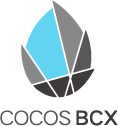
标准区块链产品

游戏行业定制版1.0.4.1

互操作接口使用说明

cocos专版

*2018年12月18日*



目录

[1 WEB容器的互操作接口概述 5](#_Toc530994723)

[1.1 类库引用说明 5](#_Toc530994724)

[1.1.1 引入方式 5](#_Toc530994725)

[1.1.2 实例化类库对象 5](#_Toc530994726)

[1.2 API参数统一说明 5](#_Toc530994727)

[1.3 状态码 7](#_Toc530994728)

[2 区块链系统的互操作API 14](#_Toc530994729)

[2.1 钱包模式 14](#_Toc530994730)

[2.1.1 创建账户 14](#_Toc530994731)

[2.1.2 备份钱包 14](#_Toc530994732)

[2.1.3 加载备份钱包文件 14](#_Toc530994733)

[2.1.4 从备份文件恢复钱包 15](#_Toc530994734)

[2.1.5 导入私钥 15](#_Toc530994735)

[2.1.6 删除钱包 15](#_Toc530994736)

[2.1.7 获取钱包账户列表 16](#_Toc530994737)

[2.1.8 切换账户 16](#_Toc530994738)

[2.1.9 解锁账户 16](#_Toc530994739)

[2.1.10 锁定账户 17](#_Toc530994740)

[2.2 账户模式 17](#_Toc530994741)

[2.2.1 创建账户 17](#_Toc530994742)

[2.2.2 账户登录 17](#_Toc530994743)

[2.2.3 私钥登录 18](#_Toc530994744)

[2.2.4 退出登录 18](#_Toc530994745)

[2.2.5 修改密码 18](#_Toc530994746)

[2.2.6 获取当前账户信息 18](#_Toc530994747)

[2.3 账户操作 19](#_Toc530994748)

[2.3.1 升级成为终身会员账户 19](#_Toc530994749)

[2.3.2 导出用户私钥 19](#_Toc530994750)

[2.3.3 查询账户记录 19](#_Toc530994751)

[2.3.4 订阅用户操作记录变更 19](#_Toc530994752)

[2.3.5 查询账户信息 20](#_Toc530994753)

[2.4 数据结构 20](#_Toc530994754)

[2.5 代币操作接口 20](#_Toc530994755)

[2.5.1 代币资产转移 20](#_Toc530994756)

[2.5.2 创建资产 21](#_Toc530994757)

[2.5.3 更新资产 21](#_Toc530994758)

[2.5.4 资产销毁 22](#_Toc530994759)

[2.5.5 资产发行 22](#_Toc530994760)

[2.5.6 注资资产手续费池 22](#_Toc530994761)

[2.5.7 领取资产手续费 23](#_Toc530994762)

[2.5.8 查询链上发行的资产 23](#_Toc530994763)

[2.5.9 查询账户指定资产余额 23](#_Toc530994764)

[2.5.10 查询账户所有资产余额列表 24](#_Toc530994765)

[2.6 NH资产操作 24](#_Toc530994766)

[2.6.1 注册开发者 24](#_Toc530994767)

[2.6.2 创建世界观 24](#_Toc530994768)

[2.6.3 创造NH资产 24](#_Toc530994769)

[2.6.4 删除NH资产 25](#_Toc530994770)

[2.6.5 转移NH资产 25](#_Toc530994771)

[2.6.6 提议关联世界观 25](#_Toc530994772)

[2.6.7 批准关联世界观的提议 25](#_Toc530994773)

[2.6.8 获取当前用户收到的提议 26](#_Toc530994774)

[2.7 NH资产买卖接口 27](#_Toc530994775)

[2.7.1 创建NH资产出售单 27](#_Toc530994776)

[2.7.2 购买订单NH资产 27](#_Toc530994777)

[2.7.3 取消NH资产出售单 27](#_Toc530994778)

[2.8 NH资产查询类接口 28](#_Toc530994779)

[2.8.1 查询全网用户NH资产售卖订单 28](#_Toc530994780)

[2.8.2 查询指定用户的NH资产售卖订单 28](#_Toc530994781)

[2.8.3 查询账户下所拥有的道具NH资产 28](#_Toc530994782)

[2.8.4 查询开发者所关联的世界观 29](#_Toc530994783)

[2.8.5 查询开发者创建的NH资产 29](#_Toc530994784)

[2.8.6 查询NH资产详细信息 29](#_Toc530994785)

[2.8.7 查询世界观详细信息 29](#_Toc530994786)

[2.9 节点投票 30](#_Toc530994787)

[2.9.1 查询节点投票信息数据 30](#_Toc530994788)

[2.9.2 用户提交投票信息 30](#_Toc530994789)

