区块链智能合约开发

中山大学

数据科学与计算机学院

上节回顾

智能合约概念

“一个智能合约是一套以数字形式定义的承诺（promises），包括合约参与方可以在上面执行这些承诺的协议。”

——尼克·萨博，1993

- 事件驱动
- 自动执行
- 价值转移
- 中心化，出错难以追溯

  大额交易不可靠

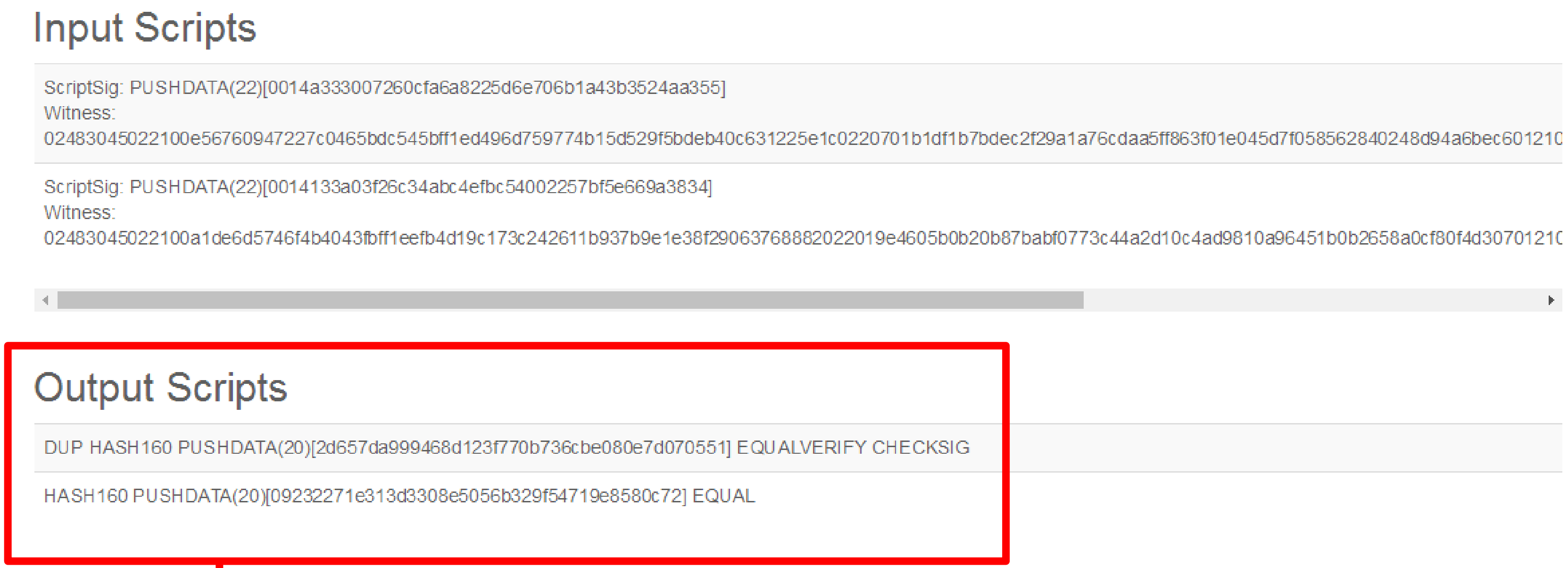
上节回顾

■ 中心化程序

您正在使用个人时间进行游戏，将在2022分钟后过期．您将在游戏时间结束之前的30分钟得到通知。

上节回顾

■ 比特币中的脚本（非图灵完备）



Input Scripts

ScriptSig: PUSHDATA(22)[0014a333007260cfa6a8225d6e706b1a43b3524aa355]
Witness:
02483045022100e56760947227c0465bdc545bff1ed496d759774b15d529f5bdeb40c631225e1c0220701b1df1b7bdec2f29a1a76cdaa5ff863f01e045d7f058562840248d94a6bec601210

ScriptSig: PUSHDATA(22)[0014133a03f26c34abc4efbc54002257bf5e669a3834]
Witness:
02483045022100a1de6d5746f4b4043fbff1eefb4d19c173c242611b937b9e1e38f29063768882022019e4605b0b20b87babf0773c44a2d10c4ad9810a96451b0b2658a0cf80f4d30701210

Output Scripts

DUP HASH160 PUSHDATA(20)[2d657da999468d123f770b736cbe080e7d070551] EQUALVERIFY CHECKSIG

HASH160 PUSHDATA(20)[09232271e313d3308e5056b329f54719e8580c72] EQUAL

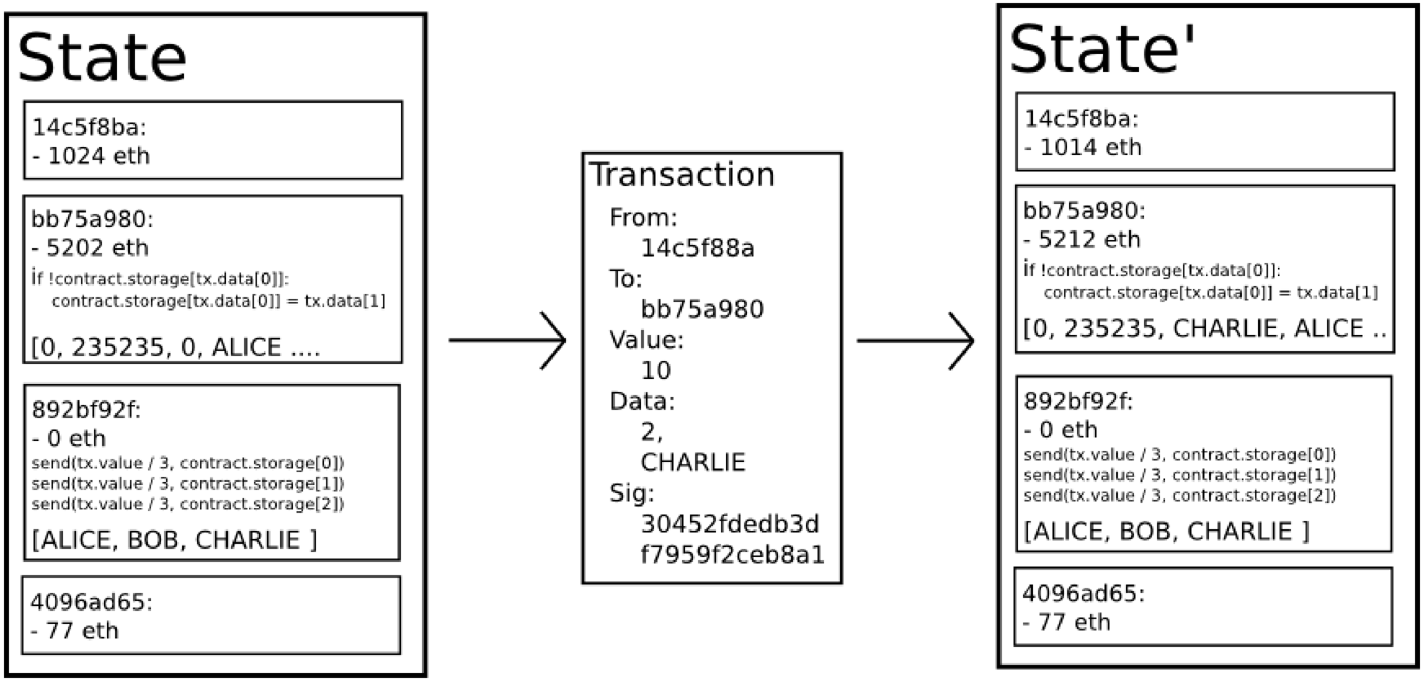
非图灵完备　　对比特币进行改良？

Photo: Courtesy of Dmitry Buterin

上节回顾

以太坊（Ethereum）

- 重要思路：通过区块链交易进行系统的状态转换。



State

14c5f8ba:
- 1024 eth

bb75a980:
- 5202 eth

```
if !contract.storage[tx.data[0]]:
    contract.storage[tx.data[0]] = tx.data[1]
```

[0, 235235, 0, ALICE ....

892bf92f:
- 0 eth

```
send(tx.value / 3, contract.storage[0])
send(tx.value / 3, contract.storage[1])
send(tx.value / 3, contract.storage[2])
```

[ALICE, BOB, CHARLIE]

4096ad65:
- 77 eth

Transaction

From:
14c5f88a
To:
bb75a980
Value:
10
Data:
2,
CHARLIE
Sig:
30452fdedb3d
f7959f2ceb8a1

State'

14c5f8ba:
- 1014 eth

bb75a980:
- 5212 eth

```
if !contract.storage[tx.data[0]]:
    contract.storage[tx.data[0]] = tx.data[1]
```

[0, 235235, CHARLIE, ALICE ..

892bf92f:
- 0 eth

```
send(tx.value / 3, contract.storage[0])
send(tx.value / 3, contract.storage[1])
send(tx.value / 3, contract.storage[2])
```

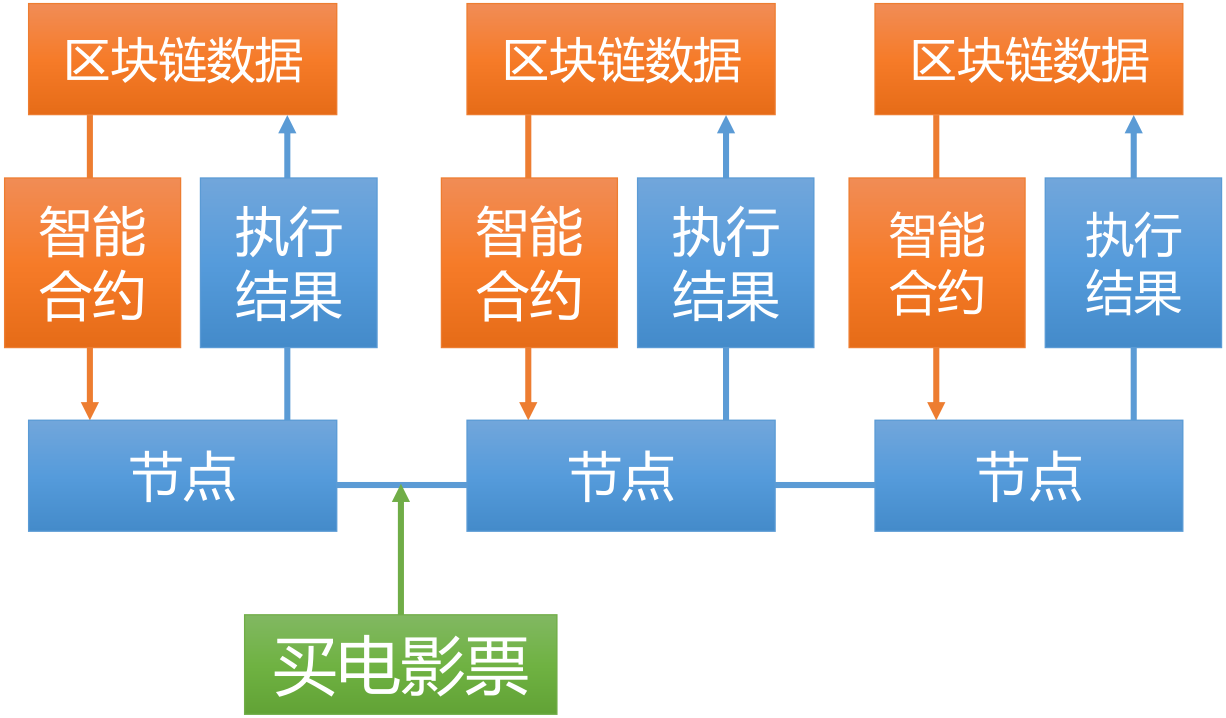
[ALICE, BOB, CHARLIE]

4096ad65:
- 77 eth

上节回顾

区块链智能合约

- 去中心化执行
- 不可篡改
- 可回溯



上节回顾

以太坊账户结构

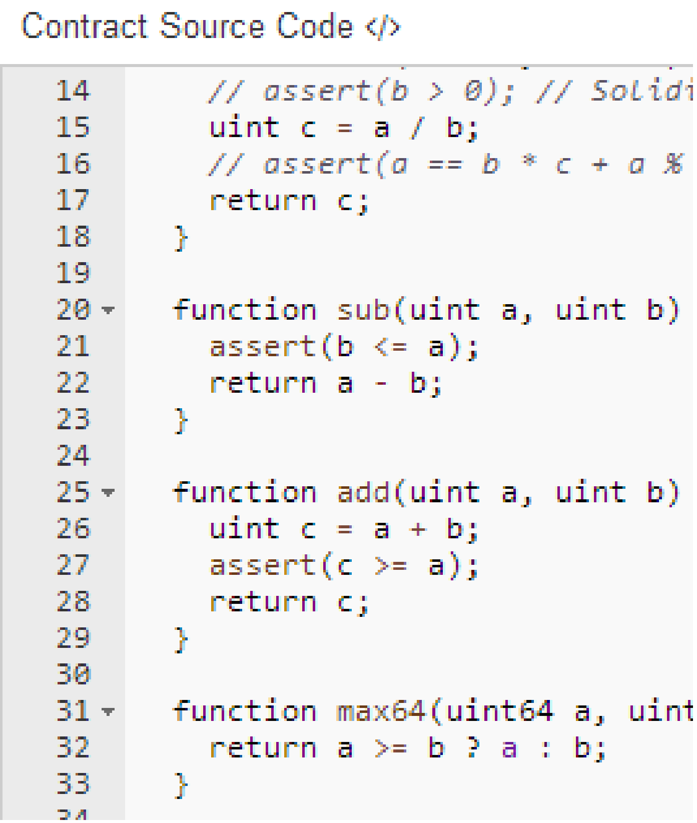
- 以太坊采用Merkle Patricia Tree对账户信息哈希
- 人+合约的存储信息->Worldstate
- 延伸概念：Radix Tree、RLP编码、Merkle Patricia Tree

问题：为什么要有stateRoot?

方便节点间状态的互相验证，保证在交易的每个区块（每时每刻），所有节点的状态是一致的。

上节回顾

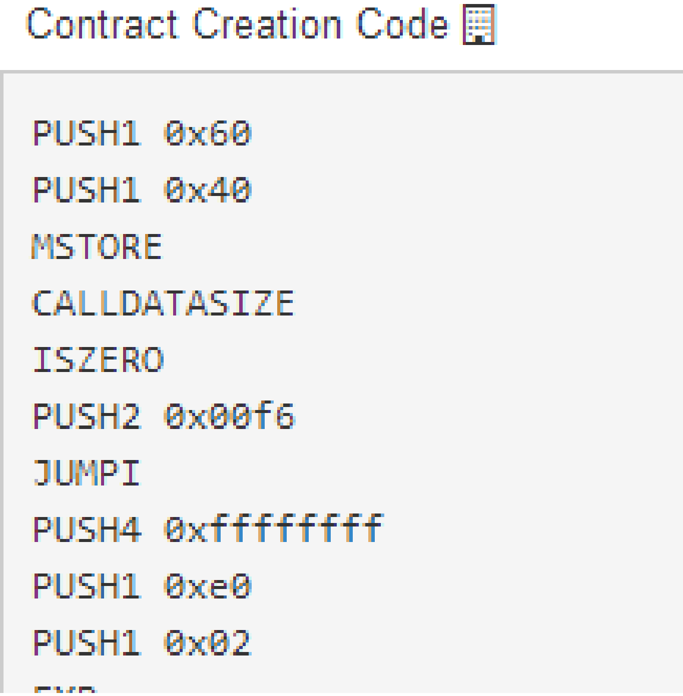
- **Gas**



Contract Source Code </>

```
    // assert(b > 0); // Solidi
    uint c = a / b;
    // assert(a == b * c + a %
    return c;
  }

  function sub(uint a, uint b)
    assert(b <= a);
    return a - b;
  }

  function add(uint a, uint b)
    uint c = a + b;
    assert(c >= a);
    return c;
  }

  function max64(uint64 a, uint
    return a >= b ? a : b;
  }
```

=>



Contract Creation Code

```
PUSH1 0x60
PUSH1 0x40
MSTORE
CALLDATASIZE
ISZERO
PUSH2 0x00f6
JUMPI
PUSH4 0xffffffff
PUSH1 0xe0
PUSH1 0x02
EXP
```

问题：Gas，GasPrice，GasLimit，GasUsed的区别？

上节回顾

以太坊交易(事务)结构

交易中的Nonce值

from账户发出交易的次数, 同一账户的交易会被依次确认

问题：nonce值有什么用?

