


1. SJ-GBS-API	2
2. CAS-API	5
3. CAS-OAuth 中央认证	5
4. GAS-API 1.0	7
5. GDS-API	13

SJ-GBS-API

 本文档为草稿, 意在选择沟通, 非最终文档!

名词解释:

- GBS - Game Billing Server

接口规范

接口采用OAuth 1.0a签名认证规范.

consumer key和secret请联系相关人员取得
access token 无须提供, 可直接调用相关接口
签名方法目前仅支持HMAC-SHA1.

接口地址规则

`http(s)://_game_billing_server_/gbs/internalapi/_method_name_`



 线上测试：
测试地址：<http://gbs0.sj.playcool.com>

测试API key：GAME-KEY
测试API secret：GAME-SECRET

返回数据格式

返回JSON格式数据, 字符集编码为UTF-8.

数据	说明
status	返回状态, 0表示返回正常, 其它表示不正常, 具体状态表示以具体接口为准
data	返回数据
error	错误描述

 如果返回不是JSON格式,无法json_decode
如果返回不是JSON格式, 可能是服务器故障, 请重试, 如果多次不正常, 说明真的服务器挂了, 赶快通知相关部门处理吧

资产查询

gbs.getAsset - 游戏用户资产查询接口, 获取用户资产信息

URL:

`/gbs/internalapi/gbs.getAsset`

参数：

参数	说明
userid	用户标志（CAS系统中的用户标志）

状态说明：

返回状态	说明
0	获取用户资产成功
1	无法找到该用户资产或尚未建立

返回数据：

当获取用户资产存在时, 返回的数据为二维数组, KEY 为币种标志, VALUE 是资产值, 币种标志具体查看 [附表-币种配置说明](#)

成功样例
<pre>{ "status":0, "data":{ "11": "189.00", "12":null }, "error":null }</pre>
失败样例
<pre>{ "status":1, "data":null, "error": "\u65e0\u6cd5\u627e\u5230\u8be5\u7528\u6237\u8d44\u4ea7\u6216\u5c1a\u672a\u5efa\u7acb" }</pre>

消费接口

gbs.transaction - 游戏用户消费接口

URL:

<code>/gbs/internalapi/gbs.transaction</code>

参数：

参数	说明	参数类型
userid	用户标志（CAS系统中的用户标志）	integer
currencyid	币种标志, 金元宝为 “11”，银元宝为 “12”。参见 附表-币种配置说明	integer
amount	消费金额, 可保留2位小数点, 如 10.04, 第三位将被四舍五入处理	float
memo	备注说明, 说明扣费用途. 如用户在进行购买药水, 备注说明：用户 XXX 购买 XX 瓶药水, 单价 XX, 折扣 XX	string

备注格式：

物品标识:物品数量;其他说明

- 物品标识，是游戏中物品的唯一标识，一般为数字。如：10001 红药水 10002 蓝药水
- 物品数量，消费购买的物品数，纯数字
- 其他说明，请用一句话简单描述下该用户消费的行为。如:用户XXX购买了10瓶红药水

特别说明

其他说明中如果出现分隔符号 “,” 或 “|”，请使用反斜杠进行转义。如果是不同物品消费，每个物品使用 “|” 分割。备注说明中的符号全部使用英文半角符号

样例：

单物品消费:

10001:10:用户XXX购买了10瓶红药水

多物品消费:
10001:10:用户XXX购买了10瓶红药水|10002:1:用户XXX购买了1瓶蓝药水

返回数据：

- 返回状态

返回状态	说明
0	消费成功
1	金额不足
2	缺少参数请求失败
3	消费失败, 内部错误, 如: 连接数据库失败等
4	指定币种不可用
5	缺少备注说明

- 成功返回数据字段说明

字段	说明	样例
asset	序列化后的交易后用户资产	{"11":"12967.00"}

样例：

成功样例
<pre>{"status":0,"data":{"11":"12967.00"},"error":null}</pre>
失败样例1
<pre>{"status":1,"data":null,"error": "\u5e10\u6237\u91d1\u989d\u4e0d\u8db3\u65e0\u6cd5\u7ee7\u7eed\u6d88\u8d39"}</pre>
失败样例2
<pre>{"status":2,"data":null,"error": "\u7f3a\u5c11\u53c2\u6570\u8bf7\u6c42\u5931\u8d25"}</pre>
失败样例3
<pre>{"status":3,"data":null,"error": "SQLSTATE[42000] [1049] Unknown database 'ice_gbs1'"}</pre>

(多方)交易

尚未实现.