[2.10 区块链浏览器类接口 30](#_Toc530994790)

[2.10.1 查询区块 30](#_Toc530994791)

[2.10.2 查询交易 31](#_Toc530994792)

[2.10.3 订阅区块 31](#_Toc530994793)

[2.10.4 订阅区块链交易 31](#_Toc530994794)

[2.10.5 查看节点出块信息 31](#_Toc530994795)

[2.10.6 查看账户节点出块奖励 31](#_Toc530994796)

[2.10.7 领取节点出块奖励 32](#_Toc530994797)

[2.11 API服务器节点相关接口 32](#_Toc530994798)

[2.11.1 查看API服务器节点列表 32](#_Toc530994799)

[2.11.2 连接API服务器节点 32](#_Toc530994800)

[2.11.3 删除API服务器节点 33](#_Toc530994801)

[2.11.4 监听与API服务器节点的连接状态变化 33](#_Toc530994802)

[2.12 合约 33](#_Toc530994803)

[2.12.1 一键生成私钥/公钥（随机生成） 33](#_Toc530994804)

[2.12.2 合约创建 33](#_Toc530994805)

[2.12.3 合约更新 34](#_Toc530994806)

[2.12.4 合约调用 34](#_Toc530994807)

[2.12.5 查询合约信息 35](#_Toc530994808)

[2.12.6 查询账户合约数据 35](#_Toc530994809)

[2.13 其他 35](#_Toc530994810)

[2.13.1 获取交易类型基础手续费 35](#_Toc530994811)

[3 游戏资产OTC设计简述 37](#_Toc530994812)

[3.1 简易OTC设计思路 37](#_Toc530994813)

# WEB容器的互操作接口概述

## 类库引用说明

### 引入方式

<script type="text/javascript" src="bcx.min.js"></script>

### 实例化类库对象

bcx对象为类与区块链交互入口，bcx对象实例化调用方式如下：

Var config={

api\_node:{//api服务器节点

url:”ws://XXXXXXXXX”,//节点websocket地址

name:”XXXX”//节点名称,开发者自定义名称

},

ws\_node\_list:[

{url:"ws://xxxxxxx",name:"xxxxx"}

]//API服务器节点列表

faucet\_url:"http://\*\*\*.\*\*\*.\*\*\*.\*\*\*:\*\*\*\*", //注册入口

networks:[{

core\_asset:"\*\*\*",//核心资产符号

chain\_id:"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

}],

auto\_reconnect:false,//当RPC断开时是否自动连接，默认为true

app\_keys:["\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"]//合约授权，不进行合约授权，则不用配置此选项

}

Var bcx=new BCX(config)

## API参数统一说明

**API不做特殊说明均有一个可选参数callback：**

result为Object对象,结构为{code:0,message:””},

code=1的时候表示成功，无message状态描述,

code!=1时, 意味执行失败，message为失败状态描述，

**API不做特殊说明均只有一个参数，该参数为一个对象，对象包含所有相关参数，其中也包含callback**

调用示例：Var options={

callback:function(res){}

}

bcx.getPrivateKey(options)

**除订阅类API，其他API在不传callback参数情况下均返回promise对象**

调用实例

//转账

bcx.transferAsset({

to:test2,

amount:1

assetId:"1.3.0",

memo:"",

onlyGetFee:false

}).then(res=>{

console.info('transferAsset res',res);

})

