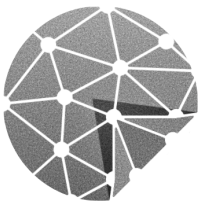
8月7日OpenSpace公开课

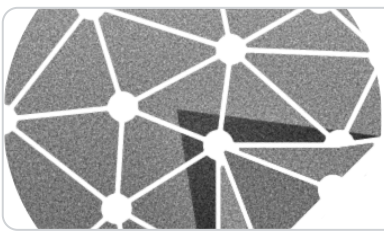
比特币UTXO模型智能合约入门

比特币UTXO模型的区块链上如何实现智能合约？同EVM有什么不同？

为什么比特币脚本是图灵完备的？如何用高级语言写比特币脚本？

预备资料





https://noteprotocol.org/zh/docs/tutorial

Note Protocol

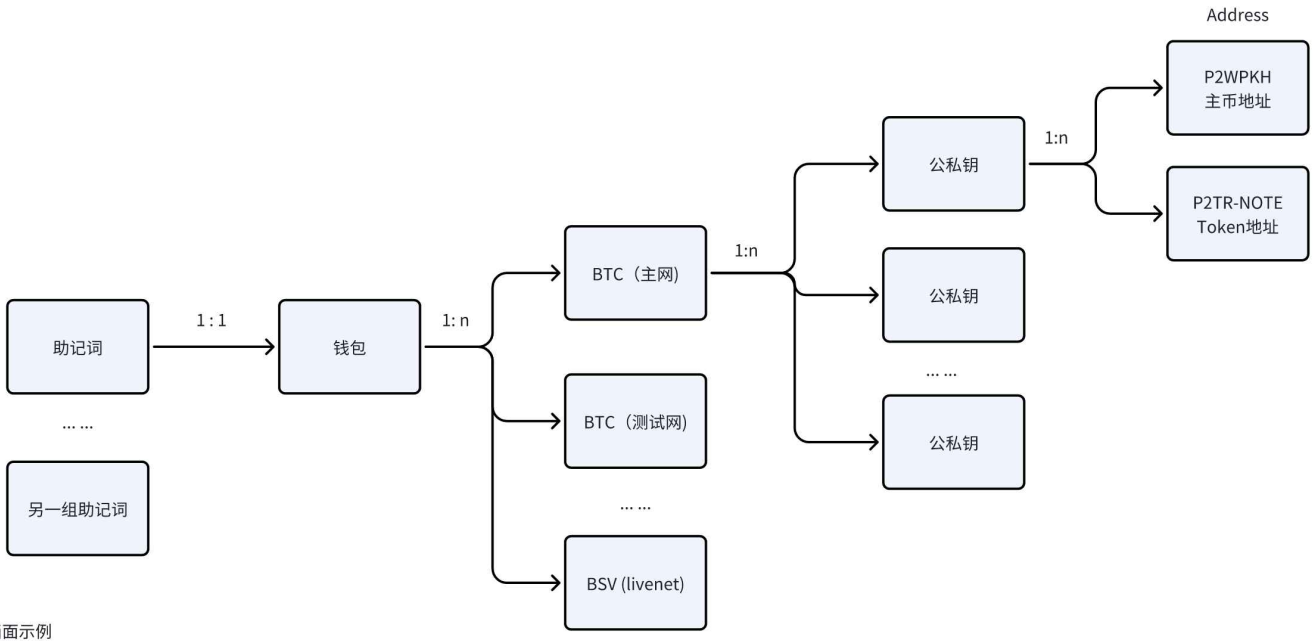
本教程将引导您完成使用NOTE协议开发智能合约，从而发布数字资产的整个流程。

KISS原则

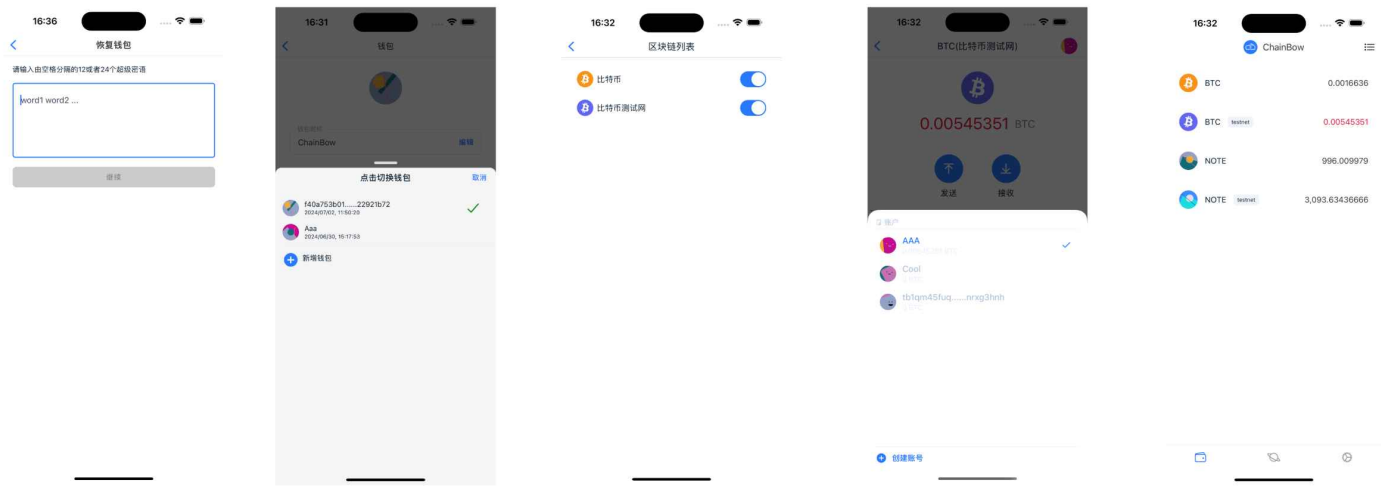
基本概念

在比特币UTXO模型中，首先需要理解以下的名词。部分概念内容同以太坊账号模型稍有不同。

- 助记词：助记词是一组单词，通常由12个或者24个英文单词组成（也有其他语言其他单词数的助记词）。可以通过钱包软件，也可以使用专门软件或者硬件，甚至掷股子生成。助记词需要严格保存好，最好使用纸和笔记录下来，物理保管。