测试说明

帐号充值：<http://bts0.playcool.com/bts/payment/gateway?paymentid=7®ionid=301>

测试帐号：icetest，密码 icetest123

申请帐号：<http://cas0.playcool.com/sign/up>

帐号充值完毕后，使用接口 gbs.getAsset 获取当前用户资产，确认无误后再进行 gbs.transaction 消费操作。再进行 gbs.getAsset 查询当前用户资产看是否消费正常。



帐号充值时，充值面额和游戏内的资产不相等。如充值 10 元金元宝可能为 100

附录

币种配置说明

币种字符标志、别名	币种标志	币种	说明
1	RMB	人民币	
3	USD	美元	
11	JYB	金元宝	游戏币
12	YYB	银元宝	游戏币

CAS-API

认证及调用流程

OAuth 认证接口参见 [CAS-OAuth 中央认证](#)

接口地址：

`http://cas.trac.cn/cas/Api`

获取用户信息

users.getLoggedInUser

获取当前登录用户(授权当前TOKEN的用户)信息.

参数:

参数	默认值	说明
fields	userid	获取字段, 多个以逗号分隔, 可选字段: 'userid', 'username', 'nickname', 'gender', 'ctime'

CAS-OAuth 中央认证

认证流程及访问资源流程

认证过程采用OAuth 1.0a <http://oauth.net/core/1.0a/>规范

CAS-API通过以下四个步骤来完成认证授权并访问或修改受限资源的流程

1. 获取未授权的Request Token
2. 请求用户授权Request Token
3. 使用授权后的Request Token换取Access Token
4. 使用 Access Token 访问或修改受保护资源

获取未授权的Request Token

通过访问以下 URL 获取未授权的 Request Token

```
http://platform/cas/OAuth/RequestToken
```

包含的主要参数

参数	意义
oauth_consumer_key	API Key
oauth_signature_method	签名方法，建议使用HMAC-SHA1
oauth_signature	签名值
oauth_timestamp	时间戳
oauth_nonce	单次值，随机字符串，防止重放攻击
oauth_callback	认证成功后浏览器会被重定向到形如http://callback?oauth_token=ab3cd9j4ks73hf7g的url，其中oauth_token为Request Token

本步骤用于签名的secret是API Key Secret

返回值包括未授权的Request Token和对应的Request Token Secret

请求用户授权Request Token

获得Request Token之后，需要请求用户授权该Request Token

你需要将浏览器跳转到如下URL（如果无法自动跳转，则需要提示用户手工跳转）。这会是一个CAS上的页面，提示用户授权给你的应用，以允

```
http://platform/cas/OAuth/AuthorizeToken
```

跳转后用户会看到请求授权的页面，用户可以选择同意或者拒绝授权

该请求包含如下两个参数

参数	意义
oauth_token	上一步中获得的Request Token

否则需要用户手工通知第三方应用以完成授权

使用授权后的Request Token换取Access Token

用户完成授权后，第三方应用可以通过访问如下url，将已授权的Request Token换取Access Token。Access Token将被用于访问或修改受限资源

```
http://platform/cas/OAuth/GetAccessToke
```

包含的主要参数

参数	意义
oauth_consumer_key	API Key
oauth_token	第一步中获得的Request Token
oauth_signature_method	签名方法
oauth_signature	签名值
oauth_timestamp	时间戳
oauth_nonce	单次值