**API的参数类型不做特殊说明均为字符串**

**API的参数不做特殊说明均不能为空，callback可选参数**

**查询类API的callback返回数据实例:{status:1,data:[]}**

**非查询类API的callback返回数据会多一个数据字段trxData,值为一个对象**

示例trxData:{

block\_num:112260,//区块高度

trx\_id:"c34021555e01e846ade1e119e2060a60eb514309"//交易ID

}

**非查询类API如果涉及到关联ID业务(如创建道具产生道具ID)的callback返回数据中将包含data对象**

//示例

data:{

real\_running\_time: 387//运行时间

result: "4.2.288"//关联业务id

}

**demo上的API调用返回数据在控制台会有打印**

## 状态码

|  |  |  |  |
| --- | --- | --- | --- |
| **code** | **message** | **说明** | **关联API** |
| 300 | Chain sync error, please check your system clock | 链同步错误，请检查您的系统时钟 | init |
| 301 | RPC connection failed. Please check your network | 连接RPC失败，请检查你的网络 | init |
| 1 | 无 | 操作成功 |  |
| 0 | failed | 操作失败，返回错误状态描述不固定，可直接提示res.message或统一提示为操作失败 |  |
| 1010 | Parameter is missing | 参数缺失 |  |
| 1011 | Parameter error | 参数错误 | QueryBlock, queryTXID |
| 1020 | The network is busy, please check your network connection | 网络繁忙，请检查你的网络连接 | passwordLogin等 |
| 1030 | Please enter the correct account name(/^[a-z]([a-z0-9\.-]){4,63}/$) | 请输入正确的账户名(正则/^[a-z]([a-z0-9\.-]){4,63}/$) | CreateAccountWithPassword |
| 1040 | \*\* not found | \*\* 不存在 | passwordLogin等 |
| 105 | wrong password | 密码错误 | passwordLogin,unlockAccount |
| 106 | The account is already unlocked | 账户已经处于解锁状态 | unlockAccount |
| 107 | Please import the private key | 请先导入私钥 | unlockAccount |
| 108 | User name or password error (please confirm that your account is registered in account mode, and the account registered in wallet mode cannot be logged in using account mode) | 用户名或密码错误(请确认你的账户是通过账户模式注册的，钱包模式注册的账户不能使用账户模式登录) | passwordLogin |
| 109 | Please enter the correct private key | 请输入正确的私钥 | keyLogin |
| 110 | The private key has no account information | 该私钥没有对应的账户信息 | keyLogin |
| 111 | Please login first | 请先登录 | changePassword |
| 112 | Must have owner permission to change the password, please confirm that you imported the ownerPrivateKey | 必须拥有owner权限才可以进行密码修改,请确认你导入的是ownerPrivateKey | changePassword |
| 113 | Please enter the correct original/temporary password | 请输入正确的原始密码/临时密码 | changePassword |
| 114 | Account is locked or not logged in. | 帐户被锁定或未登录 | changePasswor、transferAsset、getPrivateKey等 |
| 115 | There is no asset \*\* on block chain | 区块链上不存在资产\*\* | transferAsset等 |
| 116 | Account receivable does not exist | 收款方账户不存在 | transferAsset等 |
| 117 | The current asset precision is configured as \*,and the decimal cannot exceed \* | 当前资产精度配置为\*，小数点不能超过\* | transferAsset等 |
| 118 | Encrypt memo failed | 备注加密失败 | transferAsset等 |
| 119 | Expiry of the transaction | 交易过期 | transferAsset等 |
| 120 | Error fetching account record | 获取帐户记录错误 | queryUserOperations |
| 121 | block and transaction information cannot be found | 查询不到相关区块及交易信息 | queryBlockTXID |
| 122 | Parameter blockOrTXID is incorrect | 参数blockOrTXID不正确 | queryBlockTXID |
| 123 | Parameter account can not be empty | 参数account不能为空 | queryUserInfo |
| 124 | Receivables account name can not be empty | 收款方账户名不能为空 | transferAsset |
| 125 | Users do not own \* \* assets | 用户未拥有\*\*\*资产 | queryAccountBalances |
| 127 | No reward available | 没有可领取的奖励 | getVestingBalances,claimVestingBalance |
| 129 | Parameter 'memo' can not be empty | memo不能为空 | decodeMemo |
| 130 | Please enter the correct contract name(/^[a-z]([a-z0-9\.-]){4,63}$/) | 请输入正确的合约名称(正则/^[a-z]([a-z0-9\.-]){4,63}$/) | createContract |
| 131 | Parameter 'worldView' can not be empty | 世界观名称不能为空 | creatWorldView |
| 133 | Parameter 'toAccount' can not be empty | toAccount不能为空 | transferNHAsset |
| 135 | Please check parameter data type | 请检查参数数据类型 | creatNHAssetOrder |
| 136 | Parameter 'orderId' can not be empty | orderId不能为空 | cancelNHAssetOrder |
| 137 | Parameter 'NHAssetHashOrIds' can not be empty | NHAssetHashOrIds不能为空 | lookupNHAssets |
| 138 | Parameter 'url' can not be empty | 接入点地址不能为空 | switchAPINode |
| 139 | Node address must start with ws:// or wss:// | 节点地址必须以 ws:// 或 wss:// 开头 | addAPINode |
| 140 | API server node address already exists | API服务器节点地址已经存在 | switchAPINode |
| 141 | Please check the data in parameter NHAssets | 请检查参数NHAssets中的数据 | creatNHAsset批量创建NH |
| 142 | Please check the data type of parameter NHAssets | 请检查参数NHAssets的数据类型 | creatNHAsset批量创建NH |
| 144 | Your current batch creation / deletion / transfer number is \*, and batch operations can not exceed \* | 您当前批量 创建/删除/转移 NH资产数量为\*,批量操作数量不能超过\* | creatNHAsset、deleteNHAsset、transferNHAsset |
| 145 | \*\* contract not found | XX合约不存在 | callContractFunction，queryContract，queryAccountContractData |
| 146 | The account does not contain information about the contract | 账户没有该合约相关的信息 | queryAccountContractData |
| 147 | NHAsset do not exist | 非同质资产不存在 | queryNHAssets |
| 148 | Request timeout, please try to unlock the account or login the account | 请求超时,请尝试解锁账户或登录账户 | queryVotes |
| 149 | This wallet has already been imported | 此私钥已导入过钱包 | importPrivateKey |
| 150 | Key import error | 导入私钥失败 | importPrivateKey |
| 151 | File saving is not supported | 您的浏览器不支持文件保存 | backupDownload |
| 152 | Invalid backup to download conversion | 无效的备份下载转换 | backupDownload |
| 153 | Please unlock your wallet first | 请先解锁钱包 | importPrivateKey |
| 154 | Please restore your wallet first | 请先恢复你的钱包 | backupDownload、unlockAccount |
| 155 | Your browser may not support wallet file recovery | 浏览器不支持钱包文件恢复 | loadWalletFile |
| 156 | The wallet has been imported. Do not repeat import | 该钱包已经导入,请勿重复导入 | restoreWallet |
| 157 | Can't delete wallet, does not exist in index | 请求超时,请尝试解锁账户或登录账户 | deleteWallet |
| 158 | Imported Wallet core assets can not be \*\*\*, and it should be \*\*\* | 导入的钱包核心资产不能为\*\*\*，应为\*\*\* | restoreWallet |
| 159 | Account exists | 账户已存在 | createAccountWithWallet |
| 160 | You are not the creator of the Asset \*\*\*. | 你不是该资产的创建者 | issueAsset |
| 161 | Orders do not exist | 订单不存在 | fillNHAssetOrder |
| 162 | The asset already exists | 资产已存在 | createAsset |
| 163 | The wallet already exists. Please try importing the private key | 钱包已经存在,请尝试导入私钥 | restoreWallet |
| 164 | worldViews do not exist | 世界观不存在 | queryWorldViews |
| 165 | There is no wallet account information on the chain | 链上没有该钱包账户信息 | restoreWallet |
| 166 | The Wallet Chain ID does not match the current chain configuration information. The chain ID of the wallet is:\*\*\*\*\* | 该钱包链id与当前链配置信息不匹配，该钱包的链id为：\*\*\*\* | restoreWallet |
| 167 | The current contract version ID was not found | 当前合约版本id没有找到\* | queryContract |

