

# BITGAMEX 小游戏接入文档

## 目录

1. 版本日志 .....	5
2. 交互示意 .....	6
3. 结算脚本函数 .....	6
4. 登录接口 .....	7
4.1. 接口说明 .....	7
4.2. 接口参数 .....	7
4.3. 响应结果 .....	8
4.4. 特殊说明 .....	9
5. 修改密码接口 .....	10
5.1. 接口说明 .....	10
5.2. 接口参数 .....	10
5.3. 响应结果 .....	10
5.4. 特殊说明 .....	11
6. 绑定账号接口 .....	11
6.1. 接口说明 .....	11
6.2. 接口参数 .....	11
6.3. 响应结果 .....	12
6.4. 特殊说明 .....	13
7. 切换账号接口 .....	13
7.1. 接口说明 .....	13
7.2. 接口参数 .....	13
7.3. 响应结果 .....	14
7.4. 特殊说明 .....	15
8. 存盘数据获取接口 .....	15
8.1. 接口说明 .....	15
8.2. 接口参数 .....	15
8.3. 响应结果 .....	16
9. 存盘接口 .....	16

9.1. 接口说明 .....	16
9.2. 接口参数 .....	17
9.3. 响应结果 .....	17
9.4. 特殊说明 .....	18
10. 获取待领游戏币列表 .....	18
10.1. 接口说明 .....	18
10.2. 接口参数 .....	18
10.3. 响应结果 .....	19
10.4. 特殊说明 .....	19
11. 领取游戏币 .....	20
11.1. 接口说明 .....	20
11.2. 接口参数 .....	20
11.3. 响应结果 .....	20
11.4. 特殊说明 .....	21
12. 消耗游戏币 .....	21
12.1. 接口说明 .....	21
12.2. 接口参数 .....	22
12.3. 响应结果 .....	22
12.4. 特殊说明 .....	23
13. 转账游戏币给其他玩家的接口 .....	23
13.1. 接口说明 .....	23
13.2. 接口参数 .....	23
13.3. 响应结果 .....	24
13.4. 特殊说明 .....	25
14. 绑定 BIT.GAME 交易所账号的接口 .....	25
14.1. 接口说明 .....	25
14.2. 接口参数 .....	25
14.3. 响应结果 .....	26
14.4. 特殊说明 .....	26
15. 转账游戏币给绑定的交易所账号的接口 .....	27
15.1. 接口说明 .....	27

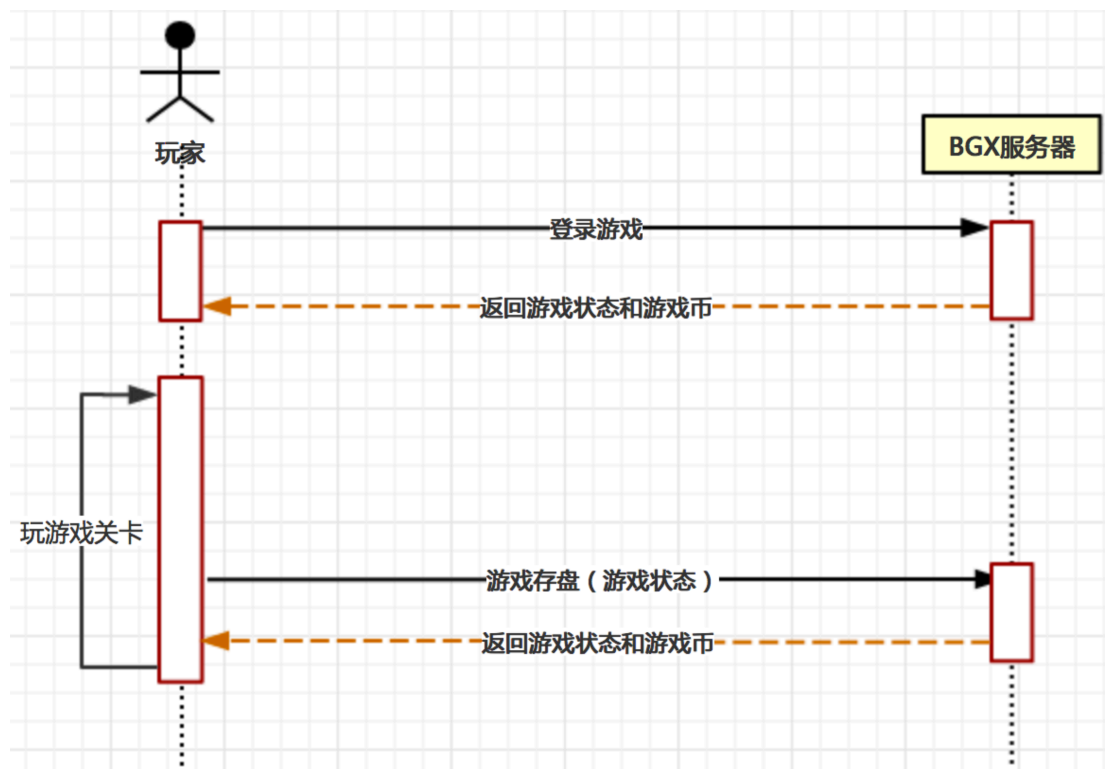
15.2. 接口参数 .....	27
15.3. 响应结果 .....	27
15.4. 特殊说明 .....	28
16. 绑定以太坊钱包地址的接口 .....	28
16.1. 接口说明 .....	28
16.2. 接口参数 .....	29
16.3. 响应结果 .....	29
16.4. 特殊说明 .....	30
17. 转账游戏币给绑定（或指定）的钱包的接口 .....	30
17.1. 接口说明 .....	30
17.2. 接口参数 .....	30
17.3. 响应结果 .....	31
17.4. 特殊说明 .....	32