- 钱包：钱包从助记词(种子)可以派生出多个公私钥。钱包主要负责地址的生成与管理，对交易进行签名和广播到区块链，从数据服务商获取交易记录，展示给用户。
- 数字货币：钱包通过BIP44等规范，管理包括BTC/BCH/BSV/RXD在内的多种区块链，以及对应的同名数字货币。每个区块链（数字货币）还可以分主网和测试网。
- 地址：钱包每生成一组公私钥，可以按照模版生成多种格式的地址。常见的包括P2PKH，P2WPKH，P2WSH，P2TR。NOTE协议有两种缺省的模版，P2TR-NOTE和P2TR-COMMIT-NOTE。
- 多地址：钱包可以生成多组公私钥，也就可以生成多组地址。生成数量一般没有限制。



画面示例

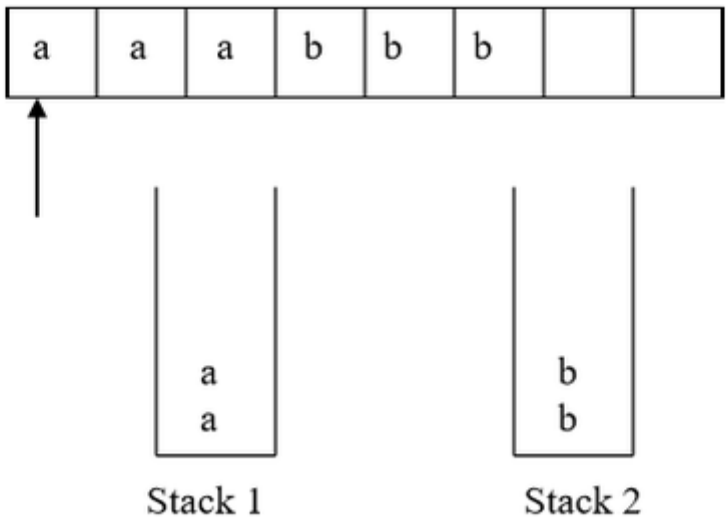


> 以太坊等账号模型的区块链不分主网和测试网，每组公私钥对应的账号有一个固定的地址。而在比特币UTXO模型中，没有账号的概念，每组公私钥按照模版可以生成多种地址，所有的地址形式不同，但都受私钥控制。