# 区块链系统的互操作API

## 钱包模式

### 创建账户

方法：createAccountWithWallet

功能：钱包模式创建账户，钱包模式创建的账户不能用账户名密码登录。如果钱包模式已经存在账户，该操作会创建子账户，创建该子账户需要先成为终身会员账户

参数：

account：用户名

password：密码

callback ：回调函数

### 备份钱包

方法：backupDownload

功能：备份钱包，调用该API会生成一个钱包文件，自动下载

参数：

callback ：回调函数

### 加载备份钱包文件

方法：loadWalletFile

功能：对于web来说，file input绑定change事件，读取钱包文件

参数：

file：宿主环境若在web => file input 的 change事件触发返回的事件对象event.target,files[0]

callback ：回调函数

### 从备份文件恢复钱包

方法：restoreWallet

功能：备份钱包，调用该API会生成一个钱包文件，自动下载。该API调用后，钱包处于锁定状态。

参数：

password: 备份文件的钱包密码

callback ：回调函数

### 导入私钥

方法：importPrivateKey

功能：导入私钥到钱包

参数：

privateKey：明文私钥

password: 如果是已经创建钱包或恢复钱包，此时的密码为原来钱包的密码，否则是可以随意填写的临时密码

callback ：回调函数

### 删除钱包

方法：deleteWallet

功能：删除钱包，使用账户模式时，最好让用户先执行删除钱包

参数：

callback ：回调函数

### 获取钱包账户列表

方法：getAccounts

功能：获取钱包账户列表

参数：

直接返回数据格式示例:

"accounts": ["tom0002"],

"currentAccount": {

"userId": "1.2.20",

"isLocked": true,

"name": "tom0002"

}

### 切换账户

方法：setCurrentAccount

功能：钱包模式切换当前使用账户

参数：

account:要切换的账户

callback:

### 解锁账户

方法：unlockAccount

功能：导入私钥或钱包模式才可以使用此方法解锁账户

参数：

password:导入私钥时设置的临时密码

callback:

### 锁定账户

方法：lockAccount

功能：锁定账户

参数：callback

## 账户模式

### 创建账户

方法：createAccountWithPassword

功能：账户注册。如果账户模式已经有账户登录，该操作会创建子账户，创建该子账户需要操作账户为终身会员账户

参数：

account：用户名

password：密码

autoLogin: boolean类型，指定是否自动登录，默认值为false

callback ：回调函数

### 账户登录

方法：passwordLogin

功能：账户登录

参数：

account：用户名

password：密码

callback ：回调函数

### 私钥登录

方法：privateKeyLogin

功能：之前这个API只是过渡，严格上导入私钥只存在于钱包模式。现在为了兼容，此API还暂时留在这里,依然可用调用

参数：

privateKey：私钥

password：设置的临时密码

callback

### 退出登录

方法：logout

功能：该方法会清除用户相关缓存,其中包括清除加密后的密文key

参数：

callback

### 修改密码

方法：changePassword

功能：只有账户模式，才能修改密码；修改密码成功后,API将会自动调用退出登录。

参数：

oldPassword：旧密码

newPassword：新密码

callback：

### 获取当前账户信息

方法：getUserInfo

功能：当账户处于解锁状态，返回数据中将包含账户名name

参数：

无callback，直接返回

## 账户操作

### 升级成为终身会员账户

方法：upgradeAccount

功能：购买终身会员账户后，可以创建子账户，此操作需消耗一定的手续费

参数：

onlyGetFee：是否只获取此操作手续费

callback

### 导出用户私钥

方法：getPrivateKey

功能：获取用户Active PrivateKey,本秘钥可用于为账户所有花费行为签名

参数：

callback: 设置获取私钥成功后的回调函数，回调参数:result,为Object对象,

对象结构为：

{status:1,data:{activePrivateKey:”\*\*\*\*\*\*\*\*\*\*\*”}},其中：activePrivateKey为私钥串

### 查询账户记录

方法：queryUserOperations

功能：查询用户近期操作记录

参数：

account:账户名

limit:查询记录条数

callback:查看API统一参数说明

### 订阅用户操作记录变更

方法：subscribeToUserOperations

功能：订阅用户操作记录变更

参数：

account:账户名

callback:只要用户操作记录有变化，就调用此callback

### 查询账户信息

方法：queryAccountInfo

功能：账户信息中包含用户id用户名等信息

参数：

account:账户名

callback:查看API统一参数说明

## 数据结构

本产品支持自定义数据结构的链上存储。

对于游戏定制版而言，数据在链上的存储结构分为数据头、数据内容两部分。

数据头由游戏数字资产(assetId)、道具版本(itemVER)、道具唯一标识(itemId)组成，数据内容用于描述道具详细信息由制造者定义，通常为游戏角色、游戏装备等等游戏信息。

**需要注意的是，任何数据的写操作，都将在区块中留下记录。**

## 代币操作接口

### 代币资产转移

方法：transferAsset

功能：向目标对象发送代币

参数：

fromAccount:发款方账户名,不是发起提议，

toAccount: 收款方账号名

amount：发送的代币数量

assetId ：资产ID （如：1.3.0）或 代币符号（如：BTC）

memo: 转账备注

feeAssetId:支付手续费的代币资产符号

isPropose:是否发起提议

onlyGetFee（boolean）：是否只获取本次操作所需手续费

callback: 设置转账后的回调函数

### 创建资产

方法：createAsset

功能：创建token

参数：

assetId: 资产符号,正则^[.A-Z]+$

precision: 精度(小数位数)

maxSupply: 最大资产总量

coreExchangeRate(Object): {

quoteAmount:标价资产(即创建的代币，默认1),

baseAmount: 基准资产(即核心资产,默认1)

}

description: 资产描述，可不填

onlyGetFee: 设置只返回本次调用所需手续费

callback: 见统一API说明

### 更新资产

方法：updateAsset

功能：更新token

参数：

assetId: 资产符号,正则^[.A-Z]+$

maxSupply: 最大资产总量

newIssuer：更新发行人

coreExchangeRate(Object):{ 手续费汇率

quoteAmount:标价资产

baseAmount:基准资产

}

description: 资产描述，可不填

onlyGetFee: 设置只返回本次调用所需手续费

callback: 见统一API说明

### 资产销毁

方法：reserveAsset

功能：销毁代币资产

参数：

assetId：资产符号

amount：销毁数量

onlyGetFee：设置只返回本次调用所需手续费

callback

### 资产发行

方法：issueAsset

功能：代币资产token发行

参数：

toAccount：发行对象

amount：发行数量

assetId：资产符号

memo：备注消息，选填

onlyGetFee：设置只返回本次调用所需手续费

callback

### 注资资产手续费池

方法：assetFundFeePool

功能：所有网络手续费最终将使用核心资产代币进行支付。手续费资金池用来承担从 二级资产代币 转换为 核心资产代币 的费用，以便用户可以使用 二级资产代币 来支付手续费。如果资金池中余额用完，用户将无法继续使用 二级资产代币 支付手续费。目前支持使用二级资产代币作为手续费的API有“转账、投票、升级终身会员、资产发行”，后续会继续扩展