## 1. 版本日志

### 修改记录

版本号	修改日期	变更说明	修改人	审核	备注
0.1	2018-3-2	起草	谷力		
0.2	2018-3-5	增加转账相关接口	谷力		
0.3	2018-3-30	增加领取游戏币接口	谷力		
0.4	2018-4-13	增加绑定和切换账号接口	谷力		
0.5	2018-4-24	增加用户名密码验证机制	谷力		

## 2. 交互示意



## 3. 结算脚本函数

为方便控制和修改游戏币的产出，每个关卡结束时游戏币的结算将放在 BGX 游戏服务器上计算，因此，游戏研发方需要根据游戏自身的关卡奖励逻辑提供一个 Lua 脚本函数 F 来计算玩家的游戏币增量，假设关卡开始时玩家游戏状态的数据结构为 S0，游戏币余额为 B0，结束时状态变化成 S，则：

游戏币增量 =  $F(S0, S)$

说明：最终的游戏币余额，BGX 服务器会在  $B0 + \text{增量}$  的基础上进行额外规则的调整。存盘接口的返回结果中将提供 F 的结果，方便联调。

## 4. 登录接口

### 4.1. 接口说明

使用场景：游戏客户端请求 BGX 游戏服务器登录某游戏

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=login\\_game](https://api.bitgamex.com/?a=login_game)

### 4.2. 接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
device_id	String(200)	是	设备id
user_name	String(50)	是	用户名
password	String(50)	是	用户密码（请上传MD5哈希值，我们不存原始密码）
time	Int(20)	是	时间戳
new_guest	Int(1)	否	1:新建一个guest用户，默认0
device_model	String(32)	否	设备型号
os_type	String(20)	否	操作系统类型
os_ver	String(100)	否	操作系统版本
lang	String(50)	否	语言
org_device_id	String(200)	否	最初的设备id
gc_id	String(40)	否	绑定ios game center账号id
gg_id	String(40)	否	绑定google账号id
fb_id	String(40)	否	绑定facebook账号id
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+device_id+

			+user_name+password+time+key) “+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定，比如： "BIT.GAME.X.8.8.8.8"
--	--	--	--