本步骤用于签名的secret和token secret分别为，API Key Secret和Request Token Secret

返回值包括授权的Access Token，对应的Access Token Secret

使用Access Token调用API来访问或修改受保护资源

获得Access Token之后，你的应用就可以使用该Access Token调用 API来访问或修改受保护的资源

API地址

```
http://platform/cas/Api
```

在每次操作受保护资源时，请求都必须包含以下参数

参数	意义
oauth_consumer_key	API Key
oauth_token	Access Token
oauth_signature_method	签名方法
oauth_signature	签名值
oauth_timestamp	时间戳
oauth_nonce	单次值
method	调用的API方法，如users.getInfo获取用户信息
其它参数	API方法需要的其它参数

本步骤用于签名的secret和token secret分别为，API Key Secret和Access Token Secret

API文档参见 [CAS-API](#)

使用样例参考：

- /bts/configs/bts.dist.php
- /bts/configs/urmapping.dist.php
- /bts/controllers/LoginController.php # 使用中央认证登录控制器
- /bts/libraries/CAS.php # CAS 类库
- /Com/Authorization/Node.php # 节点站点身份认证类库

GAS-API 1.0



本文档为草稿, 意在需求沟通, 非最终文档!

名词解释:

- GAS - Game AuthServer

更新说明

2010-09-09: login接口新增错误状态10031帐号已冻结(需要到CAS解冻后再登录), 修正"已加入黑名单"状态码为10022

2010-09-01: getUserOnlineTime接口更新, 不再返回错误, 可放心调用, login2game接口不再返回在线时长信息及错误

2010-08-30: login接口加入areaid, 为区服号, 如tel1, 注意与其它接口同名参数的区别

2010-08-27: 改动很大! 更改了各接口的错误状态码, 使其更细致; 三界测试地址更改; login接口改为单纯的认证, 不再有其它功能;

增加login2game/logout4game接口, 分别用于进入具体游戏区-线路开始游戏和切出区-线路中止游戏; 废除timerStart/timerStop接口

2010-08-19: 增加开始计算在线时长timerStart和暂停计时timerStop接口

2010-08-13: 附录加入C语言示例

2010-08-03: login接口加入password_encrypted参数, 可以在客户端中将用户密码MD5后传递; 附录加入各游戏区对应认证服务器地址

2010-07-27: 返回数据格式加入服务器可能出错的警告

2010-07-26: login接口加入返回值token; 其它接口参数加入token

2010-07-23: login接口加入用户名可能更改的说明; logout接口参数中加入userid; getUserOnlineTime接口参数中加入userid

接口规范

接口采用OAuth 1.0a签名认证规范.

consumer key和secret请联系相关人员取得
access token 无须提供, 可直接调用相关接口
签名方法目前仅支持HMAC-SHA1.

接口地址

```
http(s)://_game_auth_server_/gas/api/_method_name_
```

例如, 三界奇缘0区的登录接口的地址为

```
http(s)://gas0.sj.playcoll.com/gas/api/login
```

返回数据格式

返回JSON格式数据, 字符集为:UTF-8.

数据	说明
status	返回状态, 0表示返回正常, -1表示系统出错, 其它大于0的值表示常规错误, 具体参见通用状态码表和各接口错误状态表
data	返回数据
error	错误描述, 当返回状态不为0时存在

样例:

```
{"status":10011,"data":null,"error":""}
```



如果返回不是JSON格式,无法json_decode

如果返回不是JSON格式, 可能是服务器故障, 请重试, 如果多次不正常, 说明真的服务器挂了, 赶快通知相关部门处理吧



通用状态码

0	返回正常
-1	系统错误/故障
20001	OAuth认证错误
20004	OAuth参数提供不完整, 必须提供的参数没有提供

基本接口

login - 验证用户密码及密保卡信息

说明: 此接口只做验证, 不做其它处理. 验证用户密码成功后, 如果用户没有设置密保卡, 则可以直接让用户登录区线并开始游戏, 如果用户设置了密保卡, 需要验证密保卡.