脚本模版/智能合约

比特币脚本运行引擎

- 2-Stack-PDA 双栈下推自动机
  - 上下文无关
  - 图灵等价《-》图灵完备



https://lilong.net/2020/09/13/start-bitcoinscript-1/

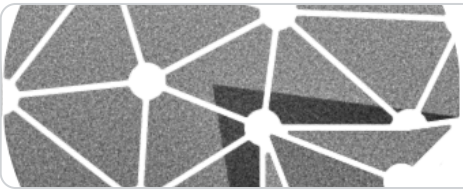
实战比特币脚本编程（1）两段

废话不多说，直接看程序，只需基础的程序知识就可以理解

https://lilong.net/2020/09/14/start-bitcoinscript-2/

实战比特币脚本编程（2）困局

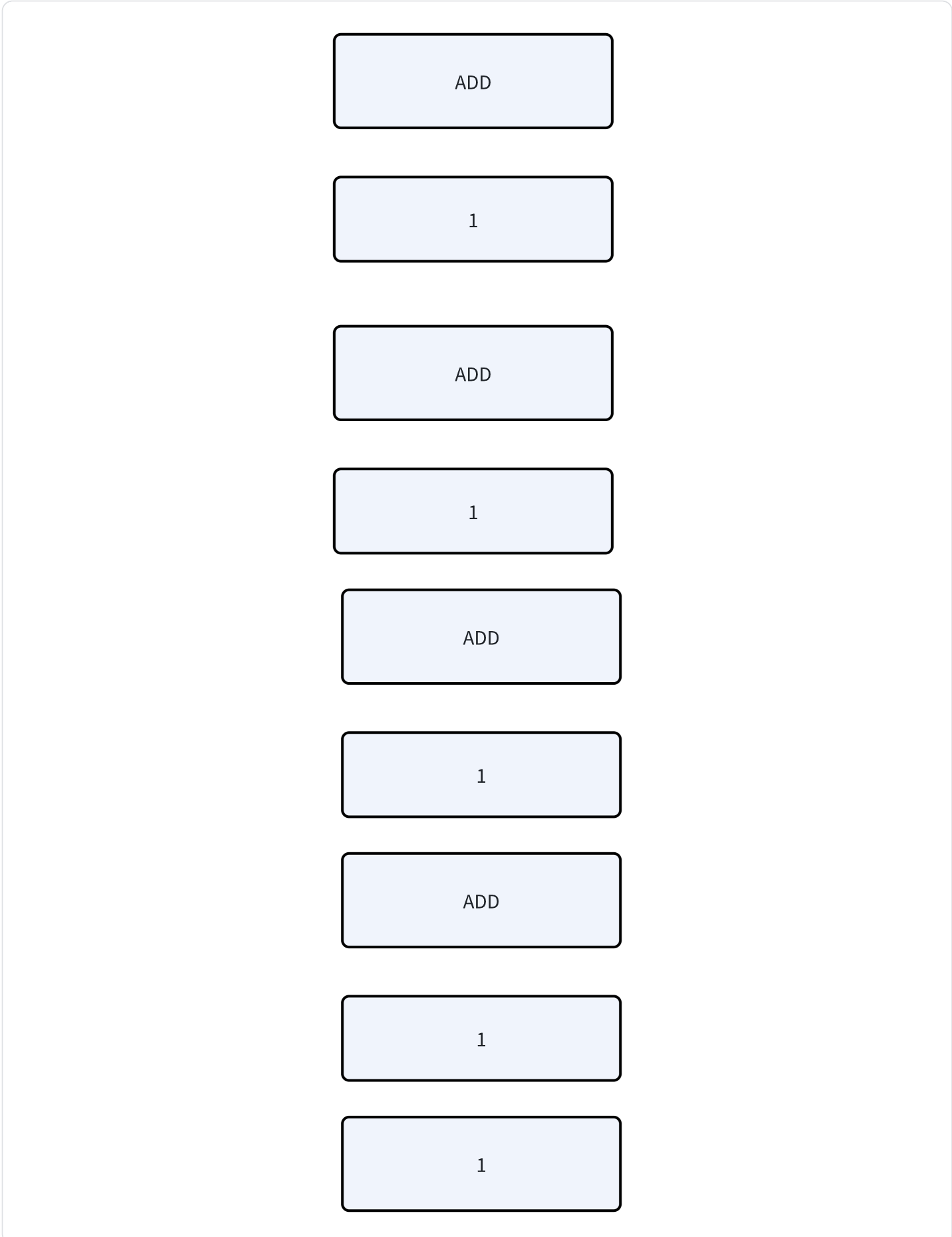
PUSH ONLY 车到山前，路呢？柳暗花明，村呢？



https://noteprotocol.org/zh/blog/turing-complete-halting-problem

图灵完备和停机问题 | Note Protocol

图灵完备和停机问题



OP_CAT

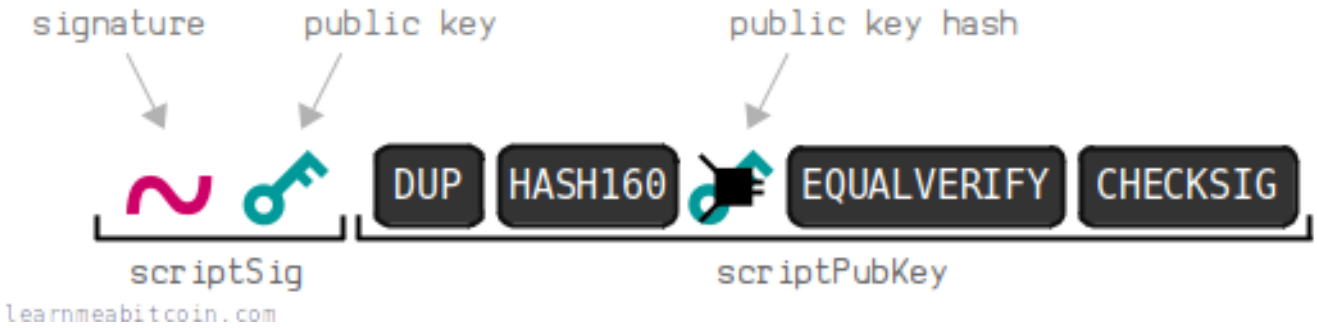
人为因素

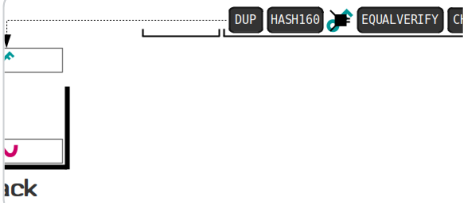
脚本模版

- P2PK
- P2PKH脚本

```
解锁脚本：签名 公钥
锁定脚本：DUP HASH160 公钥哈希 EQUALVERIFY CHECKSIG

签名 公钥 OP_DUP HASH160 公钥哈希 EQUALVERIFY CHECKSIG
```