### 4.3. 响应结果

类型：json 结构

成功：

```
{
  "succ":1,
  "uid":Int(11),
  "token":String(50), 玩家在游戏中的用于存盘的令牌
  "game_data":String(2048), base64 编码，玩家该游戏的存盘数据
  "role_balance":字典，玩家角色的游戏币余额，“键”为游戏币的类型（比如BGX、ETH、BTC），“值”为对应游戏币的余额
  "user_balance":字典，玩家用户的游戏币余额，“键”为游戏币的类型（比如BGX、ETH、BTC），“值”为对应游戏币的余额
  "exchange_accid":String(100)
  "wallet_addr":String(50)
  "gc_id":String(40)
  "gg_id":String(40)
  "fb_id":String(40)
}
```

失败：

```
{
  "succ":0,
  "req":"login_game",
```



```

    "errno":错误编号
    "errmsg":错误描述
}

```

errno值	说明
-999	ip受限
-998	参数不全
-997	参数错误
-996	校验失败
-995	协议不支持
-994	异常抛出
-993	游戏未开放
-992	读缓存失败
-991	不存在rpc服务器
-990	金币不足
-989	请求失败
-988	请求超时
-987	未确定的错误
-986	无效的钱包地址
-985	无效的交易所账号

\* 以上为公共错误号定义，也为其他接口共享，后面不再重复列出。各接口特定的错误号会重复定义，结合错误返回结果中的 **req** 字段可唯一确定。

## 4.4. 特殊说明

该接口在玩家进入某游戏时调用，用于获取该游戏的存盘数据和令牌（后面存盘校验用）。对该接口的再次调用，将会默认玩家从上次进入的游戏中退出，上次的令牌失效。

## 5. 修改密码接口

### 5.1. 接口说明

使用场景：游戏客户端请求 BGX 游戏服务器修改登录密码。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=change\\_password](https://api.bitgamex.com/?a=change_password)

### 5.2. 接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
user_name	String(50)	是	用户名
old_password	String(50)	是	老的密码（密码的MD5）
new_password	String(50)	是	新的密码（密码的MD5）
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+user_name+old_password+new_password+time+key) “+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key和登录接口中一致。

### 5.3. 响应结果

类型: json 结构

成功:

```
{  
  "succ":1  
}
```

失败:

```
{  
  "succ":0,  
  "req":"change_password",  
  "errno":错误编号  
  "errmsg":错误描述  
}
```

errno值	说明
-1	原始密码验证错误
-2	新的密码不合规则
-3	

## 5.4. 特殊说明

无

## 6. 绑定账号接口

### 6.1. 接口说明

使用场景: 游戏客户端请求 BGX 游戏服务器根据所提供的绑定信息来绑定账号。

接口使用者: 游戏客户端以 HTTPS GET 方式请求。

请求地址: [https://api.bitgamex.com/?a=bind\\_user](https://api.bitgamex.com/?a=bind_user)

### 6.2. 接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
bind_type	String(10)	是	绑定数据类型: "gc_id", "gg_id", "fb_id"
bind_val	String(40)	是	绑定的数据
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+bind_type+bind_val+time+key) “+”表示直接将两个字符串连接, 拼接值不要有空格。用标准MD5算法进行加密。其中key和登录接口中一致。

## 6.3. 响应结果

类型: json 结构

成功:

```
{
  "succ":1,
  "bind_type":String(10)
  "bind_val":String(40)
}
```

失败:

```
{
  "succ":0,
  "req":"bind_user",
  "errno":错误编号
  "errmsg":错误描述
}
```

```
}
```

errno值	说明
-1	所提供的绑定数据已经被别的用户绑定了
-2	所提供的绑定数据已经被别的用户绑定了，且别的用户下有该游戏的角色数据可供恢复
-3	所指定的绑定类型已经绑定过别的数据了

## 6.4. 特殊说明

绑定信息需要在客户端进行验证。

绑定失败，当 `errno=-2` 时，`errmsg` 将会是之前绑定过的账号下可供恢复的角色数据，JSON 格式为：

```
{
  "bind_type":绑定数据类型: "gc_id", "gg_id", "fb_id"
  "bind_val":绑定的数据
  "uid":找到的用户 id,
  "create_time":角色创建时间戳,
  "login_time":角色上次登录时间戳,
  "game_data":游戏数据
}
```