参数：

assetId：需要注资的二级资产代币符号

amount：注资核心资产代币数量

onlyGetFee：设置只返回本次调用所需手续费

callback

### 领取资产手续费

方法：assetClaimFees

功能：资产发行人可以在这里领取累积的资产手续费。

参数：

assetId：需要领取的二级资产代币符号

amount：二级资产代币数量

onlyGetFee：设置只返回本次调用所需手续费

callback

### 查询链上发行的资产

方法：queryAsset

功能：代币资产查询

参数：

assetId：资产符号,此参数若为空，则查询链上发行的所有资产

callback

### 查询账户指定资产余额

方法：queryAccountBalances

功能：获取用户对应的数字资产，如果assetId为空，则返回用户所有代币。

参数：

assetId：资产ID或代币符号，资产ID：数字代币的唯一代币标识ID（如：”1.3.0”），代币符号（如：”BTC”）

account: 用户名

callback：

### 查询账户所有资产余额列表

方法：queryAccountAllBalances

功能：查询用户拥有的所有资产列表，列表中包含资产对记账单位的换算值。当账户无任何资产余额将会返回余额为0的核心资产

参数：

unit：记账单位，将会根据手续费汇率或交易市场价格换算等价的该资产，资产ID或代币符号，资产ID：数字代币的唯一代币标识ID（如：”1.3.0”），代币符号（如：”BTC”）

account:用户名

callback：

## NH资产操作

### 注册开发者

方法：registerCreator

功能：将当前账户注册成为开发者

### 创建世界观

方法：creatWorldView

功能：创建支持的NH资产世界观，向区块链系统注册当前账号（通常为游戏的账号）支持的NH资产世界观

参数：

worldView：世界观名称,区分大小写;

### 创造NH资产

方法：creatNHAsset

功能：创建一个唯一的NH资产，具有唯一性。本接口仅限NH资产制造商（铁匠铺）使用。

参数：

assetId:当前NH资产交易时，适用的资产ID；

worldView：世界观；

baseDescribe：当前NH资产的具体内容描述，由制造者定义;

ownerAccount:指定NH资产拥有者(NH资产归属权账户，默认为NH资产创建者)

NHAssetsCount (Number):创建NH资产的数量，默认值为1，只有type为0即创建同一种NH资产生效

type:创建NH资产方式的类型：默认值为0，0是创建同一种NH资产，1是创建不同NH资产

NHAssets(Array):示例: [{"assetId":"1.3.0","worldView":"TEST","baseDescribe":"{name:\"预言家\"}","ownerAccount":"test2"},{"assetId":"1.3.0","worldView":"TEST","baseDescribe":"{name:\"女巫\"}","ownerAccount":"test2"},{"assetId":"1.3.0","worldView":"TEST","baseDescribe":"{name:\"猎人\"}","ownerAccount":"test2"}]

### 删除NH资产

方法：deleteNHAsset

功能：删除整条NH资产数据记录,通常在商品销毁时使用（**仅能由用户自己授权处理自己想要销毁的数据**）。

参数:

NHAssetIds (Array): NH资产实例的唯一标识ID;示例：[3.2.195,3.2.194]

### 转移NH资产

方法：transferNHAsset

功能：用户可以将自己的NH资产转移到另外一个用户

参数:

toAccount:转移NH资产的目标用户名

NHAssetIds（Array）:多个NH资产id组成的数组，示例:[3.2.332,3.2.333]。

### 提议**关联**世界观

方法：proposeRelateWorldView

功能：提议关联到某一个世界观，需要该世界观的创建人审批

参数:

worldView:需要关联的世界观名

viewOwner:需要关联的世界观创建者

### 批准关联世界观的提议

方法：approvalProposal

功能：批准其他用户关联自己的世界观的提议

参数:

proposalId:提议Id

### 获取当前用户收到的提议

方法：getAccountProposals

功能：获取当前操作用户收到的提议

## NH资产买卖接口

### 创建NH资产出售单

方法：creatNHAssetOrder

功能：卖出NH资产（在交易前可调用queryAccountGameItems函数，列举用户NH资产，以便用户选着卖出）

参数：

otcAccount：OTC交易平台账户，用于收取挂单费用；（OTC平台填写）

orderFee：挂单费用，用户向OTC平台账户支付的挂单费用；（OTC平台填写）

NHAssetId：NH资产实例的唯一标识ID; （用户填写）

price：商品挂单价格；（用户填写）

priceAssetId：商品挂单价格所使用的代币种类；（用户填写）

expiration:挂单时间，如60\*60=3600(秒)，为1小时

memo: 挂单备注信息；

callback: 设置执行挂单卖出后的回调函数

### 购买订单NH资产

方法：fillNHAssetOrder

功能：买入NH资产，支付购买游戏装备的代币费用，同时修改用户拥有的商品数据。该操作是一个多步合成的原子操作，在支付费用的同时完成用户账户NH资产数据的更新，如果支付动作或账户商品数据更新动作中某一个动作不被主链区块认可，则整个交易将被回滚，避免异常交易。