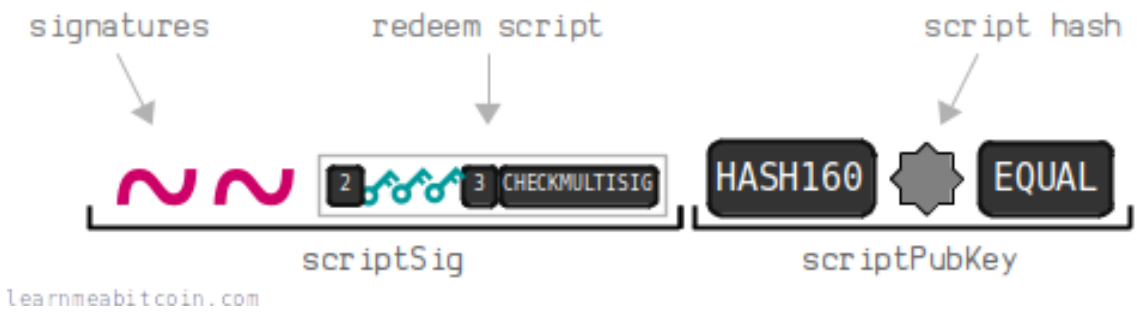


https://learnmeabitcoin.com/technical/script/

Bitcoin Script | A Mini Programming Language

An explanation of what the Script language is, how it works, and how it's used to lock and unlock bitcoins in bitcoin transactions.

- P2SH脚本

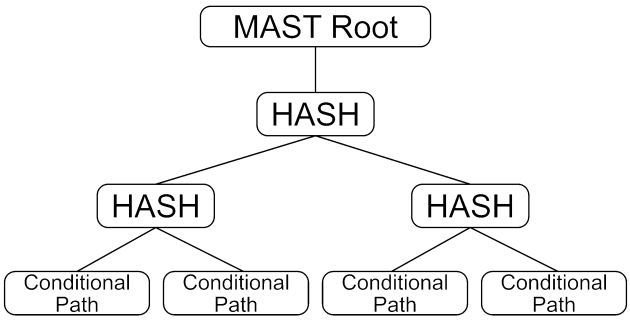



```
解锁脚本：签名 公钥
赎回脚本：DUP HASH160 公钥哈希 EQUALVERIFY CHECKSIG
锁定脚本：HASH160 赎回脚本的哈希 EQUAL


签名 公钥 “DUP HASH160 公钥哈希 EQUALVERIFY CHECKSIG” HASH160 赎回脚本的哈希 EQUAL
```

```
7
```

- P2TR脚本




```
解锁脚本：签名 公钥
Script A: OP_ADD <value> OP_EQUAL
Script B: OP_DUP OP_HASH160 <hash> OP_EQUALVERIFY OP_CHECKSIG
Script C: OP_IFDUP OP_NOTIF OP_2SWAP OP_DUP OP_HASH160 <hash> OP_EQUALVERIFY OP_CHECKSIG OP_ENDIF
锁定脚本：OP_1 <x-only 合成公钥>
```

- Ordinal/Atomicals的脚本

```
解锁脚本：签名
赎回脚本：OP_FALSE OP_IF <DATA 520B>  <DATA 520B>  <DATA 520B>  <DATA 520B>  <DATA 520B> OP_ENDIF 公钥 OP_CHECKSIG
锁定脚本：OP_1 <x-only 合成公钥>

```

Ordinals的例子

```
OP_FALSE
OP_IF
  OP_PUSH "ord"
  OP_PUSH 1
  OP_PUSH "text/plain;charset=utf-8"
  OP_PUSH 0
  OP_PUSH "Hello, world!"
OP_ENDIF

```

- NOTE脚本
  - 基础脚本

```
解锁脚本：签名 <DATA0 80B> <DATA1 80B> <DATA2 80B> <DATA3 80B> <DATA4 80B>
赎回脚本：NOTE OP_2DROP OP_2DROP OP_2DROP Pubkey OP_CHECKSIG
锁定脚本：OP_1 <x-only 合成公钥>