参数:

参数	说明
areaid	登录区服号, 如tel1
username	用户名
password	密码
password_encrypted	密码是否已MD5加密过, 1表示是, 0表示否
mbk_pos	密保卡三个坐标, 如A2C6E5 (可选)
mbk_pwd	密保卡坐标对应密码 (可选)
ip	用户IP (可选)
mac	用户网卡mac号 (可选)

返回数据：

如果用户名密码正确并且输入密保卡密码正确 (或没有设置密保卡) , status为0,data中为用户信息：

userid	用户ID
uuid	用户UUID
username	用户名
prevented	用户是否防沉迷用户, 1表示是, 0表示不是
token	TOKEN字串,用于在后续的操作中确定指定的人的会话, 有效期暂定为7天

错误状态:

10001	用户不存在
10011	密码不正确
10012	密保卡验证不正确
10021	不在白名单中
10022	已加入黑名单
10031	帐号已冻结(需要到CAS解冻后再登录)

样例1 (防沉迷用户)：

```
{ "status": 0, "data": { "userid": "10000", "username": "ahdong", "prevented": 1 }, "error": null }
```

样例2 (非防沉迷用户)：

```
{ "status": 0, "data": { "userid": "10000", "username": "ahdong", "prevented": 0 }, "error": null }
```

密保卡如果验证不正确或未输入密保卡, 即status=10012, data中为密保卡信息

mibaoka	为密保卡的矩阵大小, 如9x9 . 横坐标：1~9, 纵坐标：A~I, 样例图见附录.
---------	---

样例：

```
{ "status": 10012, "data": { "mibaoka": "9x9" }, "error": "" }
```



游戏登录流程建议方案

客户端让用户输入用户名、密码, 服务器端忽略mbk_pos/mbk_pwd参数调用login接口, 如果用户未设置密保卡, 则直接验证成功, 如果用户设置了密保卡, 则接口返回错误, status为10012,这时客户端需要提示用户输入密保卡信息, 服务器端加入mbk_pos/mbk_pwd参数再次调用login接口.

建议在登陆界面中, 用户名、密码输入框下加一个"我有密保卡"的按钮, 点击后展开提示用户输入指定坐标的密保卡, 这时候实际只要请求一次 login接口.



注意：用户名可能更改！

用户的username是有可能被更改的, 可能是情况是用户第一次登录时用户名为AAA, 而每二次登录时用户名为BBB, 但其实是同一个人, 所以游戏数据库关联要以userid来关联; 如果某一次登录时发现用户的username更改了,要更新为最新的username.



token很重要！

由于部分操作是异步的, 所以token对于确定一个人的身份很重要. 例如: 用户A登录并获得TOKEN1, 不久在另一个地方登录同一个区, 获得TOKEN2, 持有TOKEN1的登录将被强制下线; 因为强制下线是异步的, 所以如果没有指定token, 有可能持有TOKEN2的后一次登录被强制下线, 这显然是不应该的.

login2game - 用户登陆到具体区-线

说明：登录到具体的区线开始游戏并累计时长.

参数：

参数	说明
userid	用户ID, auth接口返回
token	TOKEN字串,auth接口返回
areaid	登录区-线服号, 如tel1-01, 注意与login接口同名参数的区别
ip	用户IP（可选）
mac	用户网卡mac号（可选）

返回数据：

如果正常, status为0

样例：

```
{ "status": 0, "data": null, "error": null }
```

logout4game - 登出区-线服

说明: 退出当前区, 去选择别的区或线路, 或者离开游戏. 停止累计在线时长.

参数:

参数	说明
userid	用户ID
areaid	登出区-线服号, 如tel1-01, 注意与login接口同名参数的区别
token	认证时获得的TOKEN字串

返回数据：

如果登出成功, status为0

logout - 退出游戏

说明:退出游戏, 是用户关闭客户端, 不再进入别的区游戏. 此方法为可选方法, 可以不调用.

