

多链钱包 支持XuperChain、以太坊

XuperOS/开放网络 前期准备

注册账号以及获取私钥

访问 <https://xuper.baidu.com> 百度开放网络

1. 使用平台前，请先注册百度账号，若您已有百度账号，登录即可
2. 点击右上角进入「工作台」



3. 进入设置账户安全码页，安全码作为交易密码，请务必牢记。平台无法提供安全码找回功能。设置完成后，点击「下一步」

4. 进入记录超级链账户页，请务必按照页面指引，下载账户私密钥、记下助记词，一旦遗失，会导致无法找回账户。平台无法提供找回私密钥、助记词功能。点击「进入工作台」，进入工作台页后，即注册平台账号成功！

进入控制台

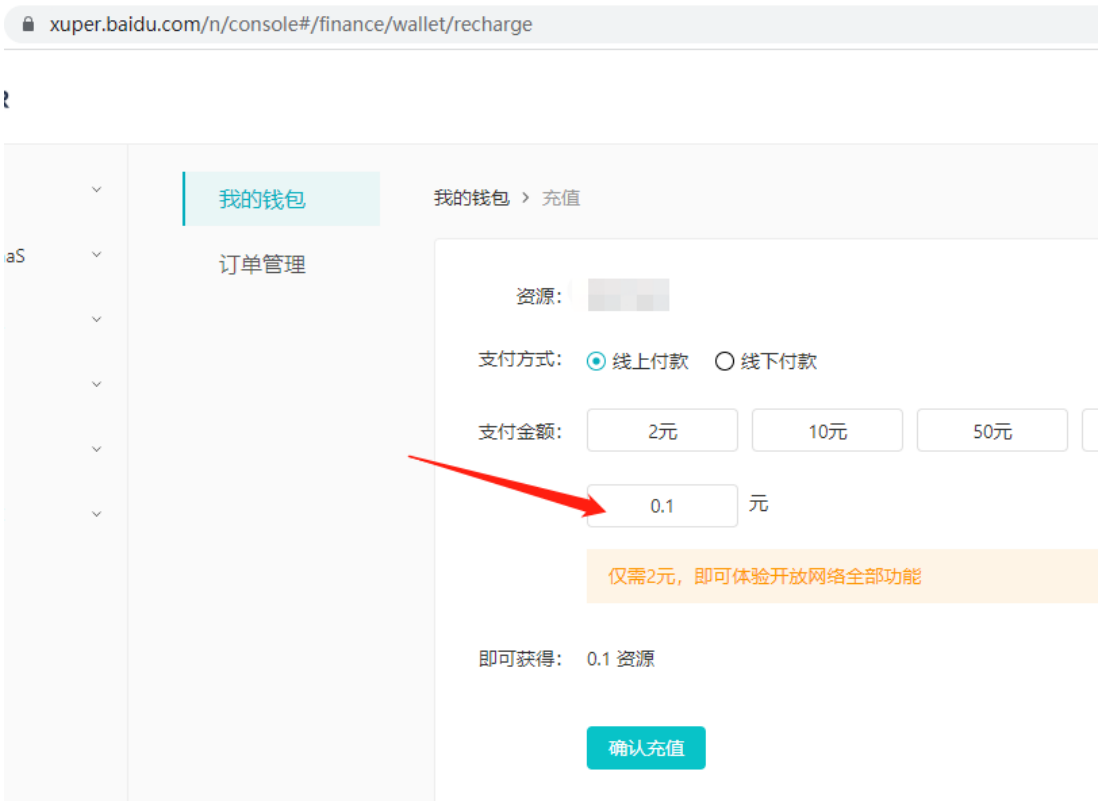
密钥托管 ^{New}

5. 点击右上角头像查看address




充值

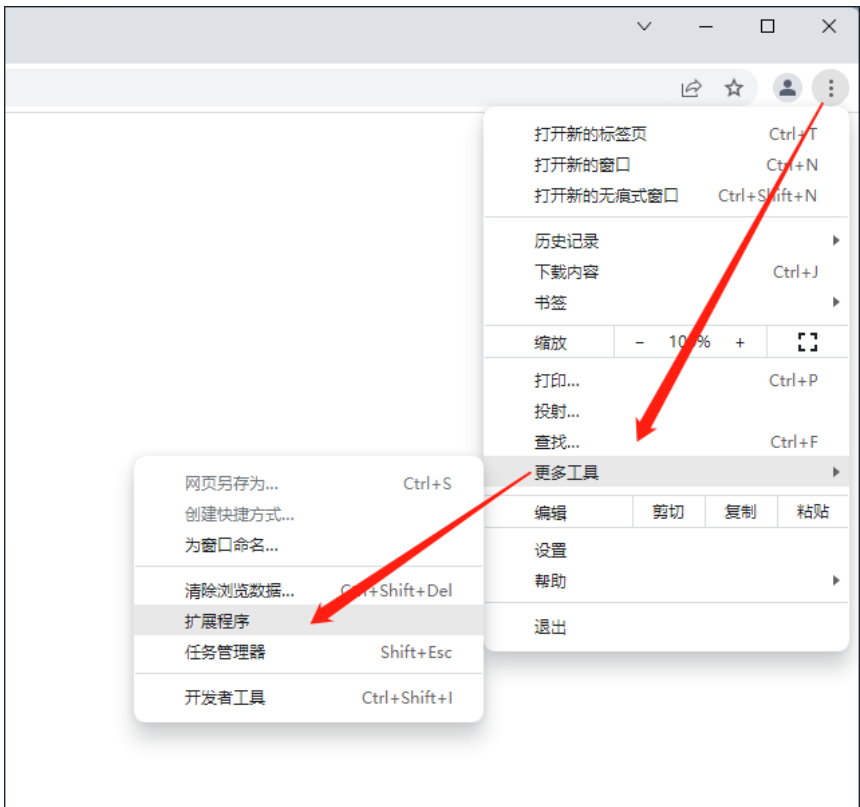
Tip: 用户地址下需要有百度开放网络余额才能使用转移资产，查询余额等功能。建议在百度开放网络充值0.1元。
充值链接: <https://xuper.baidu.com/n/console#/finance/wallet/recharge>