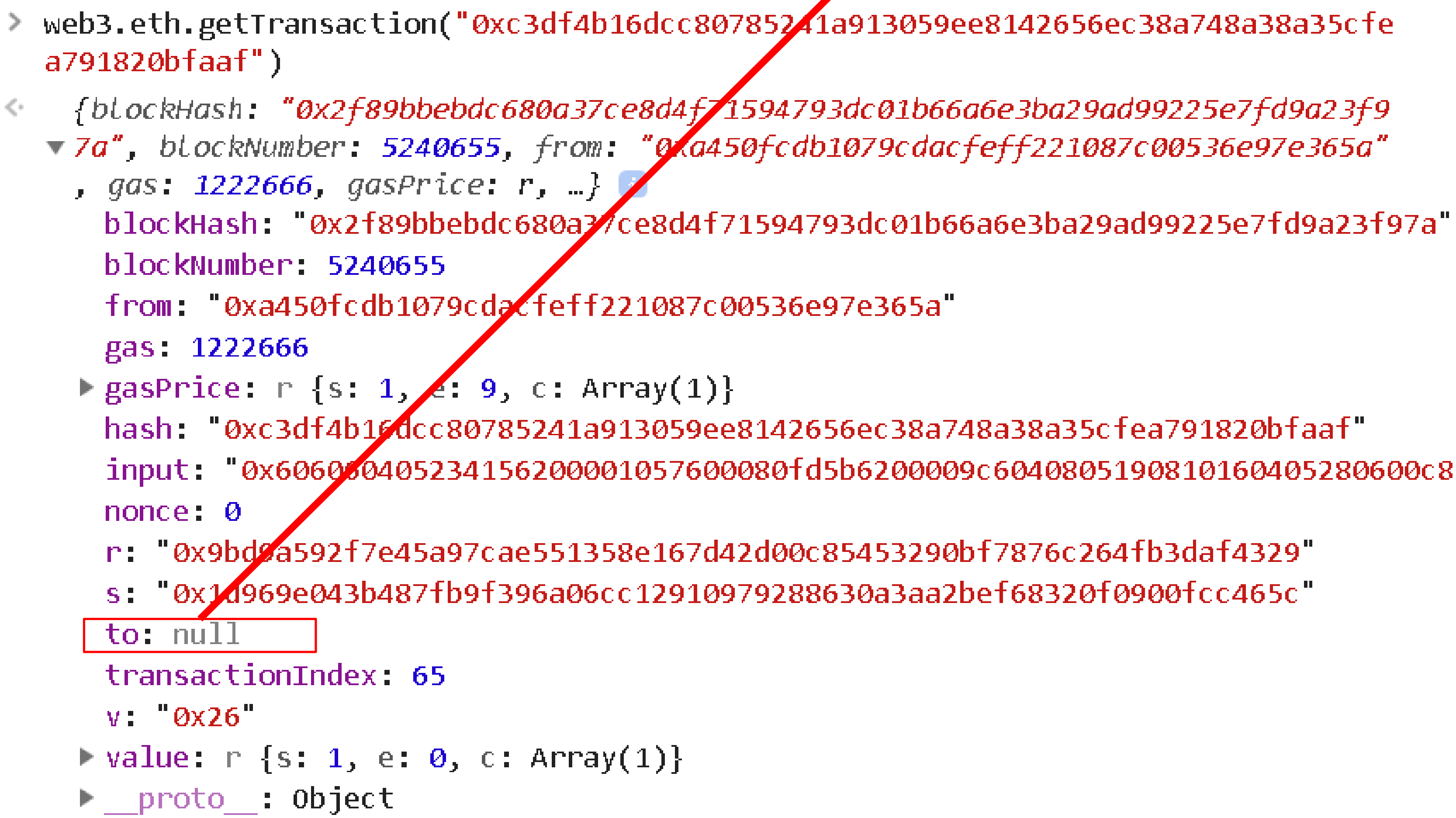
- 区块中的nonce值：挖矿
- 交易中的nonce值：
  1. 确认交易顺序
  2. 防止双花
  3. 撤销pending中的交易
  4. 确定生成的合约地址

上节回顾

以太坊交易(事务)结构

问题：to为什么是空的?

- 试试这条交易 0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfea791820bfaaf



```
> web3.eth.getTransaction("0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfe
  a791820bfaaf")
<   {blockHash: "0x2f89bbebdc680a37ce8d4f71594793dc01b66a6e3ba29ad99225e7fd9a23f9
  ▼ 7a", blockNumber: 5240655, from: "0xa450fcdb1079cdacfeff221087c00536e97e365a"
    , gas: 1222666, gasPrice: r, …}
      blockHash: "0x2f89bbebdc680a37ce8d4f71594793dc01b66a6e3ba29ad99225e7fd9a23f97a"
      blockNumber: 5240655
      from: "0xa450fcdb1079cdacfeff221087c00536e97e365a"
      gas: 1222666
    ▶ gasPrice: r {s: 1, e: 9, c: Array(1)}
      hash: "0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfea791820bfaaf"
      input: "0x606060405234156200001057600080fd5b6200009c6040805190810160405280600c8
      nonce: 0
      r: "0x9bd0a592f7e45a97cae551358e167d42d00c85453290bf7876c264fb3daf4329"
      s: "0x1d969e043b487fb9f396a06cc12910979288630a3aa2bef68320f0900fcc465c"
      to: null
      transactionIndex: 65
      v: "0x26"
    ▶ value: r {s: 1, e: 0, c: Array(1)}
    ▶ __proto__: Object
```

目录

1. 智能合约及平台简介
2. 以太坊基本操作及原理
3. Solidity语言
4. 联盟链智能合约
5. 为智能合约构造图形交互

Solidity语言

- 专用于开发智能合约的语言
    - Solidity： 类似JavaScript
    - Serpent 、Vyper：类似Python
    - Mutan：类似Go
    - LLL： 类似Lisp
    - 其他：Java, Go, C++

Solidity语言

Solidity简介

- 语法上接近于**Javascript**
- 具有一些特殊类型：**address, event, 以太币单位**等
- 具有一些特殊关键字：**payable, send, now**等
- 公有链环境中需要确定**变量位置**进行付费
- 某段代码失败，整个交易（事务）回撤

  (类比于数据库事务中的**原子性**)

Solidity语言

相关文档

- Github: https://github.com/ethereum/solidity
- 在线编译器：https://remix.ethereum.org/
- 中文编译器：https://editor.hyperchain.cn/
- 中文文档0：https://solidity-cn.readthedocs.io/zh/develop/
- 中文文档1：http://www.tryblockchain.org/
- 中文文档2：https://learnblockchain.cn/categories/ethereum/Solidity/
- 开发框架：https://truffleframework.com/

Solidity语言

Hello World

```
pragma solidity ^0.4.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Solidity语言

值类型

- 布尔类型(Booleans)
- 整型(Integers)
- 定长浮点型(Fixed Point Numbers)
- 定长字节数组(Fixed-size byte arrays)
- 有理数和整型常量(Rational and Integer Literals)
- 字符串常量（String literals）
- 十六进制常量（Hexadecimal literals）
- 枚举(Enums)
- 函数类型(Function Types)
- 地址类型(Address)
- 地址常量(Address Literals)

Solidity语言

值类型——布尔运算及短路规则

- ! （逻辑非）
- && （逻辑与， "and" ）
- || （逻辑或， "or" ）
- == （等于）
- != （不等于）
- 运算符 || 和 && 都遵循同样的短路规则
- 在表达式 f(x) || g(y) 中， 如果 f(x) 的值为 true ， 那么 g(y) 就不会被执行。

Solidity语言

值类型——函数类型

- ✓ 声明：function (<parameter types>) {internal|external} [pure|constant|view|payable] [returns (<return types>)]

内部函数只能在当前合约内被调用。 调用一个内部函数是通过跳转到它的入口标签来实现的，就像在当前合约的内部调用一个函数。

外部函数由一个地址和一个函数签名组成，可以通过外部函数调用传递或者返回。

Solidity语言

■ 值类型——函数类型

- ✓ 声明：function (<parameter types>) {internal|external} [pure|constant|view|payable] [returns (<return types>)]

public：内部、外部均可见

private：仅在当前合约内可见

external：仅在外部可见（仅可修饰函数）——就是说，仅可用于消息调用（即使在合约内调用，也只能通过 this.func 的方式)

internal：仅在内部可见（在当前 Solidity 源代码文件内均可见，不仅限于当前合约内)

Solidity语言

值类型——函数类型

- ✓ 声明：function (<parameter types>) {internal|external} [pure|constant|view|payable] [returns (<return types>)]

pure 修饰函数时：不允许修改或访问状态——但目前并不是强制的。

view 修饰函数时：不允许修改状态——但目前不是强制的。

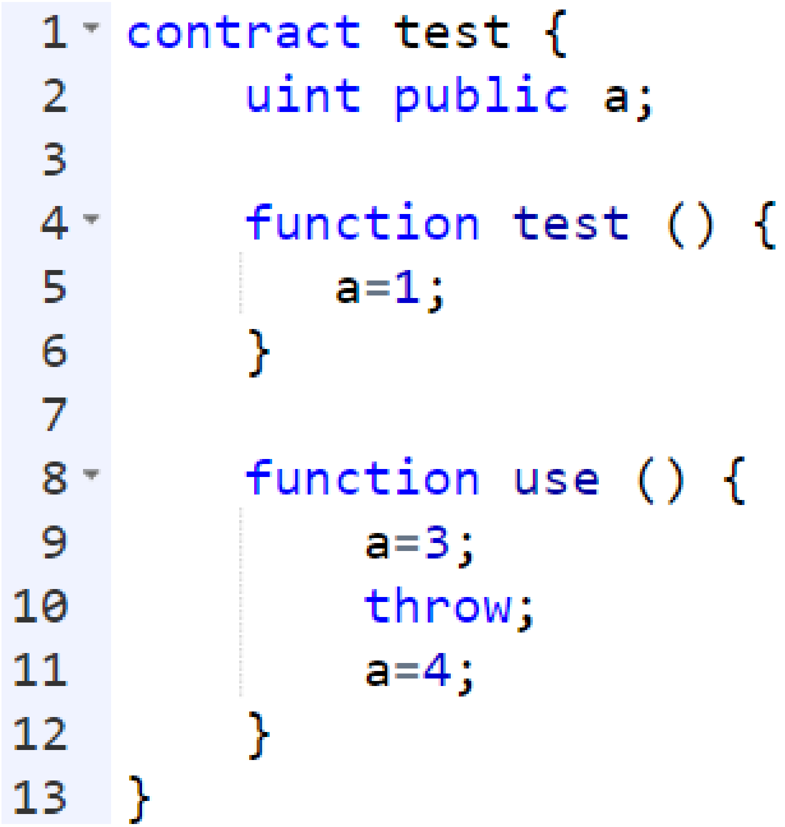
payable 修饰函数时：允许从调用中接收 以太币Ether 。

constant 修饰状态变量时：不允许赋值（除初始化以外）。

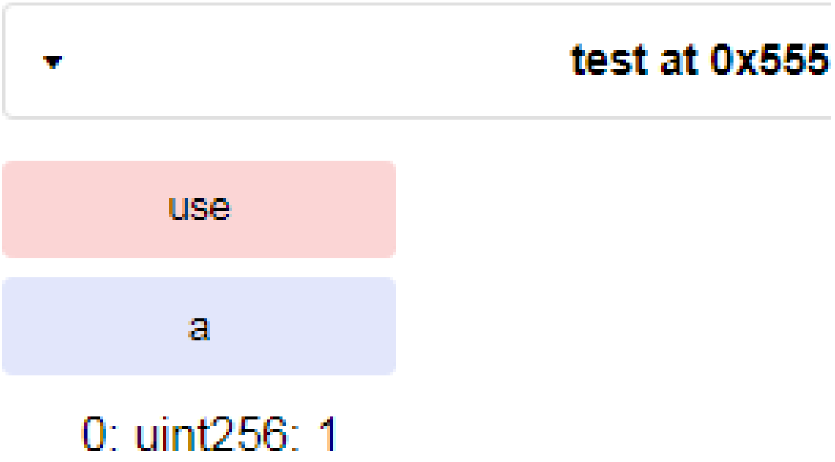
constant 修饰函数时：与 view 等价。

Solidity语言

智能合约代码执行的“原子性”



```
contract test {
    uint public a;

    function test () {
        a=1;
    }

    function use () {
        a=3;
        throw;
        a=4;
    }
}
```



```
0: uint256: 1
```

Solidity语言

■ 映射——Mapping

映射可以视作哈希表，它们在实际的初始化过程中创建每个可能的 key， 并将其映射到字节形式全是零的值：一个类型的 默认值。

映射与哈希表不同的地方： 在映射中，实际上并不存储 key，而是存储它的 keccak256 哈希值，从而便于查询实际的值。

映射是没有长度的，也没有 key 的集合或 value 的集合的概念。

Solidity语言

映射——Mapping

```
pragma solidity ^0.4.0;

contract MappingExample {
    mapping(address => uint) public balances;
    function update(uint newBalance) public {
         balances[msg.sender] = newBalance;
    }
 }