```

  - Commit脚本

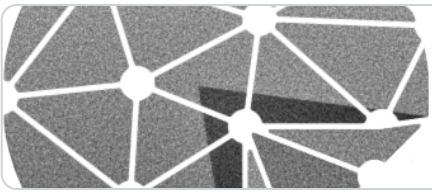
```
解锁脚本：签名
赎回脚本：<DATA0 520B> <DATA1 520B> <DATA2 520B> <DATA3 520B> <DATA4 520B> NOTE OP_2DROP OP_2DROP OP_2DROP Pubkey OP_CHECKSIG
锁定脚本：OP_1 <x-only 合成公钥>
```

  - 更复杂的脚本
    - 加法

```
解锁脚本：签名 <value0> <value1> <DATA0 80B> <DATA1 80B> <DATA2 80B> <DATA3 80B> <DATA4 80B>
赎回脚本：NOTE OP_2DROP OP_2DROP OP_2DROP OP_ADD 3 OP_EQUAL Pubkey OP_CHECKSIG
锁定脚本：OP_1 <x-only 合成公钥>

```

    - 多签
    - 链上合约



https://noteprotocol.org/zh/docs/protocol/Problems%20And%20Sol···

Note Protocol

2.4 链上合约

https://docs.scrypt.io/tutorials/auction

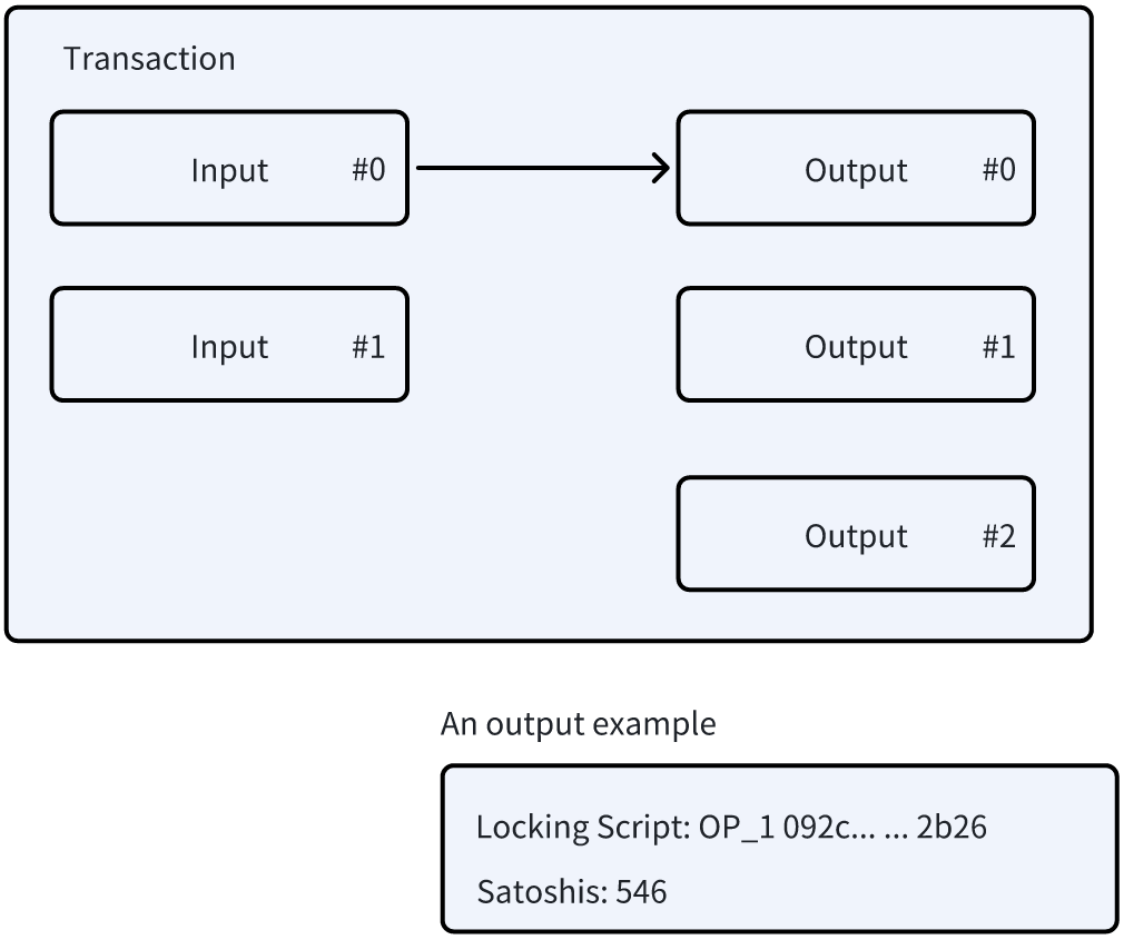
Tutorial 2: Auction | sCrypt

Overview

NOTE核心概念

账号

在比特币中没有账号的概念，只有地址。每种地址由交易输出模版构造而成，一个地址就是对交易输出脚本的编码。



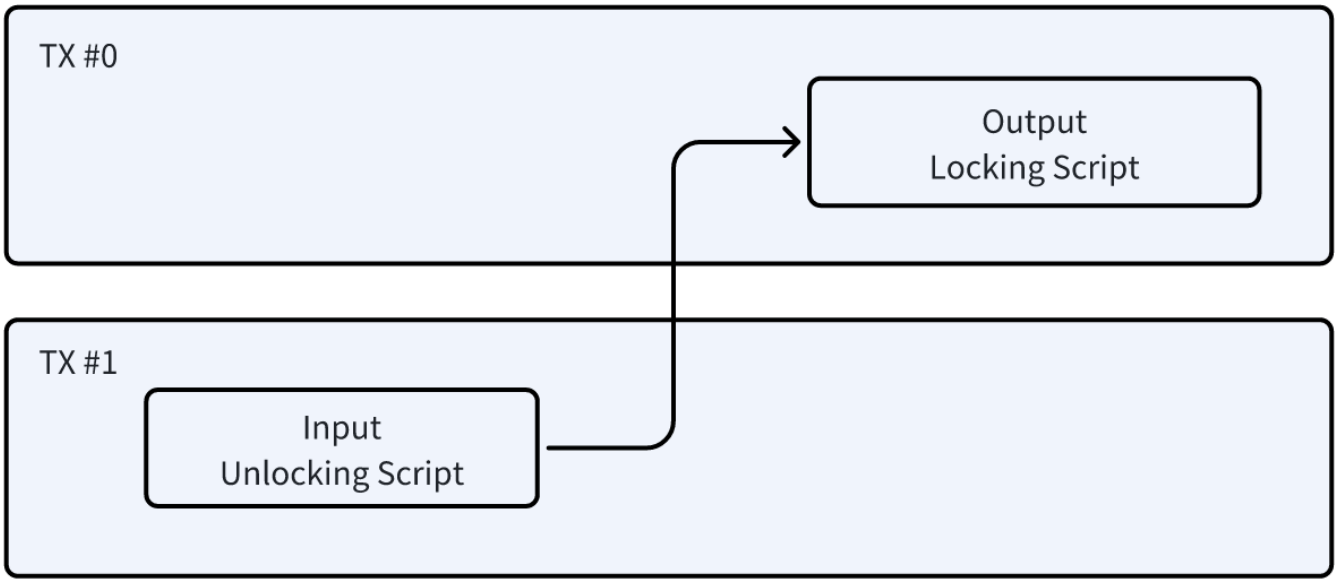
在NOTE协议中，我们规定一个交易输出的脚本哈希值为账号。这个定义来自Electrum协议

```
scriptHash = sha256(script).reverse()
```

总结，账号是交易输出的哈希值，长度固定为256位(32个字节)，hex字符串长度是64。地址是交易输出的编码，长度根据脚本的内容和编码规则不同。从地址可以转换为账号，反过来，从账号无法转换为地址。