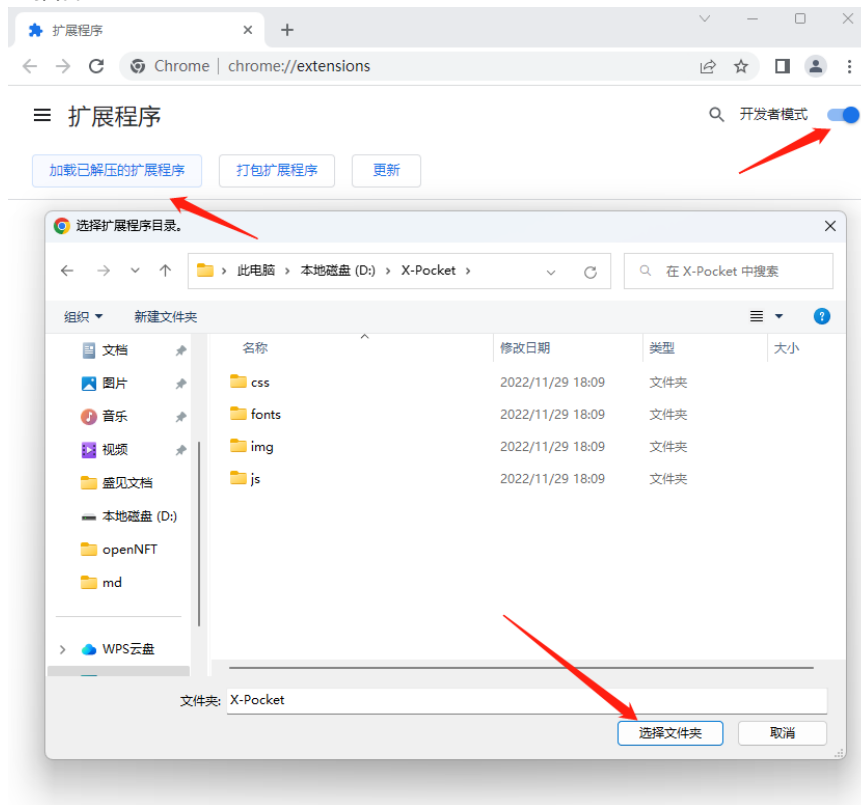
安装

钱包插件已放置根目录下  点击下载

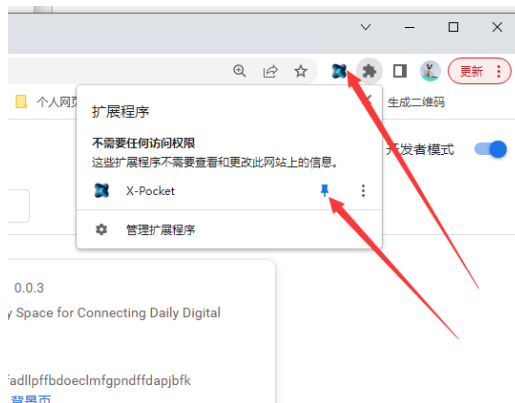
1. 浏览器选择管理扩展程序



- 首先打开开发者模式，然后解压下载的压缩包并选择加载，此时您可以看到浏览器已经安装好该钱包插件了



- 您可以选择钱包插件常驻



使用

登录

下载私钥到本地之后，打开浏览器钱包插件进入登录页，可以选择链类型：XuperOS、Ethereum：



Your Daily Space for Connecting

Daily Digital Asset

请先选择链类型

XuperOS

本地私钥

本地私钥路径



*暂无私钥，右键打开新标签页下载 [助记词登录](#)

安全码（可选，开放网络必填）

登录

助记词登录 请点击



Your Daily Space for Connecting

Daily Digital Asset

请先选择链类型

XuperOS

助记词

助记词

*暂无私钥，右键打开新标签页下载 [私钥登录](#)

没有私钥 右键打开创建

登录

私钥登录请点击



Your Daily Space for Connecting

Daily Digital Asset

请先选择链类型

Ethereum

私钥

请输入您的私钥



[创建私钥](#)

[助记词登录](#)

助记词登录请点击

登录



Your Daily Space for Connecting

Daily Digital Asset

请先选择链类型

Ethereum

助记词

请输入您的助记词

[创建私钥](#)

[私钥登录](#)