contract MappingUser {
     function f() public returns (uint) {
        MappingExample m = new MappingExample();
        m.update(100);
        return m.balances(this);
    }
}
```

Solidity语言

特殊变量

- block.blockhash(uint blockNumber) returns (bytes32)：指定区块的区块哈希
- block.coinbase (address): 挖出当前区块的矿工地址
- block.difficulty (uint): 当前区块难度
- block.gaslimit (uint): 当前区块 gas 限额
- block.number (uint): 当前区块号
- block.timestamp (uint): 自 unix epoch 起始当前区块以秒计的时间戳
- gasleft() returns (uint256)：剩余的 gas
- msg.data (bytes): 完整的 calldata
- msg.gasleft (uint): 剩余 gas

WWW.INPLUSLAB.COM

Solidity语言

特殊变量

- **msg.sender (address): 消息发送者（当前调用）**
- msg.sig (bytes4): calldata 的前 4 字节（也就是函数标识符）
- msg.value (uint): 随消息发送的 wei 的数量
- **now (uint): 目前区块时间戳（block.timestamp）**
- tx.gasprice (uint): 交易的 gas 价格
- tx.origin (address): 交易发起者（完全的调用链）

Solidity语言

特殊函数

- **assert(bool condition):**

  如果条件不满足，则使当前交易没有效果 — 用于检查内部错误。

- **require(bool condition):**

  如果条件不满足则撤销状态更改 - 用于检查由输入或者外部组件引起的错误。

- **require(bool condition, string message):**

  如果条件不满足则撤销状态更改 - 用于检查由输入或者外部组件引起的错误，可以同时提供一个错误消息。

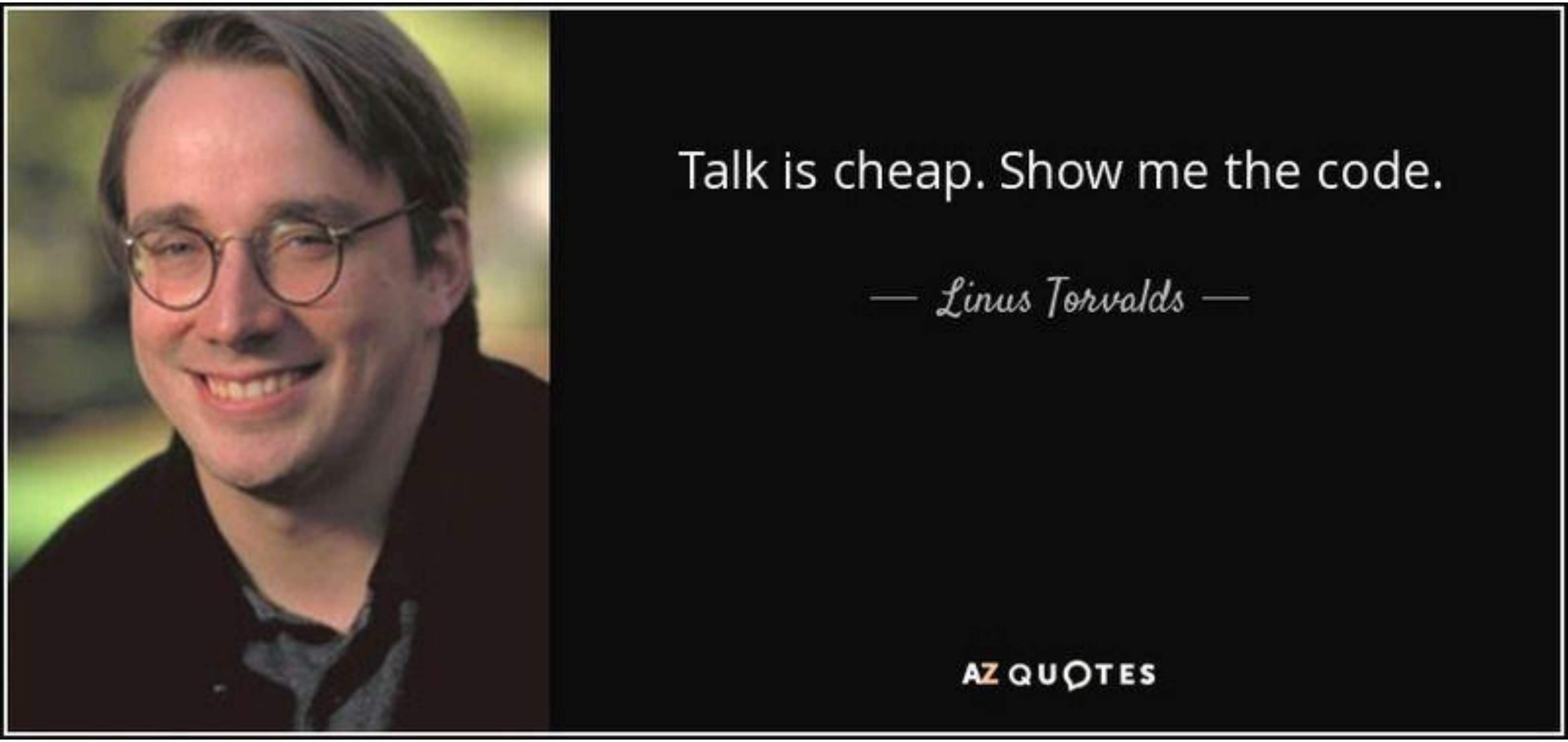
- **revert():**

  终止运行并撤销状态更改。

- **revert(string reason):**

  终止运行并撤销状态更改，可以同时提供一个解释性的字符串。

Solidity语言

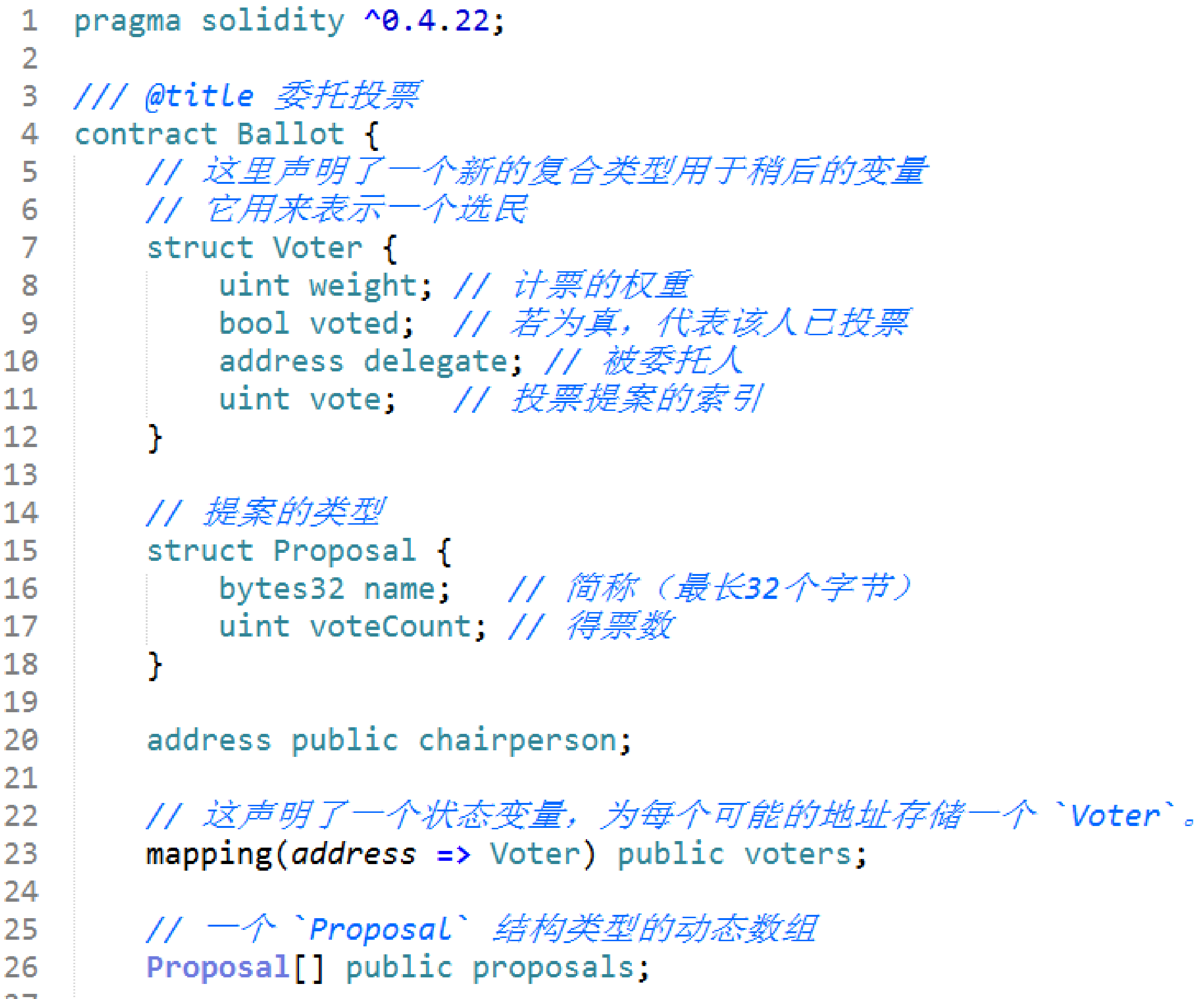


Talk is cheap. Show me the code.

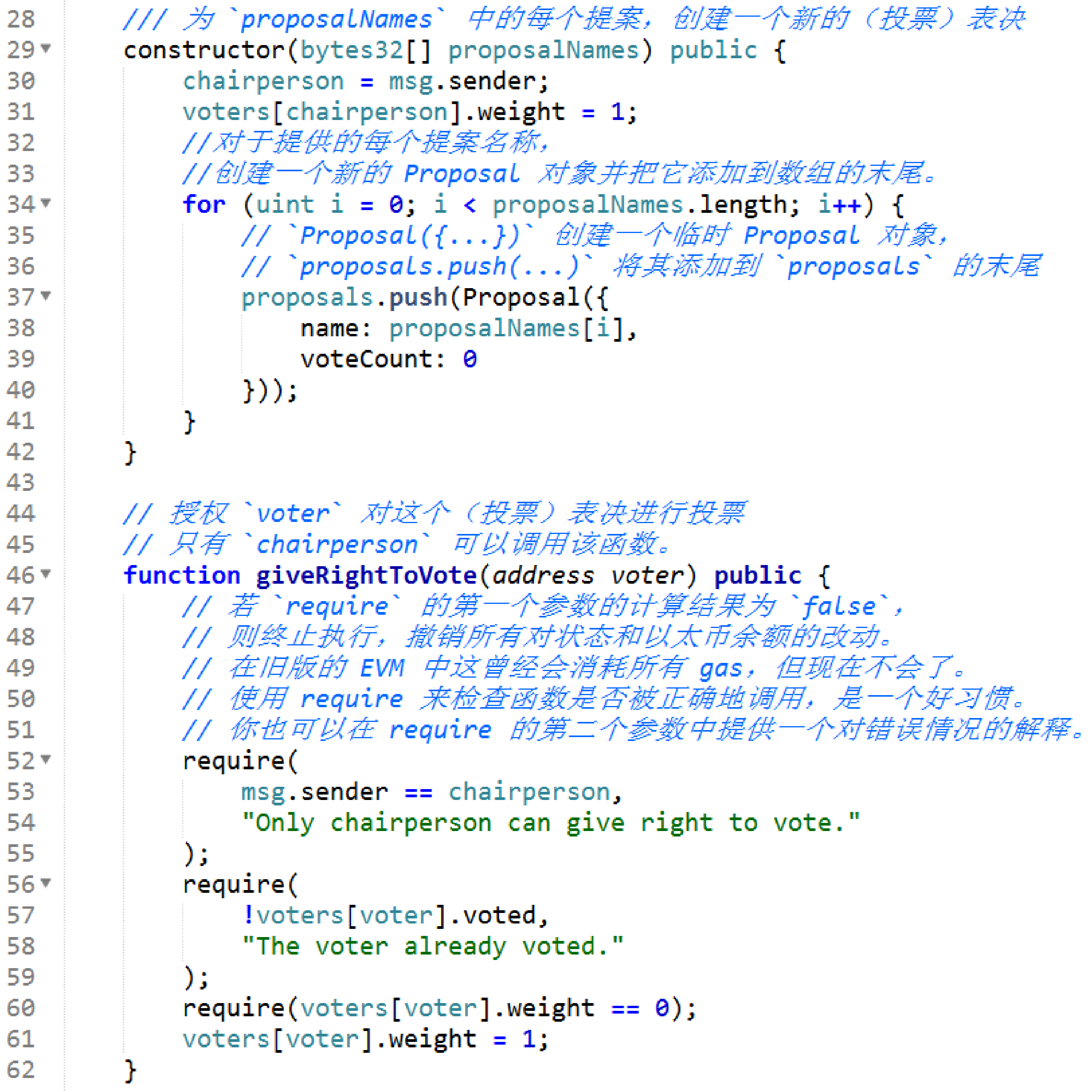
— Linus Torvalds —

Solidity语言

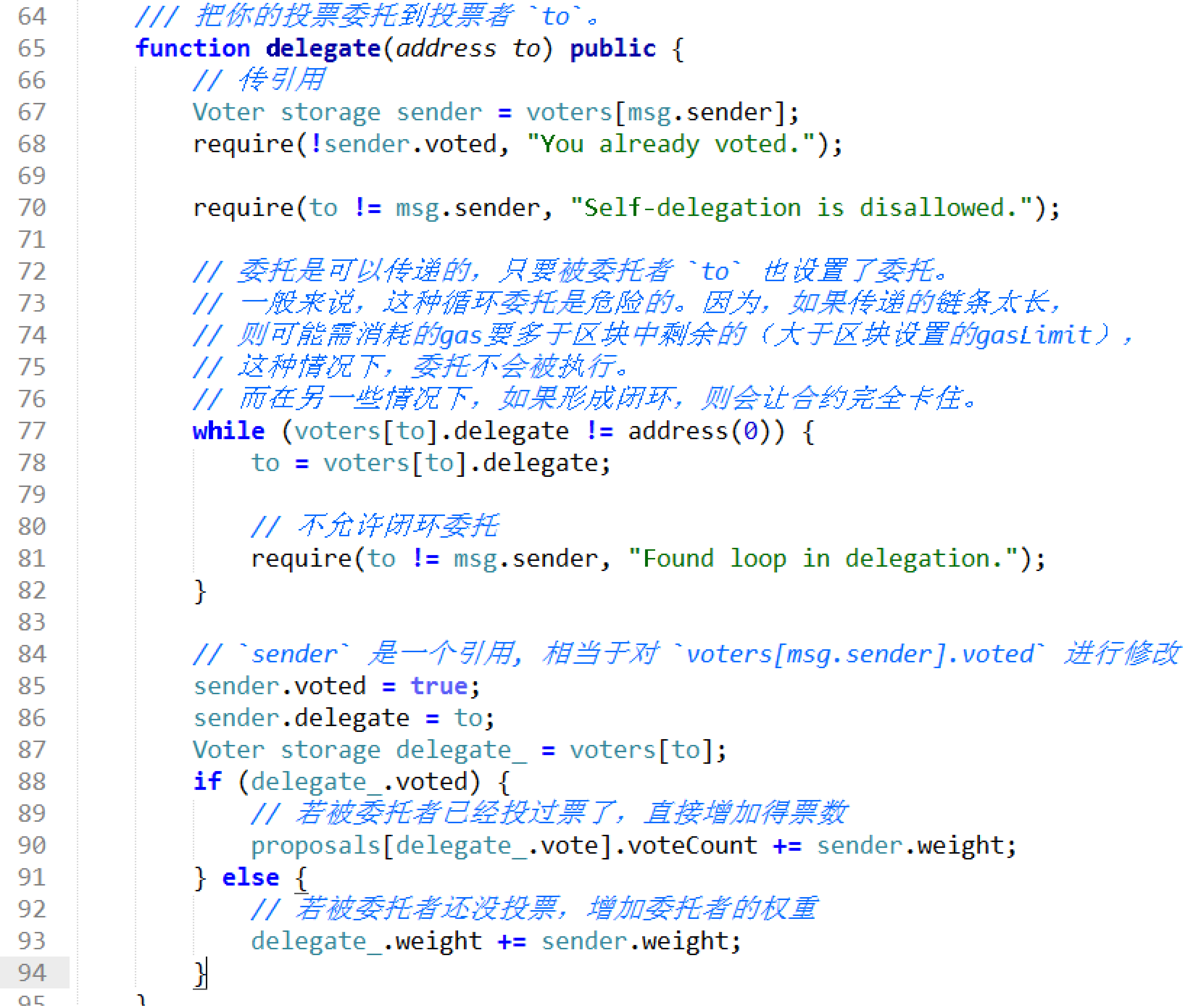
一个例子——Ballot



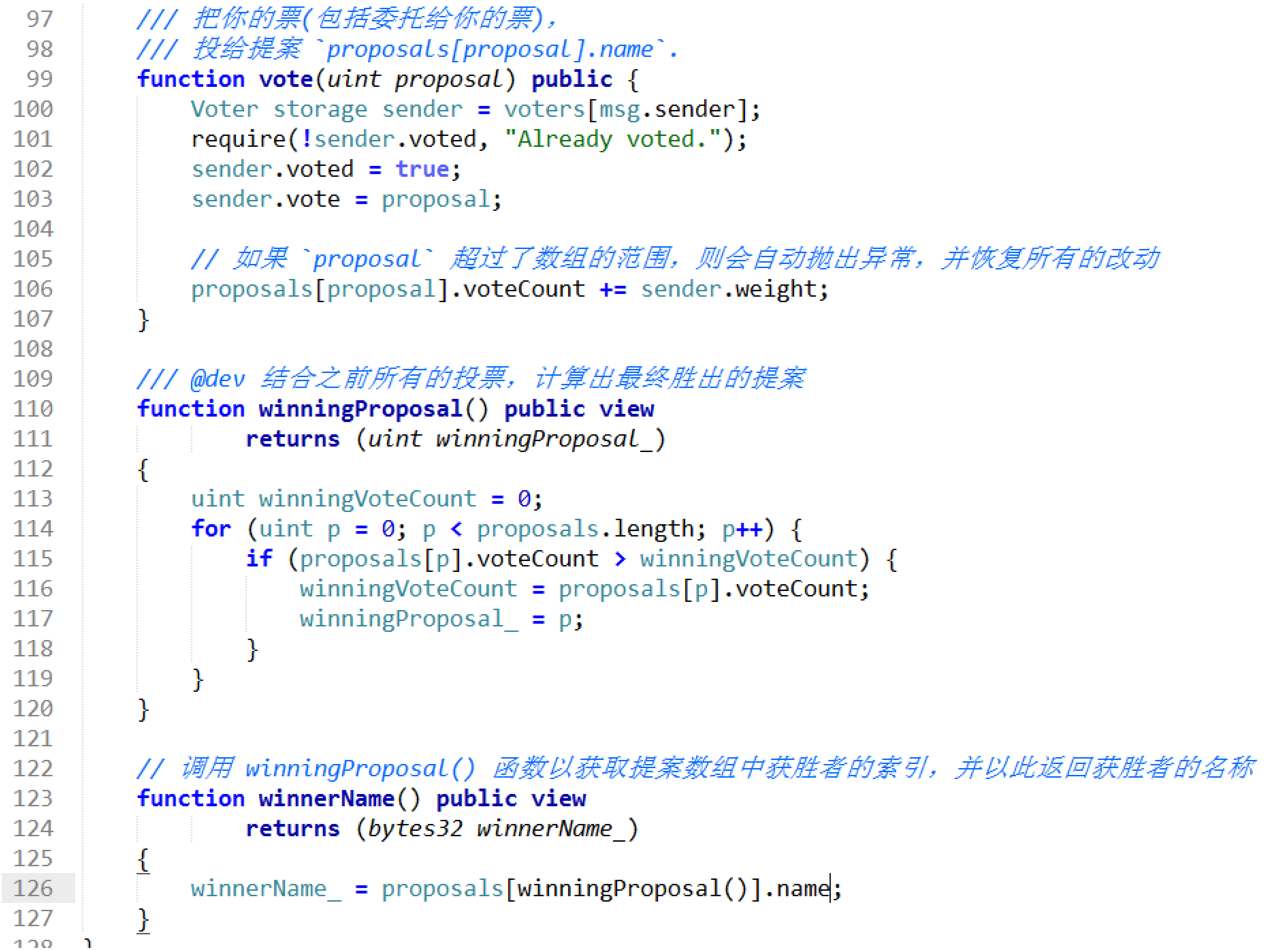
```
pragma solidity ^0.4.22;

/// @title 委托投票
contract Ballot {
    // 这里声明了一个新的复合类型用于稍后的变量
    // 它用来表示一个选民
    struct Voter {
        uint weight; // 计票的权重
        bool voted;  // 若为真，代表该人已投票
        address delegate; // 被委托人
        uint vote;   // 投票提案的索引
    }