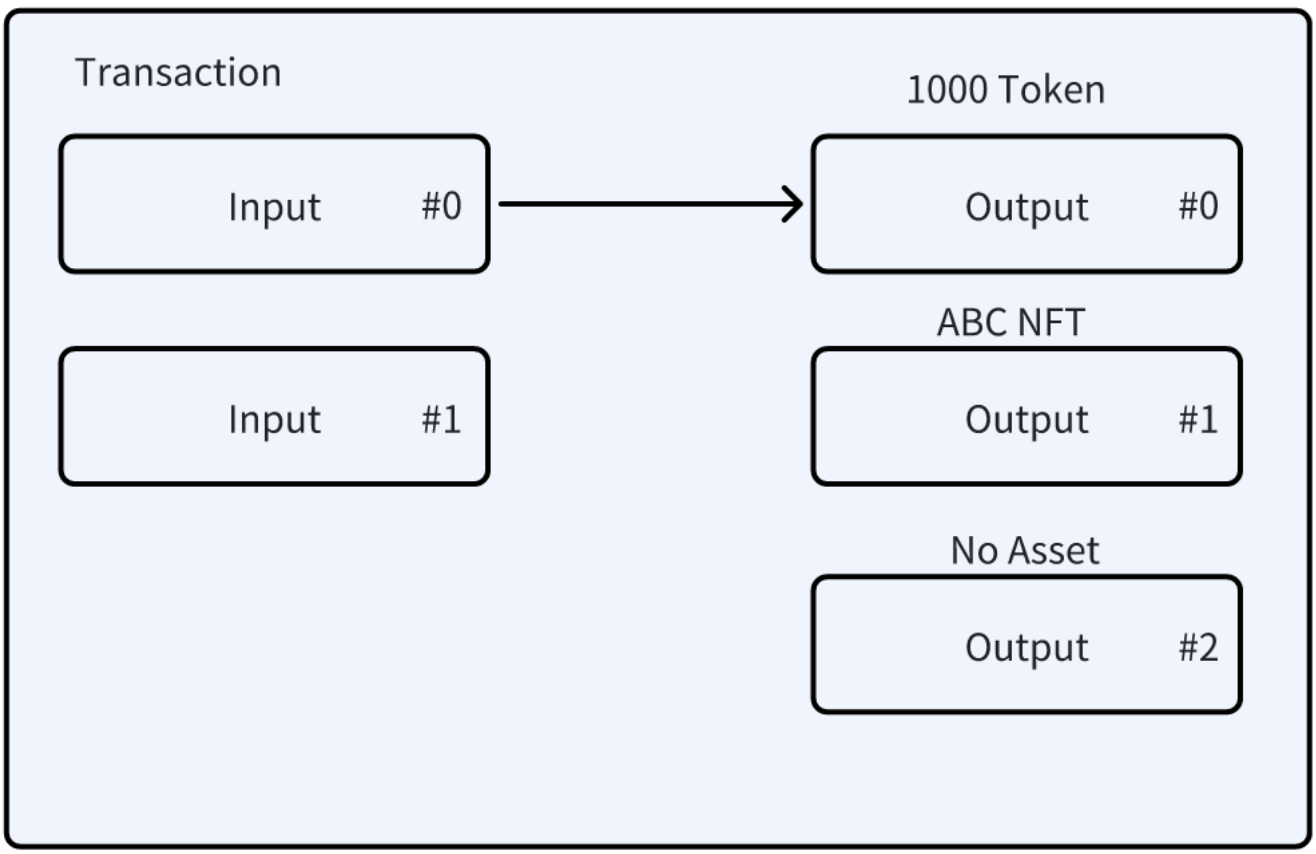
账号的所有权

账号的所有权归属于可以解锁交易输出的一个或者多个私钥持有者。在常见普通的比特币交易中，一个私钥持有者，通过对UTXO的提供解锁脚本，经过矿工检查后获得所有权。同理，如果是一个多签地址，则由多个人的私钥共同签名后获取所有权。



数字资产

数字资产由协议定义，包括N20 Token和N721 NFT等等，数字资产绑定在交易输出上。不同于染色币，这种绑定不依赖于交易输出的satoshi数量，只跟UTXO本身有关。