参数:

参数	说明
userid	用户ID
token	认证时获得的TOKEN字符串

返回数据：
如果退出成功, status为0

resetServer - 重置服务器用户SESSION

说明：
区-线服维护或宕机重启, 将线上所有用户强制下线

参数:

参数	说明
areaid	区线服号, 如tel1-01, 注意与login接口同名参数的区别

返回数据：
如果登出成功, status为0

getUserOnlineTime - 获取用户在线时长

说明：
用户在所有区的防沉迷在线及离线时长(以秒为单位), 仅对纳入防沉迷系统的用户有效
时长会根据防沉迷规则在离线时长满足的时候(累计离线5小时)清零

参数:

参数	说明
userid	用户ID
token	登录时获得的TOKEN字符串

如果用户是防沉迷用户, status为0,data中为相关信息：

onlinetime	累计在线时长
offlinetime	累计休息时长

样例：

```
{ "status":0, "data":{ "onlinetime":3600, "offlinetime":3600 }, "error":null }
```

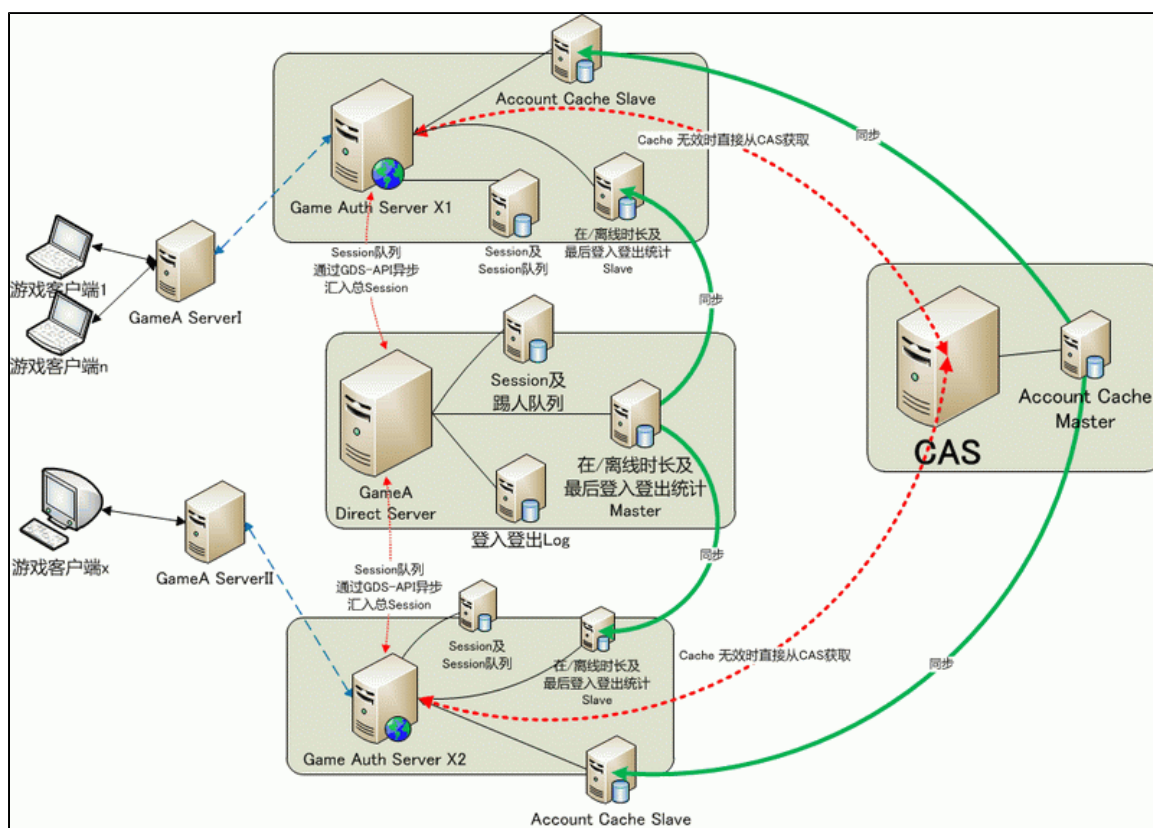
附录

密保卡样例

1001 0275 4568 7667

A	K6	Y9	BF	B2	XG	C3	MC	SM	SE
B	QY	D2	MJ	MP	AC	RX	8H	PE	YY
C	4Z	N8	ZA	X9	ER	DZ	GQ	6H	FT
D	SE	DW	4H	SG	DY	FR	ZB	6Y	BX
E	W6	KS	FZ	ZS	EP	SK	BB	JR	JG
F	BD	ZJ	SK	FW	AQ	JM	TC	EG	K8
G	XJ	GD	JJ	PK	4D	YN	FM	PE	XY
H	GY	QQ	BH	DE	TZ	AC	9M	KN	BN
I	MQ	TE	J9	MS	WG	BK	EW	4J	ZT
	1	2	3	4	5	6	7	8	9

GAS部署方案简图



各游戏区接口地址

游戏名称	区号	域名	IP
三界奇缘	1	gas0.sj.playcool.com	122.226.192.54

C语言示例

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <oauth.h>

/**
 * @param username
 * @param password
 * @param areaid
 */
void gaslogin(char *username, char *password, char *areaid) {
    const char *gas_api_login_url = "http://gas1.sj.playcool.com/gas/api/login"; //API
    const char *api_key = "sj-"; //API KEY
    const char *api_secret = ""; //API SECRET

    char *req_url = NULL;
    char *reply;

    //API, :(
    char *url = (char*)malloc(strlen(gas_api_login_url)+strlen("?username=")+strlen(username)+strlen(