    // 提案的类型
    struct Proposal {
        bytes32 name;   // 简称（最长32个字节）
        uint voteCount; // 得票数
    }

    address public chairperson;

    // 这声明了一个状态变量，为每个可能的地址存储一个 `Voter`。
    mapping(address => Voter) public voters;

    // 一个 `Proposal` 结构类型的动态数组
    Proposal[] public proposals;
```



```
    /// 为 `proposalNames` 中的每个提案，创建一个新的（投票）表决
    constructor(bytes32[] proposalNames) public {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
        //对于提供的每个提案名称，
        //创建一个新的 Proposal 对象并把它添加到数组的末尾。
        for (uint i = 0; i < proposalNames.length; i++) {
            // `Proposal({...})` 创建一个临时 Proposal 对象，
            // `proposals.push(...)` 将其添加到 `proposals` 的末尾
            proposals.push(Proposal({
                name: proposalNames[i],
                voteCount: 0
            }));
        }
    }

    // 授权 `voter` 对这个（投票）表决进行投票
    // 只有 `chairperson` 可以调用该函数。
    function giveRightToVote(address voter) public {
        // 若 `require` 的第一个参数的计算结果为 `false`，
        // 则终止执行，撤销所有对状态和以太币余额的改动。
        // 在旧版的 EVM 中这曾经会消耗所有 gas，但现在不会了。
        // 使用 require 来检查函数是否被正确地调用，是一个好习惯。
        // 你也可以在 require 的第二个参数中提供一个对错误情况的解释。
        require(
            msg.sender == chairperson,
            "Only chairperson can give right to vote."
        );
        require(
            !voters[voter].voted,
            "The voter already voted."
        );
        require(voters[voter].weight == 0);
        voters[voter].weight = 1;
    }
```



```
    /// 把你的投票委托到投票者 `to`。
    function delegate(address to) public {
        // 传引用
        Voter storage sender = voters[msg.sender];
        require(!sender.voted, "You already voted.");

        require(to != msg.sender, "Self-delegation is disallowed.");

        // 委托是可以传递的，只要被委托者 `to` 也设置了委托。
        // 一般来说，这种循环委托是危险的。因为，如果传递的链条太长，
        // 则可能需消耗的gas要多于区块中剩余的（大于区块设置的gasLimit），
        // 这种情况下，委托不会被执行。
        // 而在另一些情况下，如果形成闭环，则会让合约完全卡住。
        while (voters[to].delegate != address(0)) {
            to = voters[to].delegate;

            // 不允许闭环委托
            require(to != msg.sender, "Found loop in delegation.");
        }

        // `sender` 是一个引用，相当于对 `voters[msg.sender].voted` 进行修改
        sender.voted = true;
        sender.delegate = to;
        Voter storage delegate_ = voters[to];
        if (delegate_.voted) {
            // 若被委托者已经投过票了，直接增加得票数
            proposals[delegate_.vote].voteCount += sender.weight;
        } else {
            // 若被委托者还没投票，增加委托者的权重
            delegate_.weight += sender.weight;
        }
    }
```



```
    /// 把你的票(包括委托给你的票),
    /// 投给提案 `proposals[proposal].name`.
    function vote(uint proposal) public {
        Voter storage sender = voters[msg.sender];
        require(!sender.voted, "Already voted.");
        sender.voted = true;
        sender.vote = proposal;

        // 如果 `proposal` 超过了数组的范围，则会自动抛出异常，并恢复所有的改动
        proposals[proposal].voteCount += sender.weight;
    }

    /// @dev 结合之前所有的投票，计算出最终胜出的提案
    function winningProposal() public view
            returns (uint winningProposal_)
    {
        uint winningVoteCount = 0;
        for (uint p = 0; p < proposals.length; p++) {
            if (proposals[p].voteCount > winningVoteCount) {
                winningVoteCount = proposals[p].voteCount;
                winningProposal_ = p;
            }
        }
    }

    // 调用 winningProposal() 函数以获取提案数组中获胜者的索引，并以此返回获胜者的名称
    function winnerName() public view
            returns (bytes32 winnerName_)
    {
        winnerName_ = proposals[winningProposal()].name;
    }
}
```

Solidity语言

Event事件

问题：如何获得中间结果，通知客户端该合约被执行?

- 传统做法：Printf() / Console.log()
- Solidity（EVM）：Event（Logs）

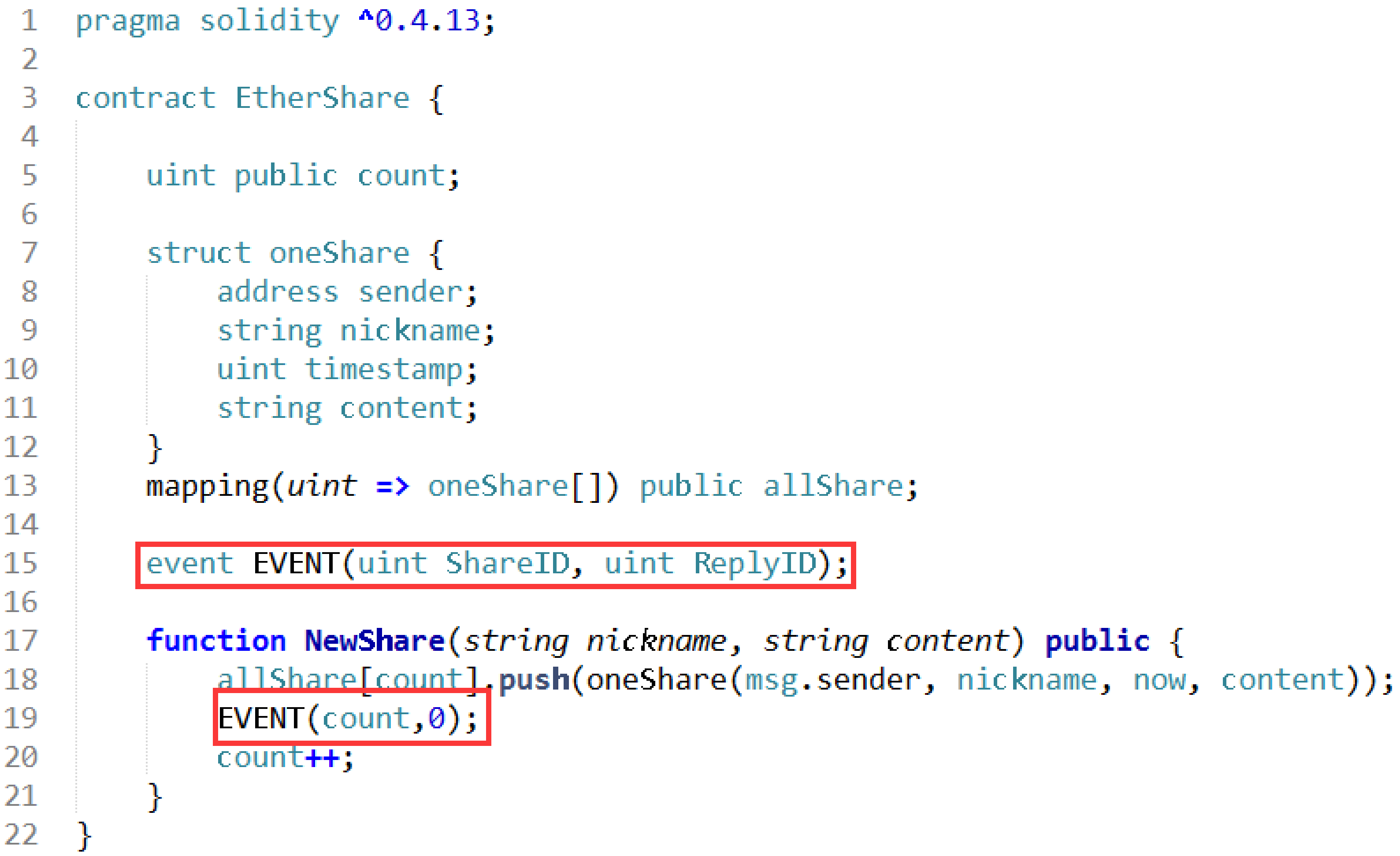
事件和日志存储在交易收据（Transaction Receipts）中，主要用途：

- 帮助用户客户端读取智能合约的返回值（web3.js）；
- 智能合约异步通知用户客户端（web3.js）；
- 用于存储（比Storage便宜得多）；

例子：https://etherscan.io/address/0xc86bdf9661c62646194ef29b1b8f5fe226e8c97e#code

Solidity语言

Event事件



```
pragma solidity ^0.4.13;

contract EtherShare {

    uint public count;

    struct oneShare {
        address sender;
        string nickname;
        uint timestamp;
        string content;
    }
    mapping(uint => oneShare[]) public allShare;

    event EVENT(uint ShareID, uint ReplyID);

    function NewShare(string nickname, string content) public {
        allShare[count].push(oneShare(msg.sender, nickname, now, content));
        EVENT(count,0);
        count++;
    }
}
```

目录

1. 智能合约及平台简介
2. 以太坊基本操作及原理
3. Solidity语言
4. 联盟链智能合约
5. 为智能合约构造图形交互

WWW.INPLUSLAB.COM

联盟链智能合约

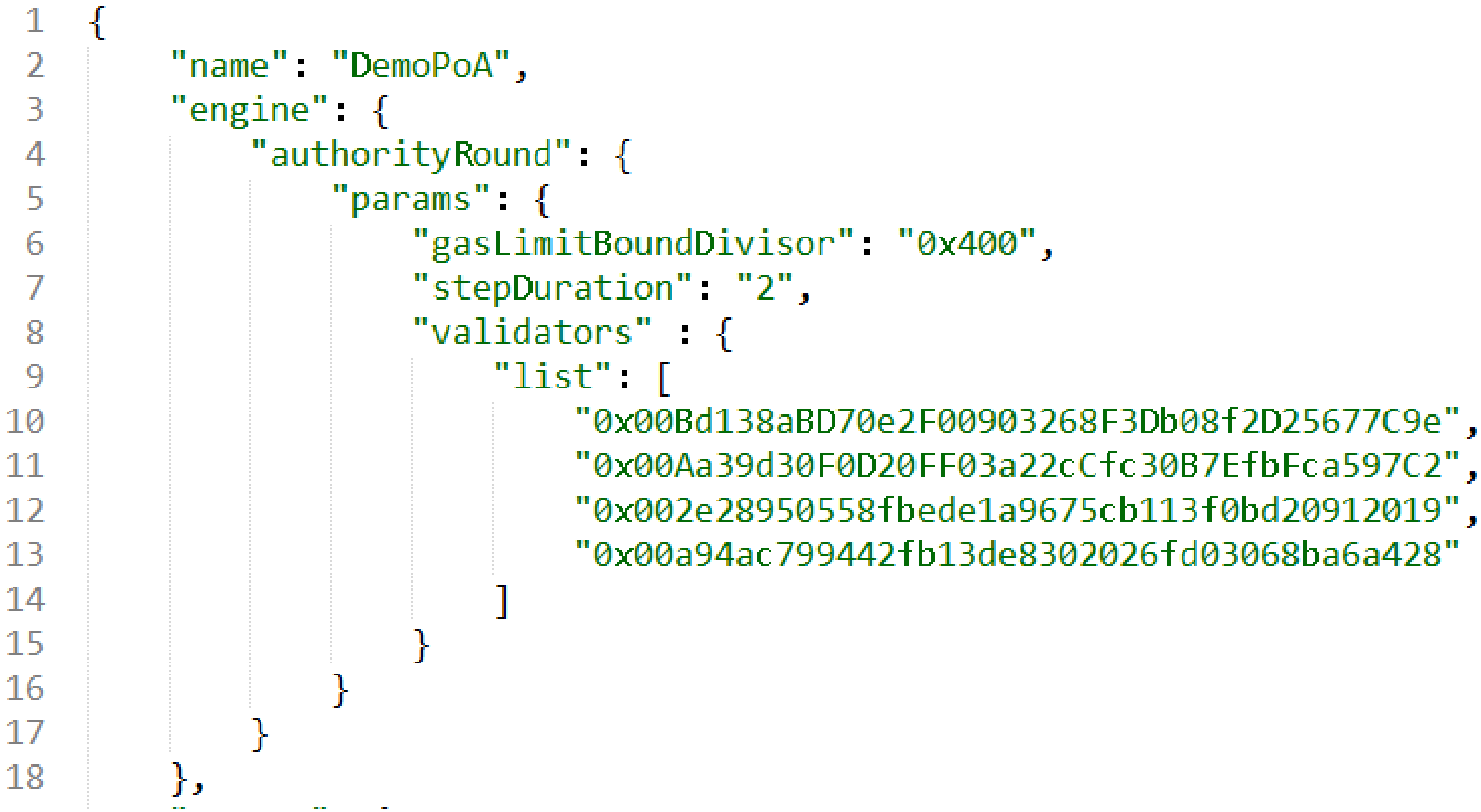
■ 子目录

- 以太坊联盟链（PoA， Parity）
- Hyperledger Fabric 联盟链（chaincode）
- Hyperledger Composer（chaincode中的chaincode）
- 其他联盟链：Hyperchain, CITA,
  FISCO-BCOS, Quorum

联盟链智能合约

以太坊联盟链（PoA，Parity）

- ➢ 与PoW公链的不同：权威验证者、出块时间

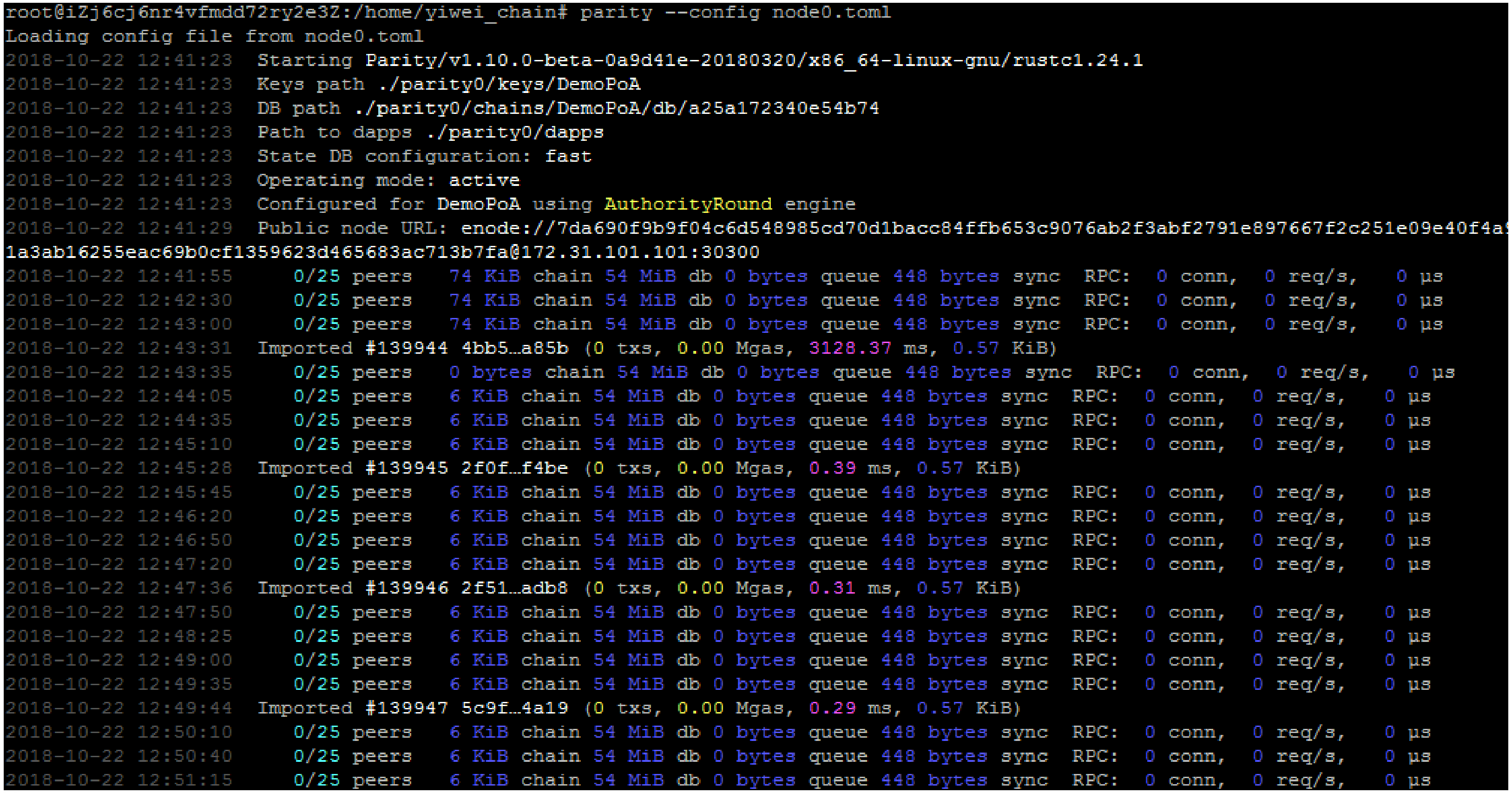


