":225,
        "uid":234721
}
```

Request Details

```
{
"method": "POST",
"url":"http://123.59.85.209:8080/server/rest/user/loginstatus.view",
"headers":{
    "Content-Type":"application/json",
    "sign":"wosTJy39ftJi0VOSJ4jvjg==",
    "sid":"TT3RDTK_TT_aaakSXnttJyPJgC8MXvrv"
    },
"data": "{"gameId":20150812,"uid":3459079}"
}
```

3.1.4. 请求样例

使用 HTTP POST 方式。

JAVA 请求样例：

a.使用 HttpURLConnection 的方式

```
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.HashMap;
import java.util.Map;

import com.alibaba.fastjson.JSONObject;

public class TestPost {

    public static byte[] readInputStream(InputStream inStream) throws
Exception {
        ByteArrayOutputStream outStream = new ByteArrayOutputStream();
        byte[] buffer = new byte[1024];
        int len = 0;
        while ((len = inStream.read(buffer)) != -1) {
            outStream.write(buffer, 0, len);
        }
        byte[] data = outStream.toByteArray();// 网页的二进制数据
        outStream.close();
        inStream.close();
        return data;
    }

    public static byte[] sendPostRequestByForm(String path, String sid,
String sign, String params) throws Exception {
        URL url = new URL(path);
        HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
        conn.setRequestMethod("POST");// 提交模式
        // 设置访问提交模式，表单提交
        conn.setRequestProperty("Content-Type", "application/json");
        conn.setRequestProperty("sid", sid);
        conn.setRequestProperty("sign", sign);

        conn.setDoOutput(true);// 是否输入参数
        byte[] bypes = params.toString().getBytes();
        conn.getOutputStream().write(bypes);// 输入参数
```

```
        InputStream inStream = conn.getInputStream();

        return readInputStream(inStream);
    }

    public static void main(String[] args) {
        // 校验地址
        String url =
"http://123.59.85.209:8080/server/rest/user/loginstatus.view";
        // sid, 客户端获取到的session id 会话token 测试参数
        String sid = "TT3RDTK_TT_aaakSXnttJyPJgC8MXvrv";
        // sign 用文档中的签名方式 测试参数
        String sign = "wosTJy39ftJi0VOSJ4jvjg==";
        int gameid = 20150812;
        Long uid = 3459079L;
        // ----------------------组合json 报文----------------------------
        Map<String, Object> urldata = new HashMap<String, Object>();
        urldata.put("uid", uid);
        urldata.put("gameId", gameid);
        String jsonBody = JSONObject.toJSONString(urldata);

        try {
            String str = new String(sendPostRequestByForm(url, sid, sign,
jsonBody));
            System.out.println(str);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

b. spring 请求方式

```
import java.util.HashMap;
import java.util.Map;
import org.springframework.http.HttpEntity;
import org.springframework.http.HttpHeaders;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.web.client.RestTemplate;
import com.alibaba.fastjson.JSONObject;

public class TestPost {
    public static void main(String[] args) throws Exception {
        // 校验地址
        String url =
"http://123.59.85.209:8080/server/rest/user/loginstatus.view";
        // sid, 客户端获取到的session id 会话token 测试参数
        String sid = "TT3RDTK_TT_aaakSXnttJyPJgC8MXvrv";
        // sign 用文档中的签名方式 测试参数
        String sign = "wosTJy39ftJi0VOSJ4jvjg==";
        Long uid = 3459079L;
        int gameid = 20150812;
        // ----------------------组合json 报文----------------------------
        Map<String, Object> urldata = new HashMap<String, Object>();
        urldata.put("gameId", gameid);
        urldata.put("uid", uid);
        String jsonBody = JSONObject.toJSONString(urldata);
        // ----------------------组合HttpHeader----------------------------
        HttpHeaders httpHeaders = new HttpHeaders();
        httpHeaders.setContentType(MediaType.APPLICATION_JSON);
        httpHeaders.add("sid", sid);
        httpHeaders.add("sign", sign);
        // ---------------获取响应内容----------------------------------------
        HttpEntity<String> httpEntity = new HttpEntity<>(jsonBody,
httpHeaders);
        RestTemplate restTemplate = new RestTemplate();
        ResponseEntity<Object> responseEntity =
restTemplate.postForEntity(url, httpEntity, Object.class);
        String strJson =
JSONObject.toJSONString(responseEntity.getBody());
        System.out.println(strJson);
    }
}
```

PHP 请求样例：

```php
<?php
//测试地址
$url = "http://123.59.85.209:8080/server/rest/user/loginstatus.view";
// sid，客户端获取到的 session id  会话 token
$sid = "TT3RDTK_TT_aaakSXnttJyPJgC8MXvrv";
// 交互密钥
$key = "927afefb8d910016a096310d43d034d4";
//------------组合请求报文--------------------------
$str = array("gameId" => 20150812, "uid" => 3459079);
$urldata = json_encode($str);
//------------使用请求报文+密钥进行签名-----------------
$sign = base64_encode(md5($urldata . $key, true));
//---------------组合请求 header---------------------
$header = array(
      "Content-type:application/json",
      "sign:{$sign}",
      "sid:{$sid}",
);
//---------------使用 curl 进行请求-------------------------
$TIMEOUT = 30; //超时时间(秒)
$ch = curl_init();
curl_setopt($ch, CURLOPT_TIMEOUT, $TIMEOUT);
curl_setopt($ch, CURLOPT_CONNECTTIMEOUT, $TIMEOUT - 2);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_POSTFIELDS, $urldata);
curl_setopt($ch, CURLOPT_HTTPHEADER, $header);
ob_start();
curl_exec($ch);
$return_content = ob_get_contents();
ob_end_clean();
$return_code = curl_getinfo($ch, CURLINFO_HTTP_CODE);
echo "return_code=" . $return_code .
 "<br/>return_content=" . $return_content;
```