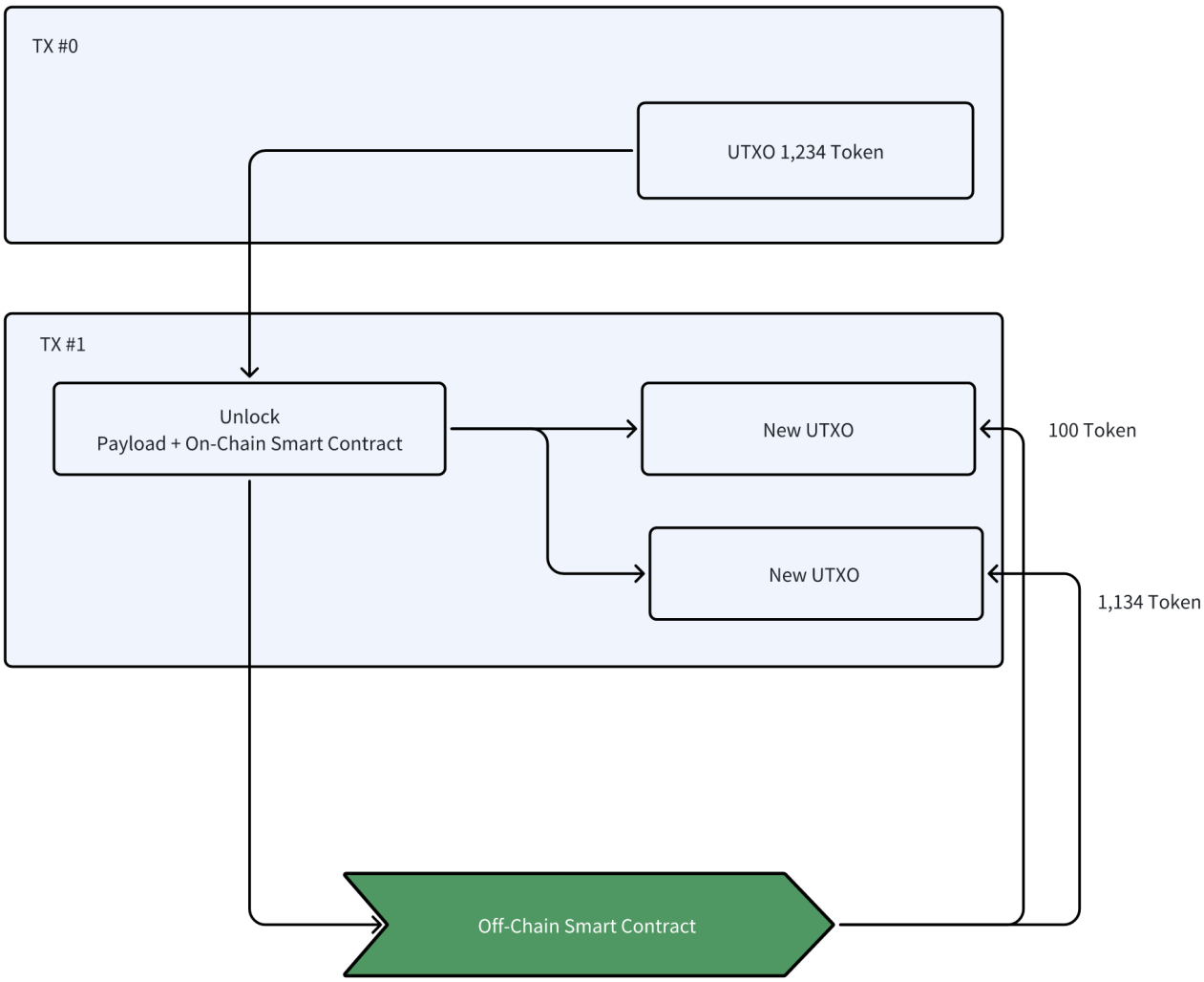
操作

操作是指mint，transfer, burn等由协议定义，对数字资产的进行的操作。操作由msgpack编码的一组key-value定义，我们叫做Payload，放置在解锁脚本中。也就是说，如果账号的所有者需要操作数字资产，首先需要通过签名解锁账号UTXO。这个过程就是比特币的交易过程，执行了链上合约，安全性由比特币的矿工机制确保。 比如一个Payload例子

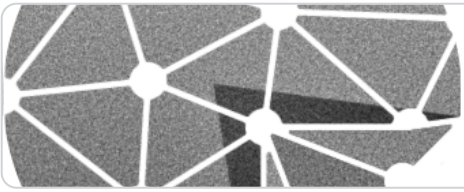
```
{
  "p": "n20",
  "op": "transfer",
  "tick": "NOTE",
  "amt": 100
}
```

它表示解锁UTXO，将绑定在UTXO中的名为NOTE的数字资产转移给第一个交易输出，如果UTXO中的数字资产大于100，那么余额将转给第二个交易输出。

一旦交易被发布到区块链上，意味着链上合约执行成功所有权已经确认。NOTE协议索引器从区块链上获取此笔交易，对解锁脚本中的Payload进行解释，检查Payload是否有效，被解锁的UTXO中是否包含足够的代币数量，然后执行链下智能合约，执行tick名字背后的智能合约里的方法，方法名就是op的值，比如transfer。一旦合约执行成功，相应数量的数字资产将被绑定在交易输出上。



链下合约

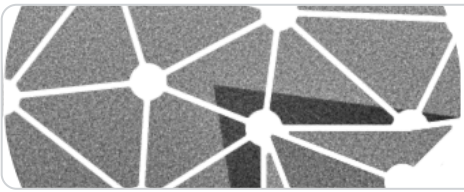


https://noteprotocol.org/zh/docs/protocol/Token/4.1-OffchainContracts

Note Protocol

4.1 链下智能合约

快速开始



https://noteprotocol.org/zh/docs/tutorial/quick-start

快速开始 | Note Protocol

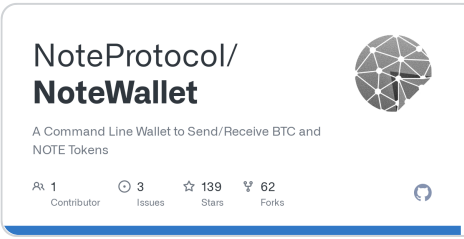
1. 安装开发环境

语言和工具

https://docs.scrypt.io/