```
{
    "name": "DemoPoA",
    "engine": {
        "authorityRound": {
            "params": {
                "gasLimitBoundDivisor": "0x400",
                "stepDuration": "2",
                "validators" : {
                    "list": [
                        "0x00Bd138aBD70e2F00903268F3Db08f2D25677C9e",
                        "0x00Aa39d30F0D20FF03a22cCfc30B7EfbFca597C2",
                        "0x002e28950558fbede1a9675cb113f0bd20912019",
                        "0x00a94ac799442fb13de8302026fd03068ba6a428"
                    ]
                }
            }
        }
    },
```

联盟链智能合约

■ 以太坊联盟链（PoA，Parity）

- ➢ 与PoW公链的不同：权威验证者、出块时间

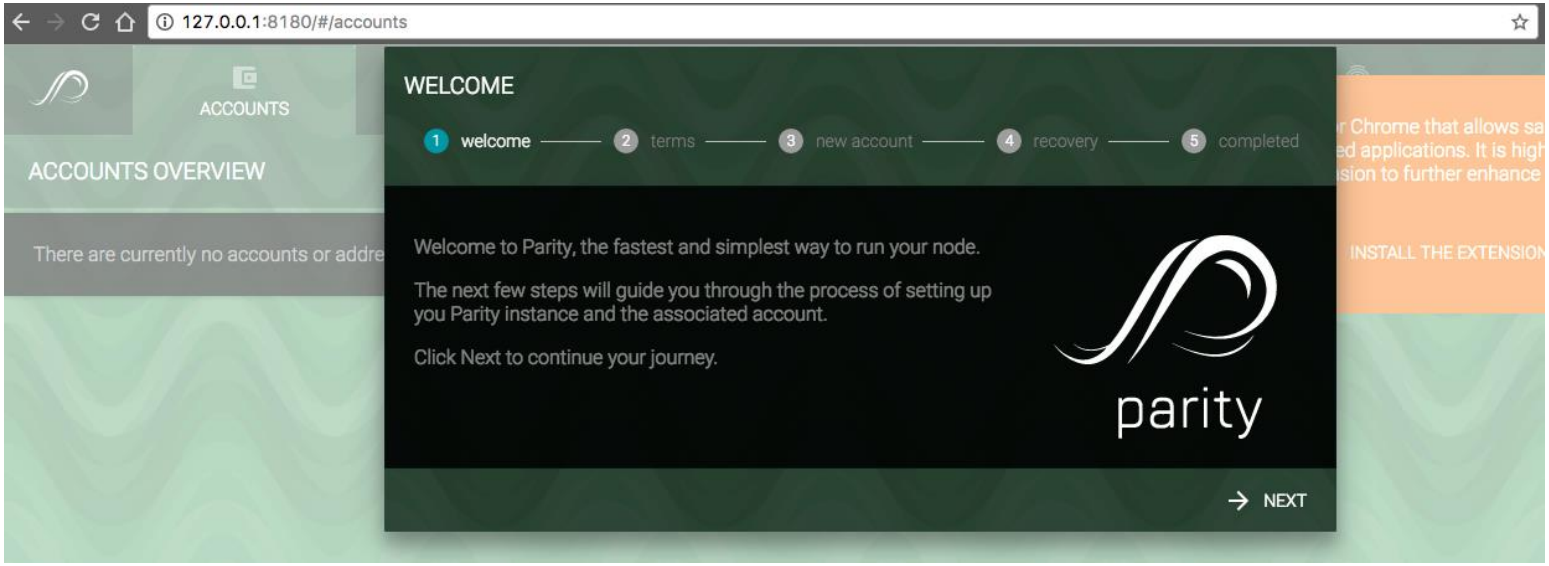


```
root@iZj6cj6nr4vfmdd72ry2e3Z:/home/yiwei_chain# parity --config node0.toml
Loading config file from node0.toml
2018-10-22 12:41:23  Starting Parity/v1.10.0-beta-0a9d41e-20180320/x86_64-linux-gnu/rustc1.24.1
2018-10-22 12:41:23  Keys path ./parity0/keys/DemoPoA
2018-10-22 12:41:23  DB path ./parity0/chains/DemoPoA/db/a25a172340e54b74
2018-10-22 12:41:23  Path to dapps ./parity0/dapps
2018-10-22 12:41:23  State DB configuration: fast
2018-10-22 12:41:23  Operating mode: active
2018-10-22 12:41:23  Configured for DemoPoA using AuthorityRound engine
2018-10-22 12:41:29  Public node URL: enode://7da690f9b9f04c6d548985cd70d1bacc84ffb653c9076ab2f3abf2791e897667f2c251e09e40f4a
1a3ab16255eac69b0cf1359623d465683ac713b7fa@172.31.101.101:30300
2018-10-22 12:41:55     0/25 peers   74 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:42:30     0/25 peers   74 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:43:00     0/25 peers   74 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:43:31  Imported #139944 4bb5…a85b (0 txs, 0.00 Mgas, 3128.37 ms, 0.57 KiB)
2018-10-22 12:43:35     0/25 peers   0 bytes chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:44:05     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:44:35     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:45:10     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:45:28  Imported #139945 2f0f…f4be (0 txs, 0.00 Mgas, 0.39 ms, 0.57 KiB)
2018-10-22 12:45:45     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:46:20     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:46:50     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:47:20     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:47:36  Imported #139946 2f51…adb8 (0 txs, 0.00 Mgas, 0.31 ms, 0.57 KiB)
2018-10-22 12:47:50     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:48:25     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:49:00     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:49:35     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:49:44  Imported #139947 5c9f…4a19 (0 txs, 0.00 Mgas, 0.29 ms, 0.57 KiB)
2018-10-22 12:50:10     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:50:40     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
2018-10-22 12:51:15     0/25 peers   6 KiB chain 54 MiB db 0 bytes queue 448 bytes sync  RPC:  0 conn,  0 req/s,   0 µs
```

联盟链智能合约

■ 以太坊联盟链（PoA，Parity）

- ➢ 创建并设置权威节点账户
- ➢ 兼容公有链智能合约、Web3.js调用等
- ➢ 交易需要设置GasLimit 但是GasUsed为0
- ➢ 新建账户等操作具有图形界面



Welcome to Parity, the fastest and simplest way to run your node.

The next few steps will guide you through the process of setting up you Parity instance and the associated account.

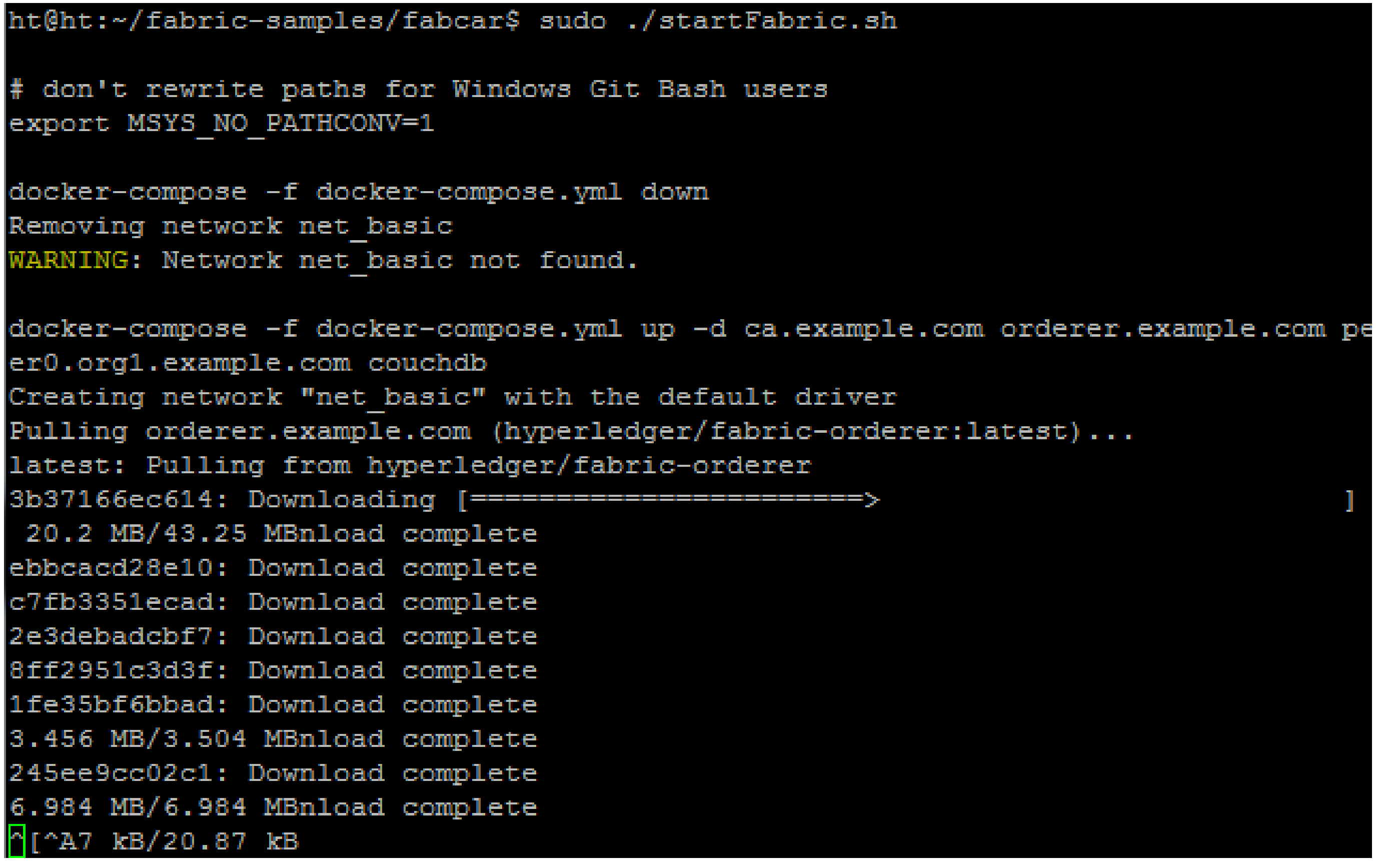
Click Next to continue your journey.

https://medium.com/taipei-ethereum-meetup/%E4%BD%BF%E7%94%A8parity%E5%BB%BA%E7%AB%8Bproof-of-authority-poa-ethereum-chain-c5c1cdd0f21a

联盟链智能合约

Hyperledger Fabric

基于docker的节点部署、chaincode部署

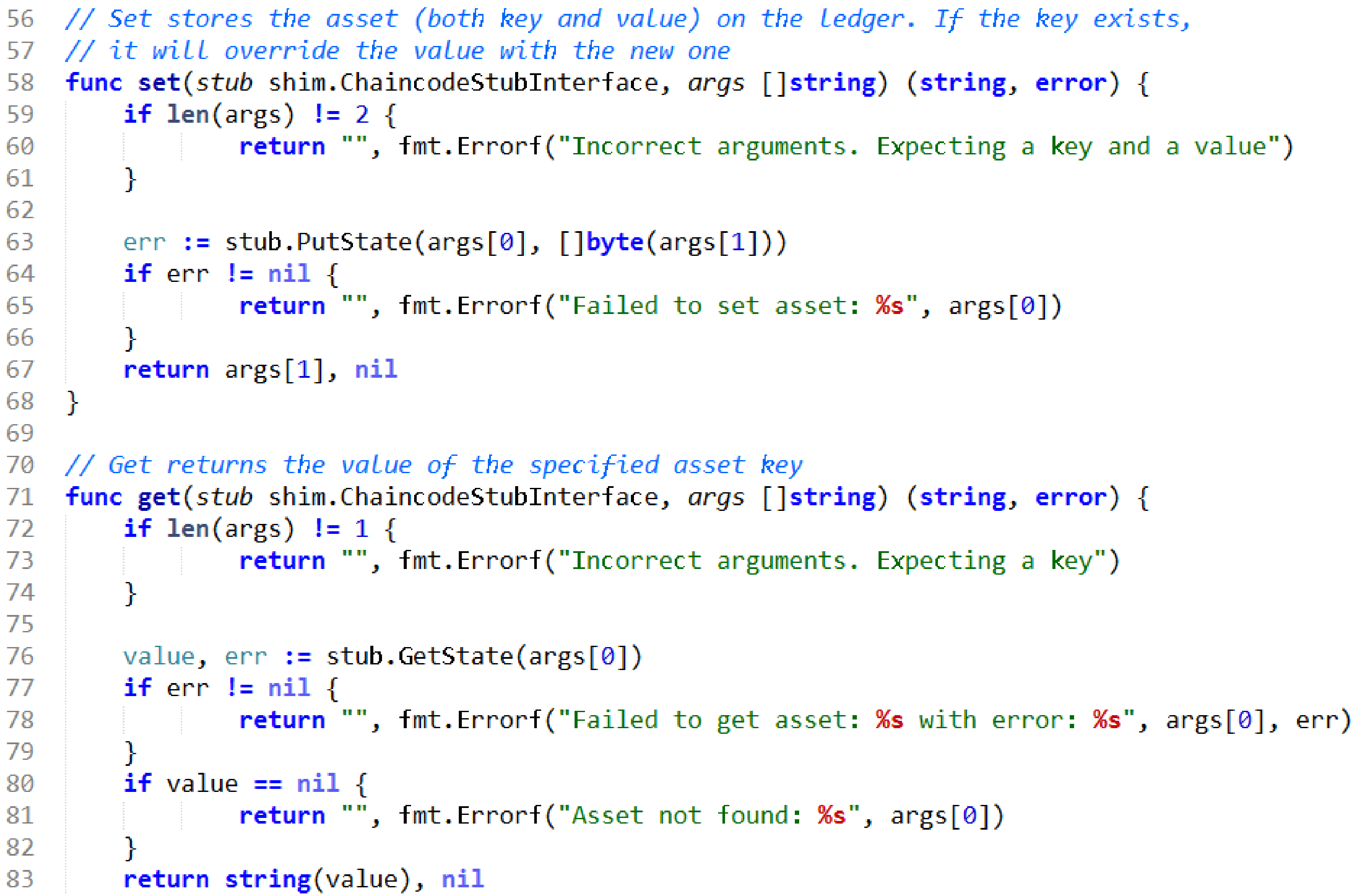


```
ht@ht:~/fabric-samples/fabcar$ sudo ./startFabric.sh

# don't rewrite paths for Windows Git Bash users
export MSYS_NO_PATHCONV=1

docker-compose -f docker-compose.yml down
Removing network net_basic
WARNING: Network net_basic not found.

docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com pe
er0.org1.example.com couchdb
Creating network "net_basic" with the default driver
Pulling orderer.example.com (hyperledger/fabric-orderer:latest)...
latest: Pulling from hyperledger/fabric-orderer
3b37166ec614: Downloading [=======================>                          ]
 20.2 MB/43.25 MBnload complete
ebbcacd28e10: Download complete
c7fb3351ecad: Download complete
2e3debadcbf7: Download complete
8ff2951c3d3f: Download complete
1fe35bf6bbad: Download complete
3.456 MB/3.504 MBnload complete
245ee9cc02c1: Download complete
6.984 MB/6.984 MBnload complete
^[^A7 kB/20.87 kB
```

联盟链智能合约

Hyperledger Fabric （存储合约示例）

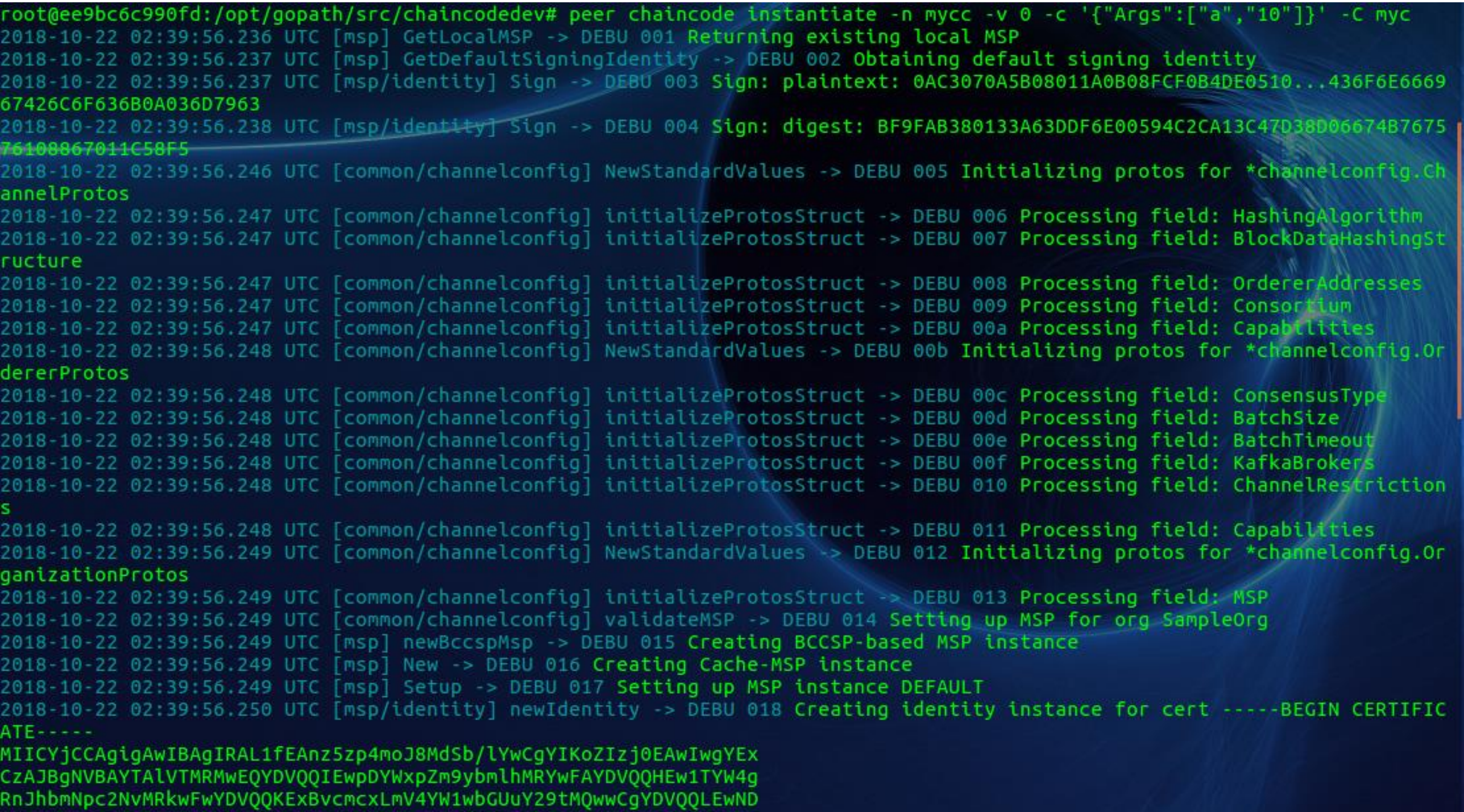
https://hyperledgercn.github.io/hyperledgerDocs/chaincode_developers_zh/