没有私钥可创建

私钥登录请点击

登录

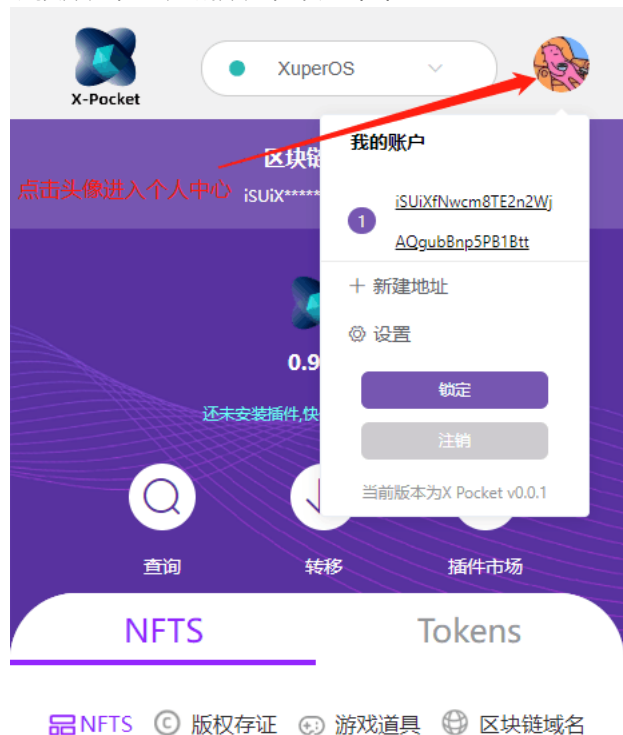
首页

展示钱包余额、网络选择、查询功能、转移功能、插件市场以及NFTs和Tokens资产



个人中心

切换账户，创建新账户，设置菜单



自定义操作

首页==>点击查询进入自定义操作

X-Pocket

XuperOS



< 首页/操作

* 操作名称

请输入您要添加的操作名称

* 操作类型

☐ 交易 ☐ 查询

* 合约名

请输入合约名

* 方法名

请输入方法名


添加操作参数

取消


确定

转移

首页==>点击转移 键入转移操作

X-Pocket

Ethereum



< 首页/转移

接收方地址

请输入接收方地址

转移金额

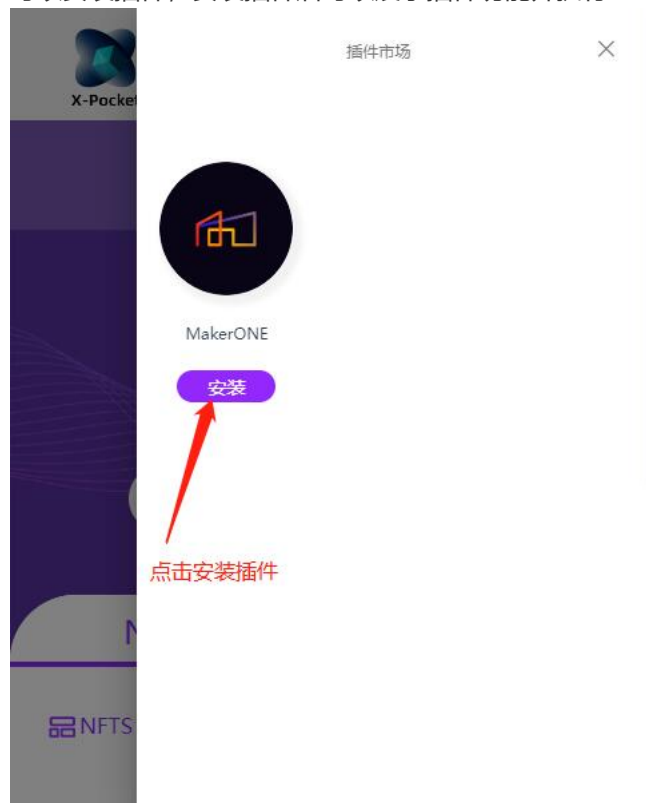
请输入转移金额

当前余额:0.0

确认

插件市场

可以安装插件，安装插件后可以展示插件功能并执行



插件首页



插件功能



切换插件



通过插件查询NFTS



品 NFTS © 版权存证 游戏道具 区块链域名



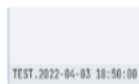
福象



福鹿



福鹿



点击NFT查看详情



Tokens



设置

包含管理网络

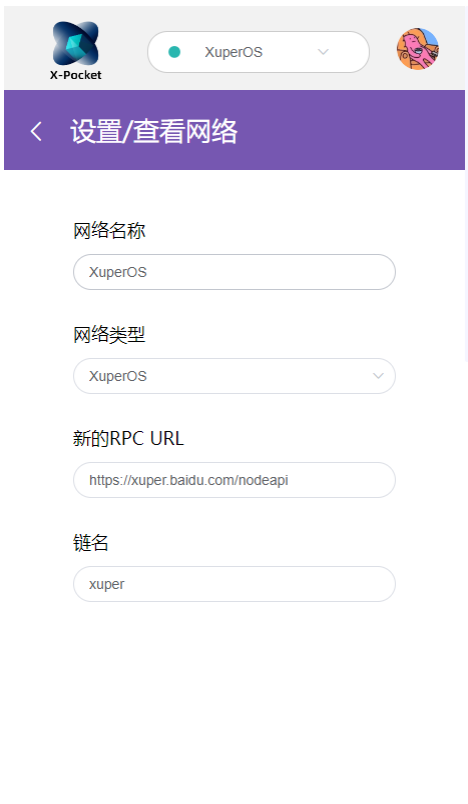


管理网络



添加网络

点击网络 查看网络



插件规范说明

插件合约

1.合约部署

由于插件ID 是在合约内部自增， 需要提供插件 ID 初始值。

```
// 合约初始化 插件id 的起始值
constructor(uint256 id) {
    owner = msg.sender;
    pluginID = id;
}
```

2.添加审核插件信息的审核者

此方法只能由合约部署者调用。 `operator` : 审核者地址; `isApprove`: `true/false` 是否授权

```
function setController(address operator, bool isApprove) public ownerOnly {}
```

3.注册插件（添加插件信息

注册没有权限限制，谁都可以调用。注册的插件需要审核后才能被查询到。

`possessor`: 插件拥有者; `name`: 插件名称，链上唯一; `desc`: 插件描述; `data`: 插件数据（一般为JSON字符串）

`typeName`: 插件类型 `version`: 插件版本 `logo`: 插件logo

returns: 返回插件ID

```
function addPluginInfo(address possessor, string memory name, string memory desc, string memory data, string memory typeName, string memory version, string memory logo) public returns (uint256) {}
```