## 7. 切换账号接口

### 7.1. 接口说明

使用场景：游戏客户端请求 BGX 游戏服务器根据所提供的绑定信息来切换账号。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=switch\\_user](https://api.bitgamex.com/?a=switch_user)

### 7.2. 接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
bind_type	String(10)	是	绑定数据类型: "gc_id", "gg_id", "fb_id"
bind_val	String(40)	是	绑定的数据
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+bind_type+bind_val+time+key) “+”表示直接将两个字符串连接, 拼接值不要有空格。用标准MD5算法进行加密。其中key和登录接口中一致。

## 7.3. 响应结果

类型: json 结构

成功:

```
{
  "succ":1,
  "uid":Int(11),
  "token":String(50), 玩家在游戏中用于存盘的令牌
  "game_data":String(2048), base64 编码, 玩家该游戏的存盘数据
  "role_balance":字典, 玩家角色的游戏币余额, “键” 为游戏币的类型 (比如 BGX、ETH、BTC), “值” 为对应游戏币的余额
  "user_balance":字典, 玩家用户的游戏币余额, “键” 为游戏币的类型 (比如 BGX、ETH、BTC), “值” 为对应游戏币的余额
  "exchange_accid":String(100)
  "wallet_addr":String(50)
```

```
"gc_id":String(40)
"gg_id":String(40)
"fb_id":String(40)
}
失败:
{
  "succ":0,
  "req":"switch_user",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明
-1	未找到所提供的绑定数据所对应的用户
-2	不能切换到当前用户
-3	切换到的账号在该游戏中没有创建角色

## 7.4. 特殊说明

绑定信息需要在客户端进行验证。

## 8. 存盘数据获取接口

### 8.1. 接口说明

使用场景：游戏客户端请求 BGX 游戏服务器下发玩家在游戏存盘数据，以及游戏币余额。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=get\\_game](https://api.bitgamex.com/?a=get_game)

### 8.2. 接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌

## 8.3. 响应结果

类型：json 结构

成功：

```
{
  "succ":1,
  "game_data":String(2048), base64 编码, 玩家该游戏的存盘数据
  "role_balance":字典, 玩家的游戏币余额, “键”为游戏币的类型（比如 BGX、ETH、BTC）, “值”为对应游戏币的余额
}
```

失败：

```
{
  "succ":0,
  "req":"get_game",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明

## 9. 存盘接口

### 9.1. 接口说明



使用场景：游戏客户端请求 BGX 游戏服务器存盘玩家的游戏状态数据，并获得新的游戏币余额。

接口使用者：游戏客户端以 HTTPS POST 方式请求。

请求地址：[https://api.bitgamex.com/?a=save\\_game](https://api.bitgamex.com/?a=save_game)

## 9.2. 接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
game_data	String(2048)	是	该游戏的状态数据
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+game_data+time+key) “+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key和登录接口中一致。

## 9.3. 响应结果

类型：json 结构

成功：

```
{
  "succ":1,
  "game_data":String(2048), base64 编码，玩家该游戏的存盘数据
  "role_balance":字典，玩家的游戏币余额，“键”为游戏币的类型（比如 BGX、ETH、BTC），“值”为对应游戏币的余额
  "f_res":Double，玩家的游戏币基础增量，方便联调检查
```

```

}
失败:
{
    "succ":0,
    "req":"save_game",
    "errno":错误编号
    "errmsg":错误描述
}

```

errno值	说明

## 9.4. 特殊说明

表单参数中的 time 字段用于存盘时间校验。

# 10. 获取待领游戏币列表

## 10.1.接口说明

使用场景：游戏客户端发消息给 BGX 游戏服务器请求待领的游戏币列表。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=get\\_coin\\_list\\_to\\_draw](https://api.bitgamex.com/?a=get_coin_list_to_draw)

## 10.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识

token	String(50)	是	在登录接口中返回的令牌
-------	------------	---	-------------

## 10.3.响应结果

类型：json 结构

成功：

```
{
  "succ":1,
  "coin_list":字典，“键”为对应的可领游戏币的 id，“值”为一个字典：键
  coin_type 对应可领游戏币的类型（比如 BGX、ETH、BTC），键 amount 对应可
  领游戏币的面值
}
```