Overview | sCrypt

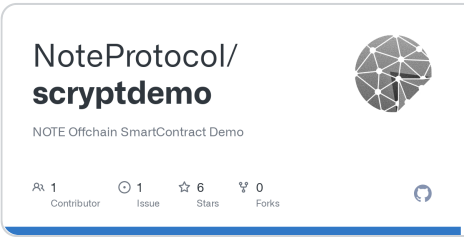
sCrypt is an embedded Domain Specific Language (eDSL) based on TypeScript for writing smart contracts on Bitcoin. Embedded means that it is a language inside another language. sCrypt is strictly a sub



https://github.com/NoteProtocol/NoteWallet

GitHub - NoteProtocol/NoteWallet: A Command Line Wallet to Send/Receive BTC and NOTE Tokens

A Command Line Wallet to Send/Receive BTC and NOTE Tokens -···



https://github.com/NoteProtocol/scryptdemo

GitHub - NoteProtocol/scryptdemo: NOTE Offchain SmartContract Demo

NOTE Offchain SmartContract Demo. Contribute to NoteProtocol/scryptdemo···

开发者社区

https://x.com/lilong

https://x.com/NoteProtocol

https://x.com/sCryptPlatform

https://t.me/NoteProtocol_org

DAPP开发

基本的理念》误区

- 交易里不适合存大数据
- 大区块的目的是更多的交易，而不是存更多的数据
  - OP_RETUN: 0 sat > 没有意义
- 比特币是为了交易而存在的，而不是为了存数据
- 对比：以太坊NFT，巨大的市值》没有存数据在ETH〉成功
- 链上：所有权确认和资产转移