```go
// Set stores the asset (both key and value) on the ledger. If the key exists,
// it will override the value with the new one
func set(stub shim.ChaincodeStubInterface, args []string) (string, error) {
    if len(args) != 2 {
        return "", fmt.Errorf("Incorrect arguments. Expecting a key and a value")
    }

    err := stub.PutState(args[0], []byte(args[1]))
    if err != nil {
        return "", fmt.Errorf("Failed to set asset: %s", args[0])
    }
    return args[1], nil
}

// Get returns the value of the specified asset key
func get(stub shim.ChaincodeStubInterface, args []string) (string, error) {
    if len(args) != 1 {
        return "", fmt.Errorf("Incorrect arguments. Expecting a key")
    }

    value, err := stub.GetState(args[0])
    if err != nil {
        return "", fmt.Errorf("Failed to get asset: %s with error: %s", args[0], err)
    }
    if value == nil {
        return "", fmt.Errorf("Asset not found: %s", args[0])
    }
    return string(value), nil
```

联盟链智能合约

■ Hyperledger Fabric （存储合约示例）

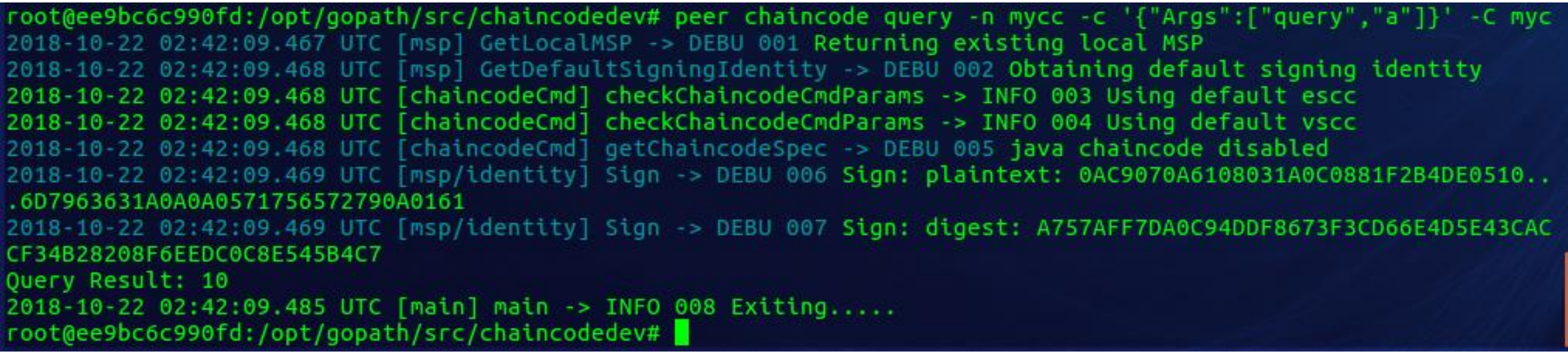


```
root@ee9bc6c990fd:/opt/gopath/src/chaincodedev# peer chaincode instantiate -n mycc -v 0 -c '{"Args":["a","10"]}' -C myc
2018-10-22 02:39:56.236 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-10-22 02:39:56.237 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default signing identity
2018-10-22 02:39:56.237 UTC [msp/identity] Sign -> DEBU 003 Sign: plaintext: 0AC3070A5B08011A0B08FCF0B4DE0510...436F6E6669
67426C6F636B0A036D7963
2018-10-22 02:39:56.238 UTC [msp/identity] Sign -> DEBU 004 Sign: digest: BF9FAB380133A63DDF6E00594C2CA13C47D38D06674B7675
76108867011C58F5
2018-10-22 02:39:56.246 UTC [common/channelconfig] NewStandardValues -> DEBU 005 Initializing protos for *channelconfig.Ch
annelProtos
2018-10-22 02:39:56.247 UTC [common/channelconfig] initializeProtosStruct -> DEBU 006 Processing field: HashingAlgorithm
2018-10-22 02:39:56.247 UTC [common/channelconfig] initializeProtosStruct -> DEBU 007 Processing field: BlockDataHashingSt
ructure
2018-10-22 02:39:56.247 UTC [common/channelconfig] initializeProtosStruct -> DEBU 008 Processing field: OrdererAddresses
2018-10-22 02:39:56.247 UTC [common/channelconfig] initializeProtosStruct -> DEBU 009 Processing field: Consortium
2018-10-22 02:39:56.247 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00a Processing field: Capabilities
2018-10-22 02:39:56.248 UTC [common/channelconfig] NewStandardValues -> DEBU 00b Initializing protos for *channelconfig.Or
dererProtos
2018-10-22 02:39:56.248 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00c Processing field: ConsensusType
2018-10-22 02:39:56.248 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00d Processing field: BatchSize
2018-10-22 02:39:56.248 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00e Processing field: BatchTimeout
2018-10-22 02:39:56.248 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00f Processing field: KafkaBrokers
2018-10-22 02:39:56.248 UTC [common/channelconfig] initializeProtosStruct -> DEBU 010 Processing field: ChannelRestriction
s
2018-10-22 02:39:56.248 UTC [common/channelconfig] initializeProtosStruct -> DEBU 011 Processing field: Capabilities
2018-10-22 02:39:56.249 UTC [common/channelconfig] NewStandardValues -> DEBU 012 Initializing protos for *channelconfig.Or
ganizationProtos
2018-10-22 02:39:56.249 UTC [common/channelconfig] initializeProtosStruct -> DEBU 013 Processing field: MSP
2018-10-22 02:39:56.249 UTC [common/channelconfig] validateMSP -> DEBU 014 Setting up MSP for org SampleOrg
2018-10-22 02:39:56.249 UTC [msp] newBccspMsp -> DEBU 015 Creating BCCSP-based MSP instance
2018-10-22 02:39:56.249 UTC [msp] New -> DEBU 016 Creating Cache-MSP instance
2018-10-22 02:39:56.249 UTC [msp] Setup -> DEBU 017 Setting up MSP instance DEFAULT
2018-10-22 02:39:56.250 UTC [msp/identity] newIdentity -> DEBU 018 Creating identity instance for cert -----BEGIN CERTIFIC
ATE-----
MIICYjCCAgigAwIBAgIRAL1fEAnz5zp4moJ8MdSb/lYwCgYIKoZIzj0EAwIwgYEx
CzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g
RnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMQwwCgYDVQQLEwND
```

联盟链智能合约

Hyperledger Fabric （存储合约示例）

查询合约

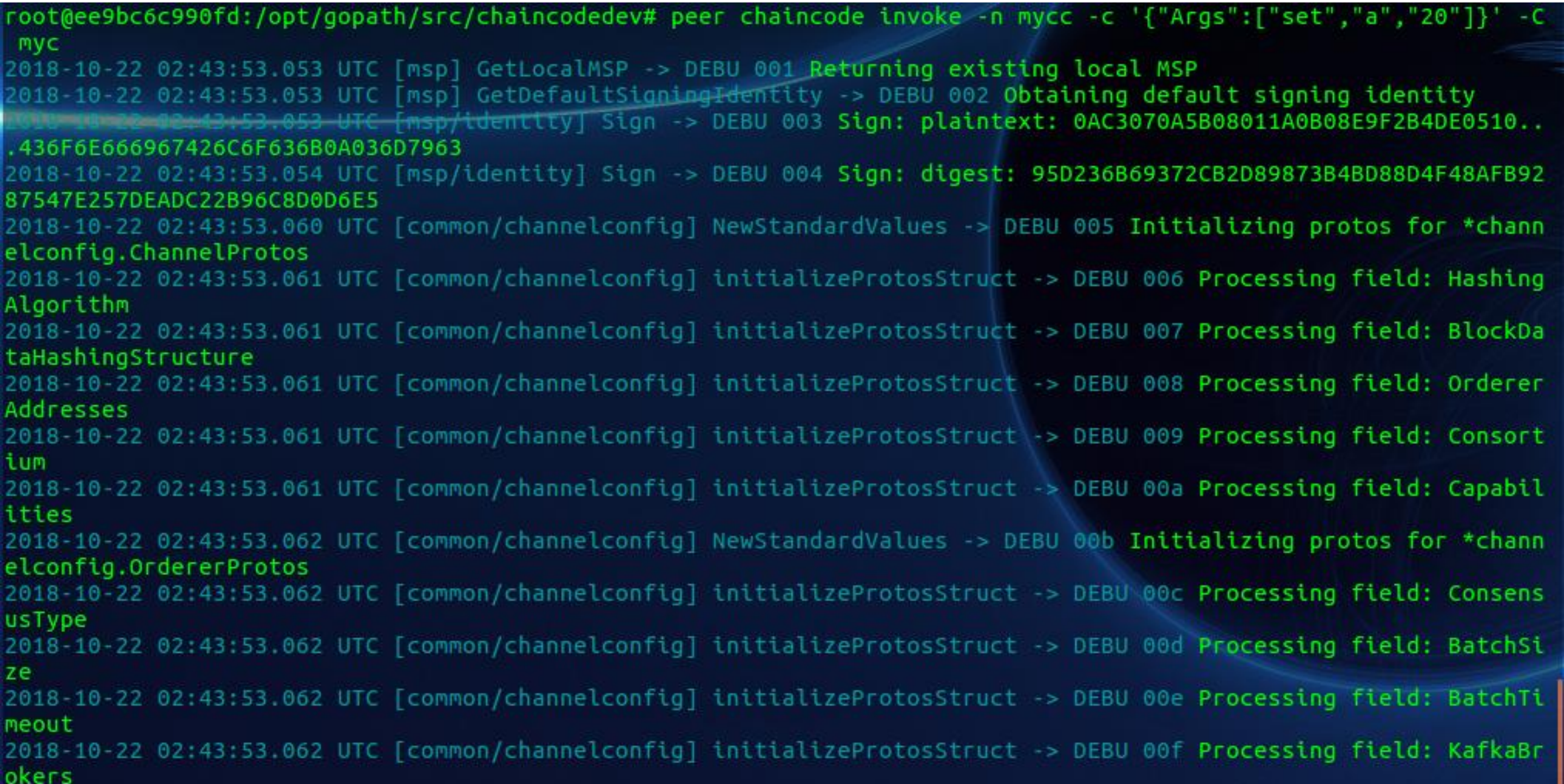


```
root@ee9bc6c990fd:/opt/gopath/src/chaincodedev# peer chaincode query -n mycc -c '{"Args":["query","a"]}' -C myc
2018-10-22 02:42:09.467 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-10-22 02:42:09.468 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default signing identity
2018-10-22 02:42:09.468 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using default escc
2018-10-22 02:42:09.468 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-10-22 02:42:09.468 UTC [chaincodeCmd] getChaincodeSpec -> DEBU 005 java chaincode disabled
2018-10-22 02:42:09.469 UTC [msp/identity] Sign -> DEBU 006 Sign: plaintext: 0AC9070A6108031A0C0881F2B4DE0510..
.6D7963631A0A0A0571756572790A0161
2018-10-22 02:42:09.469 UTC [msp/identity] Sign -> DEBU 007 Sign: digest: A757AFF7DA0C94DDF8673F3CD66E4D5E43CAC
CF34B28208F6EEDC0C8E545B4C7
Query Result: 10
2018-10-22 02:42:09.485 UTC [main] main -> INFO 008 Exiting.....
root@ee9bc6c990fd:/opt/gopath/src/chaincodedev# █
```

联盟链智能合约

Hyperledger Fabric （存储合约示例）

重新赋值（1）

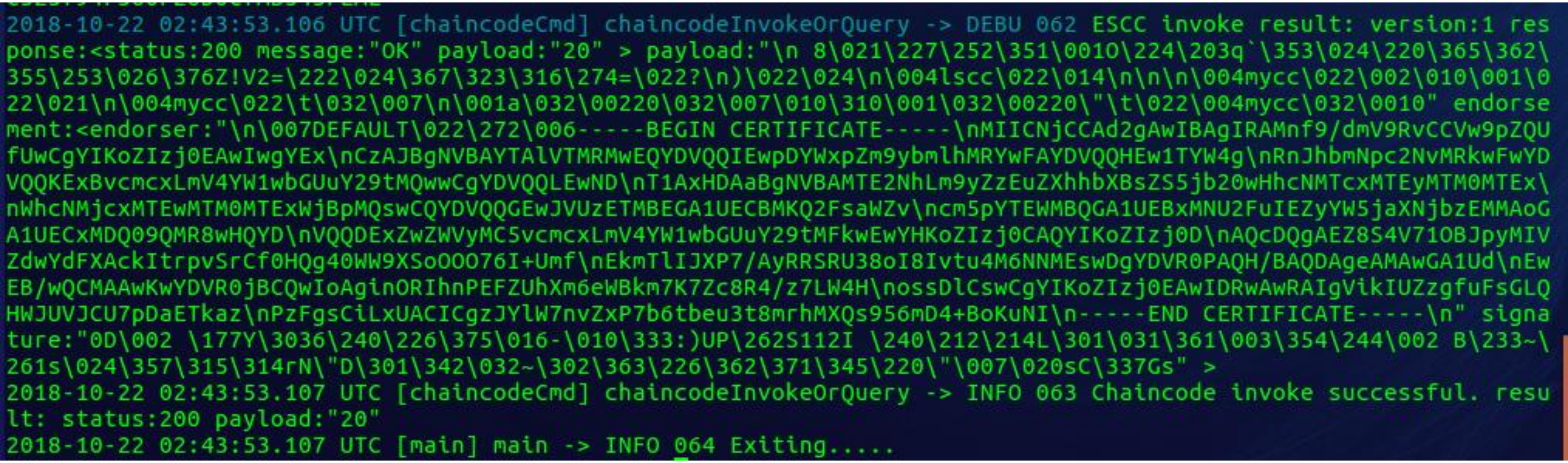