失败：

```
{
  "succ":0,
  "req":"get_coin_list_to_draw",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明

## 10.4.特殊说明

玩家在参与游戏的过程中可以获得游戏币，但需要手动领取。

## 11. 领取游戏币

### 11.1.接口说明

使用场景：游戏客户端发消息给 BGX 游戏服务器请求领取所得的游戏币。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=draw\\_coin](https://api.bitgamex.com/?a=draw_coin)

### 11.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
coin_id	Int(11)	是	游戏币id
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+coin_id+time+key) “+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定，比如：“BIT.GAME.X.8.8.8.8”

### 11.3.响应结果

类型：json 结构

成功:

```
{
  "succ":1,
  "role_balance":字典, 玩家的游戏币余额, “键”为游戏币的类型(比如 BGX、ETH、BTC), “值”为对应游戏币的余额
}
```

失败:

```
{
  "succ":0,
  "req":"draw_coin",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明
-1	Coin Id不存在, 或已领过

## 11.4.特殊说明

玩家在参与游戏的过程中可以获得游戏币, 但需要手动领取。参数中指定的 Coin Id 是之前 get\_coin\_list\_to\_draw 接口所返回的。

## 12. 消耗游戏币

### 12.1.接口说明

使用场景: 游戏客户端发消息给 BGX 游戏服务器请求消耗一部分游戏币。

接口使用者: 游戏客户端以 HTTPS GET 方式请求。

请求地址: [https://api.bitgamex.com/?a=consume\\_coin](https://api.bitgamex.com/?a=consume_coin)

## 12.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
coin_type	String(20)	是	游戏币的类型（比如BGX、ETH、BTC）
amount	Double	是	消耗的游戏币数量
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+coin_type+amount+time+key) “+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定，比如：“BIT.GAME.X.8.8.8.8”

## 12.3.响应结果

类型：json 结构

成功：

```
{
  "succ":1,
  "role_balance":字典，玩家的游戏币余额，“键”为游戏币的类型（比如 BGX、ETH、BTC），“值”为对应游戏币的余额
}
```

失败：

```
{
  "succ":0,
  "req":"consume_coin",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明

## 12.4.特殊说明

无

## 13. 转账游戏币给其他玩家的接口

### 13.1.接口说明

使用场景：游戏客户端请求 BGX 游戏服务器将自己的一部分游戏币转账给另一玩家。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=transfer\\_coin\\_in\\_game](https://api.bitgamex.com/?a=transfer_coin_in_game)

### 13.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌

dst_uid	Int(11)	是	目标用户id
coin_type	String(20)	是	币种: BGX, BTC, ETH, ...
amount	Double	是	转账的游戏币数量
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+dst_ uid+coin_type+amount+time+ key) “+”表示直接将两个字符串连接, 拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定, 比如: "BIT.GAME.X.8.8.8.8"

### 13.3.响应结果

类型: json 结构

成功:

```
{
  "succ":1,
  "role_balance":字典, 玩家的游戏币余额, “键”为游戏币的类型(比如 BGX、ETH、BTC), “值”为对应游戏币的余额
}
```

失败:

```
{
  "succ":0,
  "req":"transfer_coin_in_game",
  "errno":错误编号
  "errmsg":错误描述
}
```



errno值	说明
-1	目标玩家错误（自己，或者不存在）

## 13.4.特殊说明

转账会收取一定的手续费。

## 14. 绑定 BIT.GAME 交易所账号的接口

### 14.1.接口说明

使用场景：游戏客户端请求 BGX 游戏服务器将指定的 BIT.GAME 交易所的账号绑定到自己的用户 id 上，方便以后转账到交易所。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=bind\\_exchange\\_accid](https://api.bitgamex.com/?a=bind_exchange_accid)

### 14.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
exchange_accid	String(100)	是	交易所账号id
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+exchange_accid+time+key)

			<p>“+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定，比如：“BIT.GAME.X.8.8.8.8”</p>
--	--	--	--

### 14.3.响应结果

类型：json 结构

成功：