4.插件审核

审核者对插件进行审核通过

`pluginId`: 插件ID

```
function checkPlugin(uint256 pluginId) public controllerOnly {}
```

5.修改插件信息（只能由插件信息拥有者即注册人 修改）

`pluginId`: 插件ID; `desc`: 插件描述; `data`: 插件数据（一般为JSON字符串）

`typeName`: 插件类型 `version`: 插件版本 `logo`: 插件logo

```
function updatePlugin(uint256 pluginId, string memory desc, string memory data, string memory typeName, string memory version, string memory logo) public {}
```

6.查询插件信息

可以根据插件ID 或者 名字 查询插件

`pluginId`: 插件ID; `pluginName`: 插件名称

`id`: 插件ID; `name`: 插件名称; `desc`: 插件简述; `logo`: 插件logo

```
function getPluginById(uint256 pluginId) public view returns (uint256 id,
string memory name, string memory desc, string memory logo) {}
```

```
function getPluginByName(string memory pluginName) public view returns
(uint256 id, string memory name, string memory desc, string memory logo) {}
```

7.查询所有插件ID

查询所有插件ID (审核通过的)

returns: 所有已审核过的插件ID

```
function allIds() public view returns (uint256[] memory) {}
```

8.删除插件 (只有审核者可以调用)

修改插件审核状态为 false。不能被查询到。 `pluginId`: 插件ID

```
function deletePlugin(uint256 pluginId) public controllerOnly {}
```

插件的JSON格式

```
{
  "addList": [
    // 功能操作合集
    {
      "name": "转移资产", //操作名称
      "icon": "el-icon-sort", // 操作icon, 图标地址:https://element.eleme.io/#/zh-CN/component/icon
      "type": "transaction", // 操作类型, query查询, transaction交易转移等
      "formValue": [
        //操作参数
        {
          "label": "来源账户Address",
          "value": "from"
        },
        {
          "label": "接受者账户Address",
          "value": "to"
        },
        {
          "label": "资产ID",
          "value": "id"
        },
        {
          "label": "编号",
          "value": "sonId"
        }
      ]
    }
  ]
}
```

```

        {
            "label": "数量",
            "value": "amount"
        },
        {
            "label": "额外数据",
            "value": "data"
        }
    ],
    "methodName": "safeTransfer", //操作方法, 合约方法。针对xuper
    "contractName": "makerone", //合约名
    "txType": "1", //是否需要调用合约方法, 1, 需要, 0 不需要 只需要调用固有api
    "search": "transaction" //api取值参数, 解析xuper 中 xsdk.queryTransaction的结果内容。
},
{
    "name": "查询NFT余额",
    "icon": "el-icon-search",
    "type": "query",
    "formValue": [
        {
            "label": "资产ID",
            "value": "id"
        },
        {
            "label": "资产编号",
            "value": "sonId"
        },
        {
            "label": "查询账户",
            "value": "from"
        }
    ],
    "methodName": "getTokenBalance",
    "contractName": "makerone",
    "txType": "1",
    "search": "responses"
},
{
    "name": "查询交易",
    "icon": "el-icon-search",
    "type": "query",
    "formValue": [
        {
            "label": "交易ID",
            "value": "txId"
        }
    ],
    "methodName": "getTokenBytes",
    "contractName": "opennft",
    "txType": "0",
    "txName": "queryTransaction",
    "search": "tx"
}
],
"type": "xuper", //插件类型 xuper属于开放网络, eth属于以太坊网络
"tabCont": [
    //首页五大模块, 内容。

```

```
[
  //nfts,
  {
    "nftsname": "MakerONE", //nfts 名称,
    "nftsur1":
      "https://testmakerone.shengjian.net/qianbao/api/qianbao/xuperChain/userAsserts",
      //请求地址, xuper是本地服务, eth是nftscan的请求连接。
      eg:https://restapi.nftscan.com/api/v2/account/own/0xca1257ade6f4fa6c6834fdc42e03
      0be6c0f5a813?erc_type=erc721
    "xapikey": "PbTZQvKNwCJWA2nAuQyFELrS", //nftscan 请求头的header X-API-KEY
    字段。
    "contract_address": "makerone" //资产的 NFT 合约地址
  }
],
[],
[],
[],
[]
]
```