```
root@ee9bc6c990fd:/opt/gopath/src/chaincodedev# peer chaincode invoke -n mycc -c '{"Args":["set","a","20"]}' -C
 myc
2018-10-22 02:43:53.053 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-10-22 02:43:53.053 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default signing identity
2018-10-22 02:43:53.053 UTC [msp/identity] Sign -> DEBU 003 Sign: plaintext: 0AC3070A5B08011A0B08E9F2B4DE0510..
.436F6E666967426C6F636B0A036D7963
2018-10-22 02:43:53.054 UTC [msp/identity] Sign -> DEBU 004 Sign: digest: 95D236B69372CB2D89873B4BD88D4F48AFB92
87547E257DEADC22B96C8D0D6E5
2018-10-22 02:43:53.060 UTC [common/channelconfig] NewStandardValues -> DEBU 005 Initializing protos for *chann
elconfig.ChannelProtos
2018-10-22 02:43:53.061 UTC [common/channelconfig] initializeProtosStruct -> DEBU 006 Processing field: Hashing
Algorithm
2018-10-22 02:43:53.061 UTC [common/channelconfig] initializeProtosStruct -> DEBU 007 Processing field: BlockDa
taHashingStructure
2018-10-22 02:43:53.061 UTC [common/channelconfig] initializeProtosStruct -> DEBU 008 Processing field: Orderer
Addresses
2018-10-22 02:43:53.061 UTC [common/channelconfig] initializeProtosStruct -> DEBU 009 Processing field: Consort
ium
2018-10-22 02:43:53.061 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00a Processing field: Capabil
ities
2018-10-22 02:43:53.062 UTC [common/channelconfig] NewStandardValues -> DEBU 00b Initializing protos for *chann
elconfig.OrdererProtos
2018-10-22 02:43:53.062 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00c Processing field: Consens
usType
2018-10-22 02:43:53.062 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00d Processing field: BatchSi
ze
2018-10-22 02:43:53.062 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00e Processing field: BatchTi
meout
2018-10-22 02:43:53.062 UTC [common/channelconfig] initializeProtosStruct -> DEBU 00f Processing field: KafkaBr
okers
```

联盟链智能合约

Hyperledger Fabric （存储合约示例）

重新赋值（2）

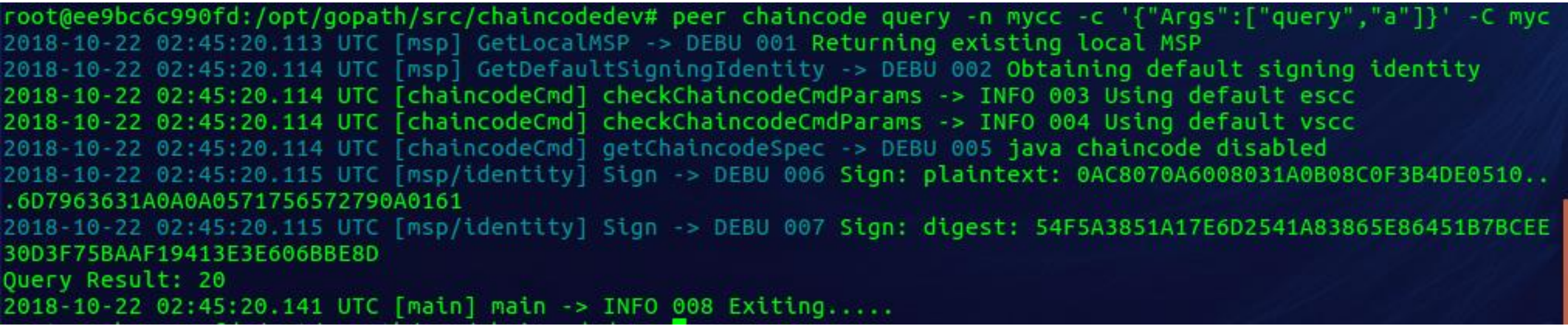


```
2018-10-22 02:43:53.106 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> DEBU 062 ESCC invoke result: version:1 res
ponse:<status:200 message:"OK" payload:"20" > payload:"\n 8\021\227\252\351\001O\224\203q`\353\024\220\365\362\
355\253\026\376Z!V2=\222\024\367\323\316\274=\022?\n)\022\024\n\004lscc\022\014\n\n\n\004mycc\022\002\010\001\0
22\021\n\004mycc\022\t\032\007\n\001a\032\00220\032\007\010\310\001\032\00220\"\t\022\004mycc\032\0010" endorse
ment:<endorser:"\n\007DEFAULT\022\272\006-----BEGIN CERTIFICATE-----\nMIICNjCCAd2gAwIBAgIRAMnf9/dmV9RvCCVw9pZQU
fUwCgYIKoZIzj0EAwIwgYEx\nCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g\nRnJhbmNpc2NvMRkwFwYD
VQQKExBvcmcxLmV4YW1wbGUuY29tMQwwCgYDVQQLEwND\nT1AxHDAaBgNVBAMTE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcxMTEyMTM0MTEx\
nWhcNMjcxMTEwMTM0MTExWjBpMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv\ncm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEMMAoG
A1UECxMDQ09QMR8wHQYD\nVQQDExZwZWVyMC5vcmcxLmV4YW1wbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0D\nAQcDQgAEZ8S4V71OBJpyMIV
ZdwYdFXAckItrpvSrCf0HQg40WW9XSoOOO76I+Umf\nEkmTlIJXP7/AyRRSRU38oI8Ivtu4M6NNMEswDgYDVR0PAQH/BAQDAgeAMAwGA1Ud\nEw
EB/wQCMAAwKwYDVR0jBCQwIoAginORIhnPEFZUhXm6eWBkm7K7Zc8R4/z7LW4H\nossDlCswCgYIKoZIzj0EAwIDRwAwRAIgVikIUZzgfuFsGLQ
HWJUVJCU7pDaETkaz\nPzFgsCiLxUACICgzJYlW7nvZxP7b6tbeu3t8mrhMXQs956mD4+BoKuNI\n-----END CERTIFICATE-----\n" signa
ture:"0D\002 \177Y\3036\240\226\375\016-\010\333:)UP\262S112I \240\212\214L\301\031\361\003\354\244\002 B\233~\
261s\024\357\315\314rN\"D\301\342\032~\302\363\226\362\371\345\220\"\007\020sC\337Gs" >
2018-10-22 02:43:53.107 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 063 Chaincode invoke successful. resu
lt: status:200 payload:"20"
2018-10-22 02:43:53.107 UTC [main] main -> INFO 064 Exiting.....
```

联盟链智能合约

Hyperledger Fabric （存储合约示例）

重新查询



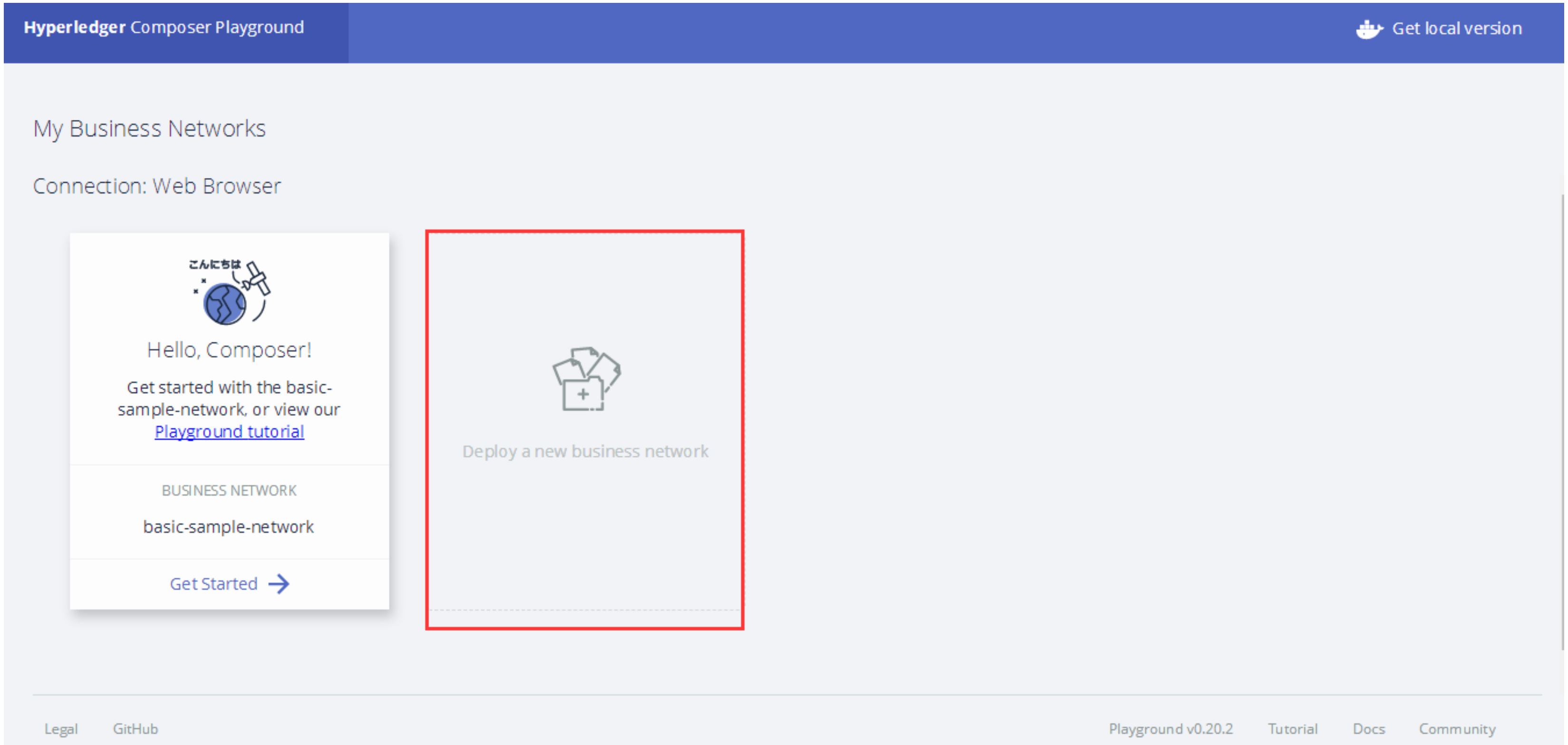
```
root@ee9bc6c990fd:/opt/gopath/src/chaincodedev# peer chaincode query -n mycc -c '{"Args":["query","a"]}' -C myc
2018-10-22 02:45:20.113 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-10-22 02:45:20.114 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default signing identity
2018-10-22 02:45:20.114 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using default escc
2018-10-22 02:45:20.114 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-10-22 02:45:20.114 UTC [chaincodeCmd] getChaincodeSpec -> DEBU 005 java chaincode disabled
2018-10-22 02:45:20.115 UTC [msp/identity] Sign -> DEBU 006 Sign: plaintext: 0AC8070A6008031A0B08C0F3B4DE0510..
.6D7963631A0A0A0571756572790A0161
2018-10-22 02:45:20.115 UTC [msp/identity] Sign -> DEBU 007 Sign: digest: 54F5A3851A17E6D2541A83865E86451B7BCEE
30D3F75BAAF19413E3E606BBE8D
Query Result: 20
2018-10-22 02:45:20.141 UTC [main] main -> INFO 008 Exiting.....
```

联盟链智能合约

Hyperledger Composer

官网 https://www.hyperledger.org/projects/composer

在线试用 https://composer-playground.mybluemix.net/login



My Business Networks

Connection: Web Browser

Hello, Composer!

Get started with the basic-sample-network, or view our Playground tutorial

BUSINESS NETWORK

basic-sample-network

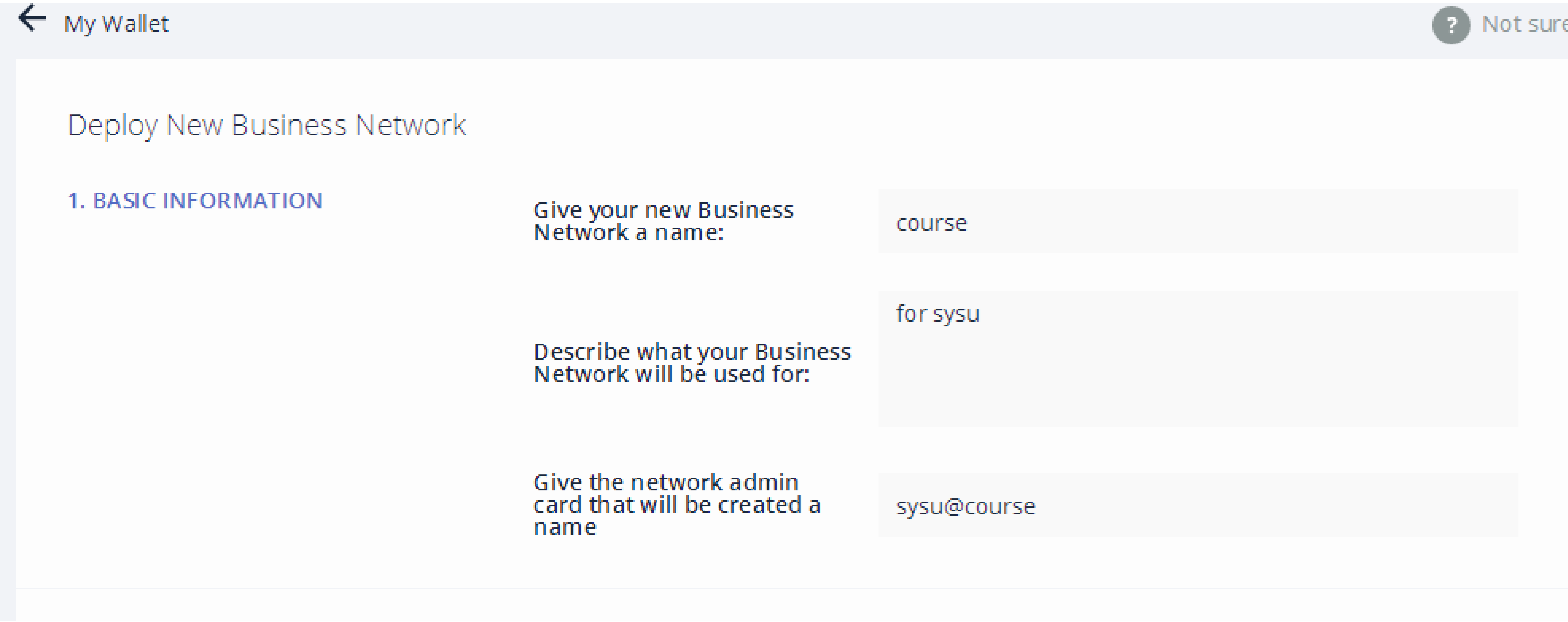
Get Started →

Deploy a new business network

联盟链智能合约

Hyperledger Composer

创建商业网络（可涉及角色创建、证书分发等）



Deploy New Business Network

1. BASIC INFORMATION

Give your new Business Network a name: course

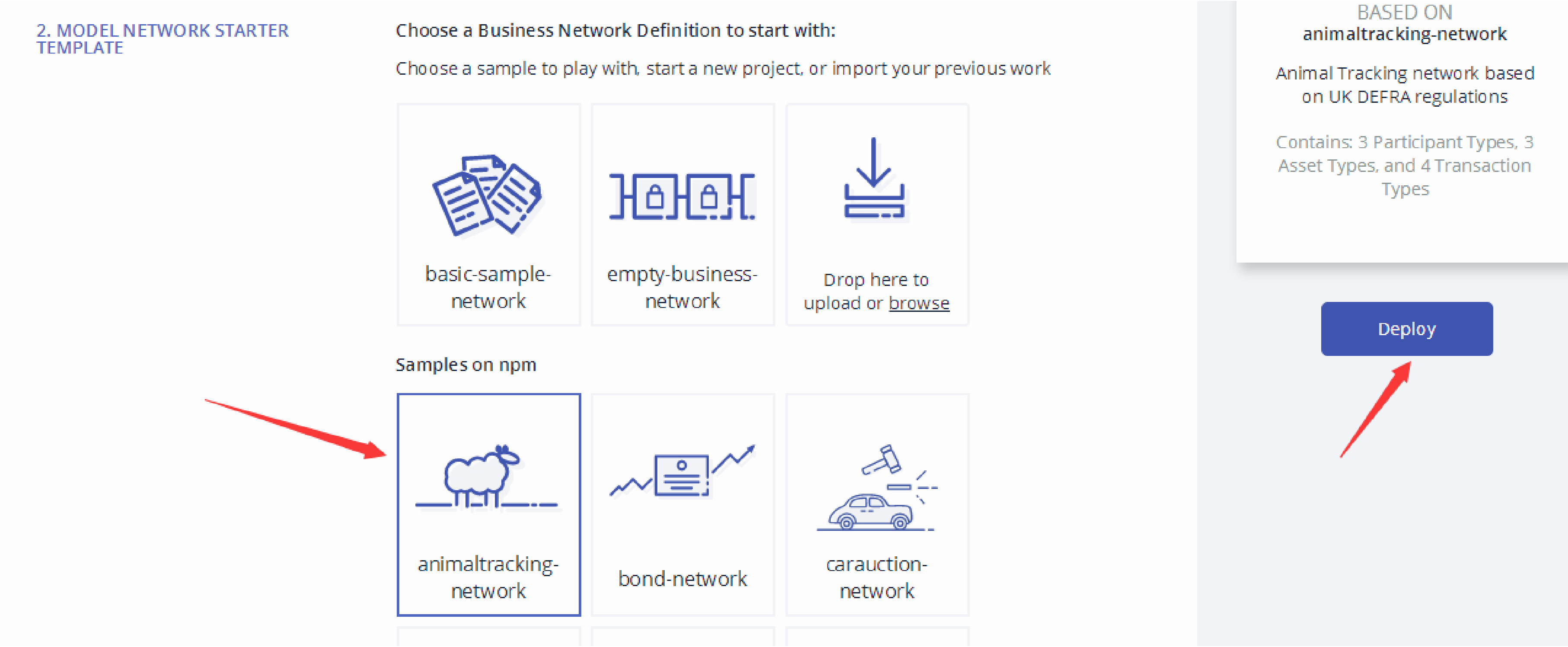
Describe what your Business Network will be used for: for sysu

Give the network admin card that will be created a name: sysu@course

联盟链智能合约

Hyperledger Composer

选择创建示例网络



2. MODEL NETWORK STARTER TEMPLATE

Choose a Business Network Definition to start with:

Choose a sample to play with, start a new project, or import your previous work

Samples on npm

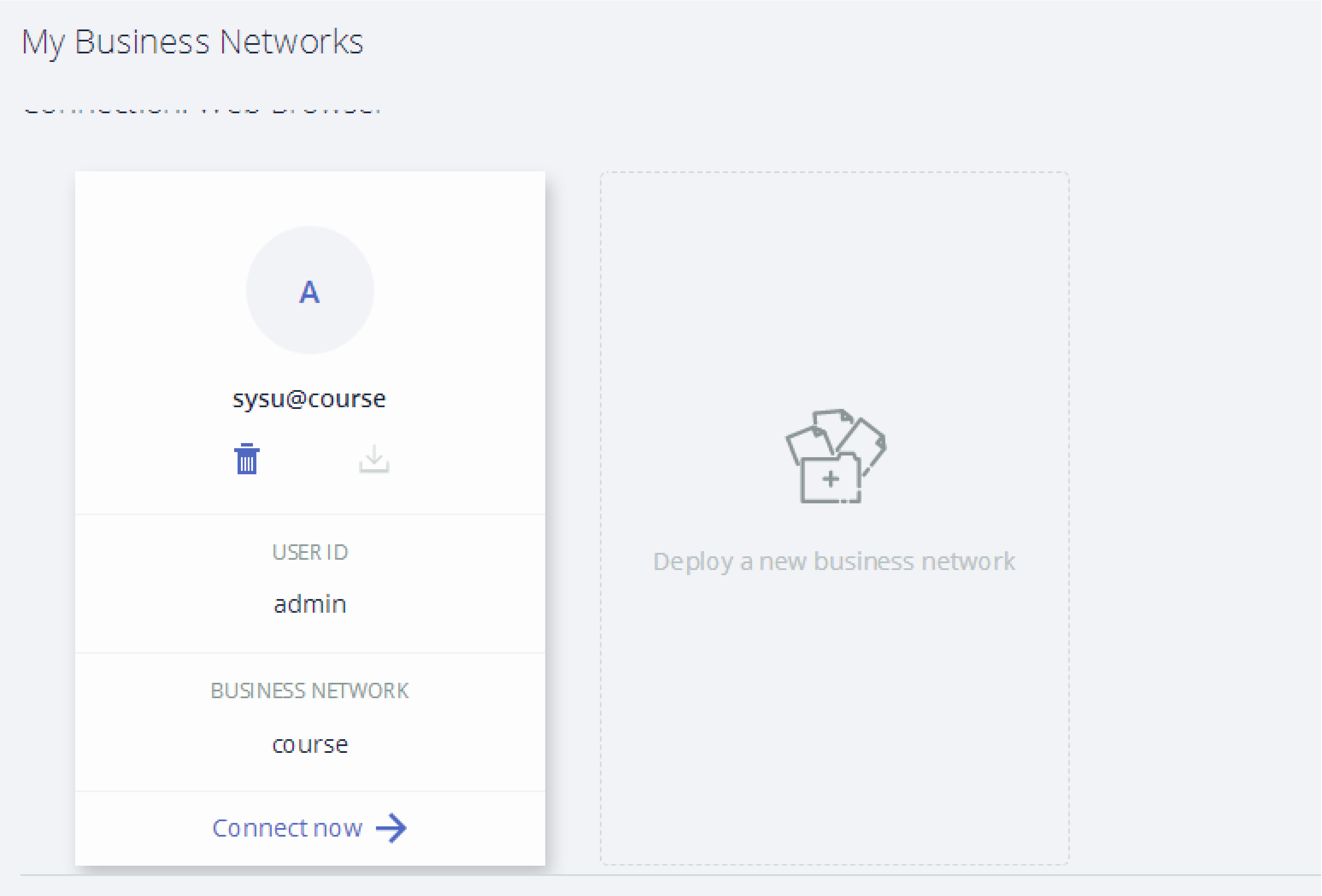
BASED ON
animaltracking-network

Animal Tracking network based on UK DEFRA regulations

Contains: 3 Participant Types, 3 Asset Types, and 4 Transaction Types

联盟链智能合约

Hyperledger Composer



My Business Networks

Connection: Web Browser

A

sysu@course

USER ID

admin

BUSINESS NETWORK

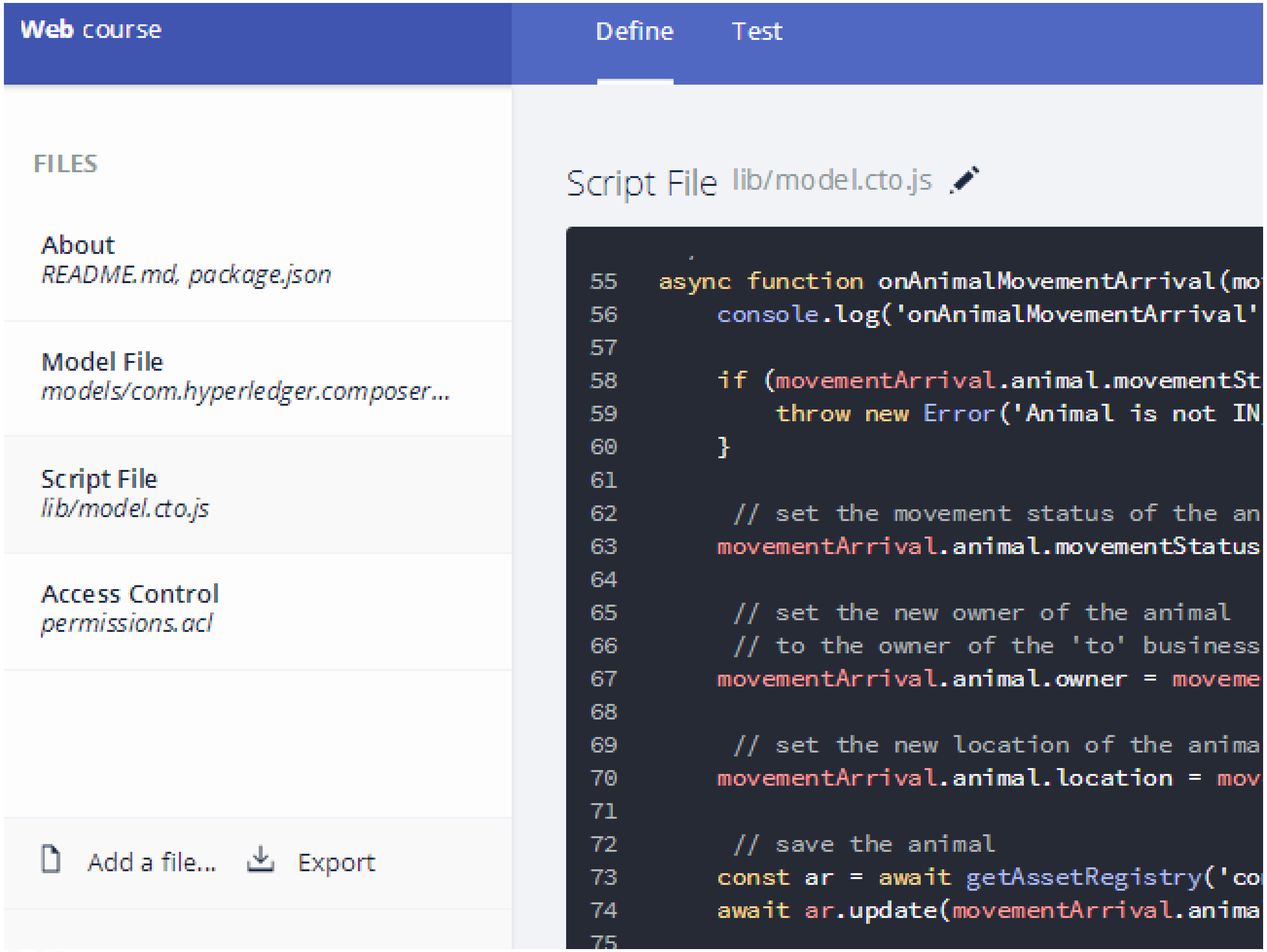
course

Connect now →

Deploy a new business network

联盟链智能合约

Hyperledger Composer



About
README.md, package.json

Model File
models/com.hyperledger.composer...

Script File
lib/model.cto.js

Access Control
permissions.acl

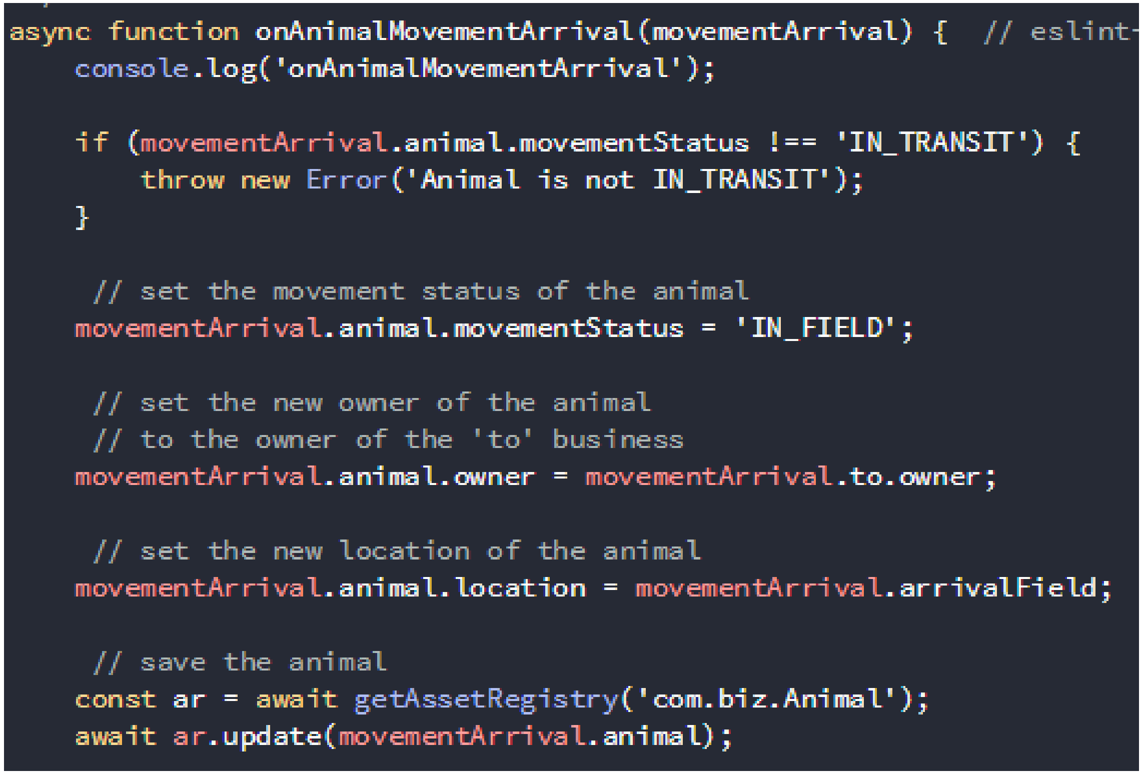
Script File lib/model.cto.js