```
{
  "succ":1,
  "exchange_accid":String(100), 指定的交易所账号 id
}
```

失败：

```
{
  "succ":0,
  "req":"bind_exchange_accid",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明

### 14.4.特殊说明

玩家自己对所指定的交易所账号的正确性负责。

如果再次调用此接口，将覆盖原来绑定的交易所账号。

## 15. 转账游戏币给绑定的交易所账号的接口

### 15.1.接口说明

使用场景：游戏客户端请求 BGX 游戏服务器将自己的一部分游戏币转账给绑定的交易所账号。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=transfer\\_coin\\_to\\_exchange](https://api.bitgamex.com/?a=transfer_coin_to_exchange)

### 15.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
amount	Double	是	转账的游戏币数量
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+amount+time+key) “+”表示直接将两个字符串连接，拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定，比如：“BIT.GAME.X.8.8.8.8”

### 15.3.响应结果

类型：json 结构

成功：

```
{  
  "succ":1,  
  "role_balance":Double, 玩家在游戏中的游戏币余额  
  "exchange_balance":Double, 玩家绑定的交易所账号中的游戏币余额  
}
```

失败：

```
{  
  "succ":0,  
  "req":"transfer_coin_to_exchange",  
  "errno":错误编号  
  "errmsg":错误描述  
}
```

errno值	说明

## 15.4.特殊说明

转账会收取一定的手续费。

## 16. 绑定以太坊钱包地址的接口

### 16.1.接口说明

使用场景：游戏客户端请求 BGX 游戏服务器将指定的以太坊钱包的地址绑定到自己的用户 id 上，方便以后转账到钱包。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址: [https://api.bitgamex.com/?a=bind\\_wallet](https://api.bitgamex.com/?a=bind_wallet)

## 16.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
wallet_addr	String(50)	是	以太坊钱包地址
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+wallet_addr+time+key) “+”表示直接将两个字符串连接, 拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定, 比如: "BIT.GAME.X.8.8.8.8"

## 16.3.响应结果

类型: json 结构

成功:

```
{
  "succ":1,
  "wallet_addr":String(50), 指定的以太坊钱包地址
}
```

失败:

```
{
  "succ":0,
```

```

    "req": "bind_wallet",
    "errno": 错误编号
    "errmsg": 错误描述
}

```

errno值	说明

## 16.4.特殊说明

玩家自己对所指定的以太坊钱包地址的正确性负责。

如果再次调用此接口，将覆盖原来绑定的钱包地址。

## 17. 转账游戏币给绑定（或指定）的钱包的接口

### 17.1.接口说明

使用场景：游戏客户端请求 BGX 游戏服务器将自己的一部分游戏币转账给绑定的钱包。

接口使用者：游戏客户端以 HTTPS GET 方式请求。

请求地址：[https://api.bitgamex.com/?a=transfer\\_coin\\_to\\_wallet](https://api.bitgamex.com/?a=transfer_coin_to_wallet)

### 17.2.接口参数

参数	字段类型	必填	说明
uid	Int(11)	是	用户id
game_id	Int(11)	是	游戏的标识
token	String(50)	是	在登录接口中返回的令牌
amount	Double	是	转账的游戏币数量

wallet_addr	String(50)	是	指定的以太坊钱包地址(不指定填空, 但字段需要)
time	Int(20)	是	时间戳
sign	String(32)	是	MD5签名 sign = md5(uid+game_id+token+amount+wallet_addr+time+key) “+”表示直接将两个字符串连接, 拼接值不要有空格。用标准MD5算法进行加密。其中key将在对接过程中由双方一同确定, 比如: "BIT.GAME.X.8.8.8.8"

## 17.3.响应结果

类型: json 结构

成功:

```
{
  "succ":1,
  "role_balance":Double, 玩家在游戏中的游戏币余额
  "exchange_balance":Double, 玩家绑定的交易所账号中的游戏币余额
}
```

失败:

```
{
  "succ":0,
  "req":"transfer_coin_to_wallet",
  "errno":错误编号
  "errmsg":错误描述
}
```

errno值	说明


## 17.4.特殊说明

转账会收取双重的手续费（1. 到交易所 2. 交易所到钱包）。