    "&password=")+strlen(password)+strlen("&areaid=")+strlen(areaid));
    sprintf(url, "%s%s%s%s%s", gas_api_login_url, "?username=", username, "&password=", password,
    "&areaid=", areaid);
    printf("URL:%s\n", url);

    //
    req_url = oauth_sign_url2(url, NULL, OA_HMAC, NULL, api_key, api_secret, NULL, NULL);
    printf("request URL:%s\n", req_url);

    //API
    reply = oauth_http_get(req_url, NULL);
    if (!reply)
        printf("HTTP request for an oauth request-token failed.\n");
    else {
        printf("Got it: %s\n", reply);
        //json_decode,
    }
}

int main(int argc, char **argv) {
    if (argc != 4) {
        printf("Usage: %s USERNAME PASSWORD AREAID!\n", argv[0]);
        printf("Example: %s ahdong 111111 tell\n", argv[0]);
    }

    //gaslogin("ahdong", "111111", "tell");
    gaslogin(argv[1],argv[2],argv[3]);
    return 0;
}

```

GDS-API

GDS-API 加入白名单及黑名单等

接口地址

```
http://__GDS__SERVER__/gds/BlackWhiteApi/__METHOD_NAME__
```


返回数据格式


返回JSON格式数据, 字符集为:UTF-8.

数据	说明
status	返回状态, 0表示返回正常, 其它表示不正常, 具体状态表示以具体接口为准
data	返回数据
error	错误描述

样例：

```
{ "status":1, "data":null, "error":"" }
```

 如果返回不是JSON格式,无法json_decode

如果返回不是JSON格式, 可能是服务器故障, 请重试, 如果多次不正常, 说明真的服务器挂了, 赶快通知相关部门处理吧

基本接口

加入白名单 addWhite

参数	说明
userid	用户ID
areaid	区服ID, 全区以*表示

移除白名单removeWhite

参数	说明
userid	用户ID (两个参数必须提供一个)
areaid	区服ID, 全区以*表示

加入黑名单 addBlack

参数	说明
userid	用户ID
areaid	区服ID, 全区以*表示

移除黑名单removeBlack

参数	说明
userid	用户ID (两个参数必须提供一个)
areaid	区服ID, 全区以*表示