响应内容

名称	类型	父节点	说明
head	Object	-	
result	String	head	返回代码 0 为用户处于登录状态，其它为失败
message	String	head	说明

样例：

```
{
    "head":{
        "result":"0",
        "message":"用户处于登录状态"
    }
}
```

4. 支付类接口

4.1. 支付结果通知

4.1.1. 描述

用户完成支付后，通知 CP 方发货。

1. 用户支付完成，SDK 服务器会以签名+报文的形式 POST 到 CP 服务器，
2. CP 服务器收到后，将报文进行 UrlDecode 后，再加上密钥进行签名。
3. 验证跟 SDK 服务器返回的签名是否一致。验证成功后，再进行响应（响应格式参照 4.1.6）

注意：如果第一次调用失败，会重复多次调用该接口。系统可能会重复通知，商户必须保证对于多次重复调用与一次调用的结果是一致的。游戏服务端需要作参数签名验证，否则存在安全风险。

4.1.2. 请求地址

由 CP 提供,在客户端传入.

4.1.3. 请求方法

HTTP POST

4.1.4. 调用方向

SDK Server → CP Server

4.1.5. 请求内容(SDK 服务器端)

支付成功后，SDK 服务器发出的内容到 CP 服务器， 在请求 Header 中取到 sign（SDK 服务器签名），请求内容中获到报文（需进行 UrlDecode）。 SDK 服务器发送格式：

请求 HTTP Header 内容

```
sign: wosTJy39ftJi0VOSJ4jvjg==
```

请求报文内容

%7B%22cpOrderId%22%3A%22016042209404998600000ff8080815438de13%22%2C%22exInfo%22%3A%22%E6%89%A9%E5%B1%95%E4%BF%A1%E6%81%AF%22%2C%22gameId%22%3A20000%2C%22payDate%22%3A%222016-04-22+09%3A40%3A50%22%2C%22payFee%22%3A%220.01%22%2C%22payResult%22%3A%221%22%2C%22sdkOrderId%22%3A%222016042209405022 3%22%2C%22uid%22%3A5447918%7D

Request Details


```
{
"method": "POST",
"url":" "http://www.baidu.com",
"headers":{
      "Content-Type":"application/json;charset=utf-8",
      "sign":" wosTJy39ftJi0VOSJ4jvjg=="
      },
"data":
"%7B%22cpOrderId%22%3A%22016042209404998600000ff8080815438de13%22%2C%22exInfo%22%3A%22%E6%89%A9%E5%B1%95%E4%BF%A1%E6%81%AF%22%2C%22gameId%22%3A20000%2C%22payDate%22%3A%222016-04-22+09%3A40%3A50%22%2C%22payFee%22%3A%220.01%22%2C%22payResult%22%3A%221%22%2C%22sdkOrderId%22%3A%2201604220940502 23%22%2C%22uid%22%3A5447918%7D"
}
```


参数说明

名称	类型	父节点	说明
uid	long		用户 ID
gameId	Int		游戏 ID

sdkOrderId	String		SDK 订单号
cpOrderId	String		CP 订单号
payFee	Float		实际支付金额(单位元)
payResult	String		支付结果 1 成功，其它失败
payDate	String		支付时间，格式'yyyy-mm-dd hh:mm:ss'
exInfo	String		CP 扩展信息

样例：


```
{
    "uid":"20150515213406829",
    "gameId":"20150515213406829",
    "sdkOrderId":"20150515213406829",
    "cpOrderId":"20150515213406829",
    "payFee":16.00,
    "payResult":"1",
    "payDate":"2015-05-15 15:27:20",
    "exInfo":"abc"
}
```

4.1.6. 响应内容(CP 服务端)

测试是否正确响应：收到 SDK 服务端发出的请求进行响应后，在客户端中查询用户充值记录，如果显示充值中，则表示响应格式有误,请按下文中的格式进行正确响应

名称	类型	父节点	限制	说明
head	Object	-	1	
result	String	head	1	返回代码 0 为成功，其它为失败
message	String	head	1	说明

响应格式样例：

```
{
    "head":{
        "result":"0",
        "message":"成功"
    }
}
```

PHP 代码样例

```
<?php
//接收 SDK 服务器报文
$urldata = isset($GLOBALS["HTTP_RAW_POST_DATA"]) ?
$GLOBALS["HTTP_RAW_POST_DATA"] : '';
//从 http header 中的 SIGN 获取到 SDK 服务器的签名
$head = isset($_SERVER['HTTP_SIGN']) ? $_SERVER['HTTP_SIGN'] : '';

if (empty($urldata)) {
    echo "非法请求！";
```

```
    exit;
} else {
    //-----------------签名方式--------------------------
    $var = urldecode($urldata); //urldecode 请求报文
    $key = "123456789ab"; //充值秘钥
    $sign = base64_encode(md5($var . $key, true));

    //验证本地签名跟服务器的是一致,响应结果
    if ($sign == $head) {
        //验签成功
        $str ['head'] = array("result" => "0", "message" => "Success");
        $msg = json_encode($str);
        echo $msg;
    } else {

        //验签失败
        $str ['head'] = array("result" => "-1", "message" => "Error");
        $msg = json_encode($str);
        echo $msg;
    }
}
```