参数：

orderId:订单ID

callback

### 取消NH资产出售单

方法：cancelNHAssetOrder

功能：取消NH资产挂卖订单

参数：

orderId：订单ID

callback

## NH资产查询类接口

### 查询全网用户NH资产售卖订单

方法：queryNHAssetOrders

功能：查询全网用户NH资产的售卖订单

参数：

assetSymbolsOrIds(array）:资产符号或id筛选条件

worldViews (array):版本名称或版本id筛选条件

pageSize：页容量

page:页数

### 查询指定用户的NH资产售卖订单

方法：queryAccountNHAssetOrders

功能：查询指定用户的NH资产售卖订单

参数：

account:查询账户名或账户Id

pageSize：页容量

page:页数

callback：

：

### 查询账户下所拥有的道具NH资产

方法：queryAccountNHAssets

功能：读取当前用户账户下所有可在对应游戏中使用的NH资产

参数：

account:账户名或账户id

worldViews (array):世界观名集合

page:页数

pageSize:页容量，每页的数据条数

callback：返回值示例{status:1,data:[],total:0}

### 查询开发者所关联的世界观

方法：queryNHCreator

功能：查询开发者所关联的世界观

参数：

account:账户名或账户ID

callback

### 查询开发者创建的NH资产

方法：queryNHCreator

功能：查询开发者所创建的世界观

参数：

account:账户名或账户ID

callback

### 查询NH资产详细信息

方法：queryNHAssets

功能：查询NH资产详细信息

参数：

NHAssetHashOrIds:NH资产id或hash

callback

### 查询世界观详细信息

方法：lookupWorldViews

功能：查询世界观详细信息

参数：

worldViewNameOrIds (array):世界观名称或id

callback

## 节点投票

### 查询节点投票信息数据

方法：queryVotes

功能：查询节点投票信息数据

参数：

callback

### 用户提交投票信息

方法：publishVotes

功能：保存的时候设置了代理账户，用户投票信息将统一跟随代理账户

参数：

witnessesIds（array）:节点账户id集合,查询节点投票信息数据中会有每个节点的账户ID

proxyAccount:代理账户名

callback

## 区块链浏览器类接口

### 查询区块

方法：queryBlock

功能：通过区块高度查询区块信息

参数：

block: 区块高度

### 查询交易

方法：queryTransaction

功能：通过交易id（即交易hash）查询交易信息

参数：

transactionId: 交易id

### 订阅区块

方法：subscribeToBlocks

功能：监听实时出块信息

参数：

callback

### 订阅区块链交易

方法：subscribeToChainTranscation

功能：监听区块链全网发生的交易

参数：

callback

### 查看节点出块信息

方法：lookupWitnessesForExplorer

功能：这里重点是参照demo解析节点出块信息数据

参数：

callback

### 查看账户节点出块奖励

方法：lookupBlockRewards

功能： 参照demo解析数据

参数：

Callback

### 领取节点出块奖励

方法：claimVestingBalance

功能： 领取节点出块奖励

参数：

id：奖励id,查询节点出块奖励中会有这个id

callback

## API服务器节点相关接口

### 查看API服务器节点列表

方法：lookupWSNodeList

功能：查看API服务器节点列表信息

参数：

callback

### 连接API服务器节点

方法：switchAPINode

功能：切换节点

参数：

url：API服务器节点地址，此地址必须是API服务器节点列表中的websoket地址

callback

#### 添加新的API服务器节点

方法：addAPINode

功能：添加新节点

参数：

name:新节点名称

url：API服务器节点websoket地址

callback

### 删除API服务器节点

方法：deleteAPINode

功能：删除节点

参数：

url：API服务器节点websoket地址

callback

### 监听与API服务器节点的连接状态变化

方法：subscribeToRpcConnectionStatus

功能：监听rpc连接状态变化

参数：

callback：回调会返回状态status: closed：rpc连接关闭,error：rpc连接错误，realopen：rpc连接成功。

## 合约

### 一键生成私钥/公钥（随机生成）

方法：generateKeys

功能：随机生成一对公私钥，创建带有权限的合约会用到,生成的私钥用于API初始化对合约授权，没有回调，直接返回

### 合约创建

方法：createContract

功能：创建智能合约，如果要对合约设置权限，创建合约时得加入特定的lua代码，并调用合约函数set\_permissions\_flag => 合约权限代码： function my\_change\_contract\_authority( publickey) assert(is\_owner()) change\_contract\_authority( publickey) end function set\_permissions\_flag(flag) assert(is\_owner()) set\_permissions\_flag(flag) end

参数：

authority：合约权限(一对公私钥中的公钥publicKey)，开发者在使用API初始化的时候，可以配置私钥，配置了该公钥对应的私钥才可以调用合约。

name: 合约名称,规则/^[a-z][a-z0-9\.-]{4,}/

data: 合约lua代码

onlyGetFee: 设置只返回本次操作所需手续费

callback: 见统一API说明

### 合约更新

方法：updateContract

功能：更新合约代码

参数：

nameOrId合约名称或Id，示例：contract.test02

data: 合约lua代码

onlyGetFee: 设置只返回本次操作所需手续费

### 合约调用

方法：callContractFunction

功能：调用合约函数接口

参数：

nameOrId：合约名称或Id，示例：contract.test02

functionName:合约里的函数名称，my\_nht\_describe\_change （修改道具属性）

valueList(array)： 调用合约函数的参数列表,示例：[‘4.2.2’,’name’,’女巫’]

runtime: 运行合约函数的时间(单位毫秒),默认为5

onlyGetFee: 设置只返回本次操作所需手续费 默认为false

callback

### 查询合约信息

方法：queryContract

功能：查询合约信息数据

参数：

nameOrId：合约名字或Id

callback

### 查询账户合约数据

方法：queryAccountContractData

功能：查询账户合约里产生数据

参数：

userNameOrId：账户名或Id

contractNameOrId: 合约名字或Id

callback

## 其他

### 获取交易类型基础手续费

方法：getTransactionBaseFee

功能： 获取交易类型基础手续费

参数：

transactionType：交易类型,示例transfer

feeAssetId:选择支付手续费的代币类型资产符号或ID

callback：见统一API说明

transactionType列表

|  |  |
| --- | --- |
| **transactionType** | **对应相关API** |
| transfer | transferAsset |
| limit\_order\_create | createLimitOrder |
| limit\_order\_cancel | cancelLimitOrder |
| account\_create | createAccountWithPassword，createAccountWithWallet |
| account\_update | changePassword |
| account\_upgrade | upgradeAccount |
| asset\_create | createAsset |
| asset\_issue | issueAsset |
| proposal\_update | submitProposal |
| vesting\_balance\_withdraw | claimVestingBalance |
| contract\_create | createContract |
| call\_contract\_function | callContractFunction |
| register\_creator | createGameVersion |
| creat\_world\_view | createGameItem |
| propose\_relate\_game\_version | proposeRelateGameVersion |
| creat\_nh\_asset | createGameItem |
| updata\_nh\_asset | updateGameItem |
| delete\_nh\_asset | deleteGameItem |
| transfer\_nh\_asset | transferGameItem |
| creat\_nh\_asset\_order | creatGameItemOrder |
| cancel\_nh\_asset\_order | cancelGameItemOrder |
| fill\_nh\_asset\_order | fillGameItemOrder |

# 游戏资产OTC设计简述

## 简易OTC设计思路



图4-1-1 OTC交易流程图

道具交易市场与链上游戏统一使用链上开放的交互接口，其中基础功能，如查询、读取区块数据等可以直接连接节点进行，但OTC的交易撮合部分功能则需要服务器进行辅助撮合。

**在初期阶段，建议不在OTC中设计自动撮合过程，而是由买家手动选择某个卖单来执行成交。**

下文是道具交易市场的基本流程设计，基本流程如图4-1-1所示。

1. 用户授权登录：

用户进入交易市场登录引导，绑定现有账户或者创建新账户，进入道具交易市场；

2. 读取当前OTC交易市场已支持的游戏数据内容：

为了提供最佳的兼容性和可定制特性，链上不会对具体封装的游戏数据进行解析或作出结构要求，以便游戏厂商更灵活的设计自己的链上游戏存储结构，这些游戏数字自查可通过游戏客户端以及资产市场解析，为游戏内容的丰富化提供强有力的支撑；

3. 游戏道具挂单和售卖：

玩家用户可选择将自己的道具以挂单方式挂入交易市场，符合买卖价格的买单和卖单将被系统自动撮合，交易内容不局限于游戏内货币，也涵盖道具、装备、游戏数据等。

完成挂单后，交易市场调用链上数据操作接口，修改用户拥有的游戏道具的数据，并在交易市场中心服务器上生成挂卖信息（出售人、价格、装备等等）。

4. 游戏内容变更：

当交易订单完成，即游戏内容买入动作顺利完成时，则OTC游戏市场调用游戏数据内容购买接口，向出售方支付代币，同时更新自身链上游戏数据，完成游戏内容数据的交付，动作完成后OTC交易市场修改挂单信息状态为已完成。