```
async function onAnimalMovementArrival(mo
    console.log('onAnimalMovementArrival'

    if (movementArrival.animal.movementSt
        throw new Error('Animal is not IN
    }

     // set the movement status of the an
    movementArrival.animal.movementStatus

     // set the new owner of the animal
     // to the owner of the 'to' business
    movementArrival.animal.owner = moveme

     // set the new location of the anima
    movementArrival.animal.location = mov

     // save the animal
    const ar = await getAssetRegistry('co
    await ar.update(movementArrival.anima
```

联盟链智能合约

Hyperledger Composer

示例合约：物流溯源

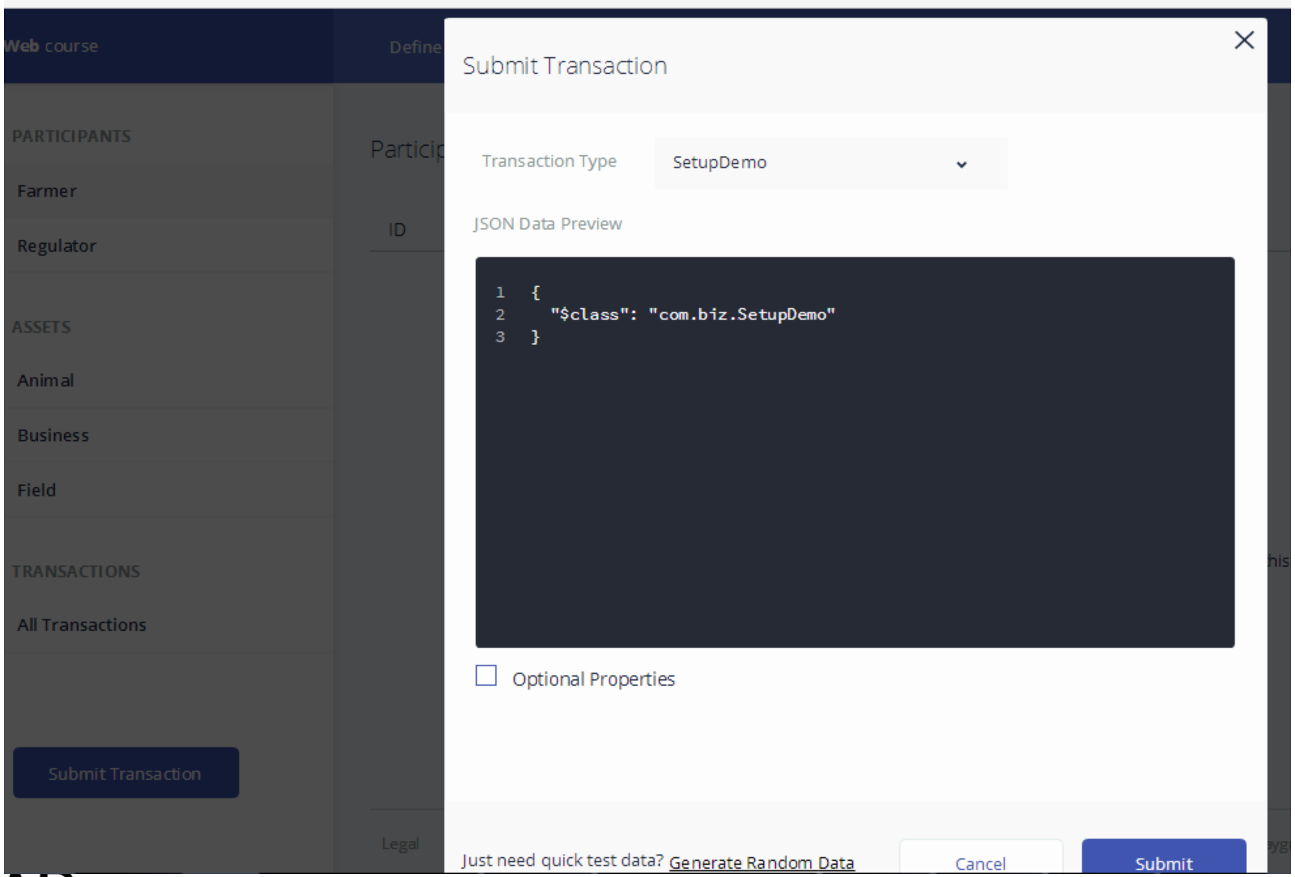


```
async function onAnimalMovementArrival(movementArrival) {  // eslint-
    console.log('onAnimalMovementArrival');

    if (movementArrival.animal.movementStatus !== 'IN_TRANSIT') {
        throw new Error('Animal is not IN_TRANSIT');
    }

     // set the movement status of the animal
    movementArrival.animal.movementStatus = 'IN_FIELD';

     // set the new owner of the animal
     // to the owner of the 'to' business
    movementArrival.animal.owner = movementArrival.to.owner;

     // set the new location of the animal
    movementArrival.animal.location = movementArrival.arrivalField;

     // save the animal
    const ar = await getAssetRegistry('com.biz.Animal');
    await ar.update(movementArrival.animal);
```

联盟链智能合约

Hyperledger Composer

运行初始化函数



Submit Transaction

Transaction Type: SetupDemo

JSON Data Preview

```
{
  "$class": "com.biz.SetupDemo"
}
```

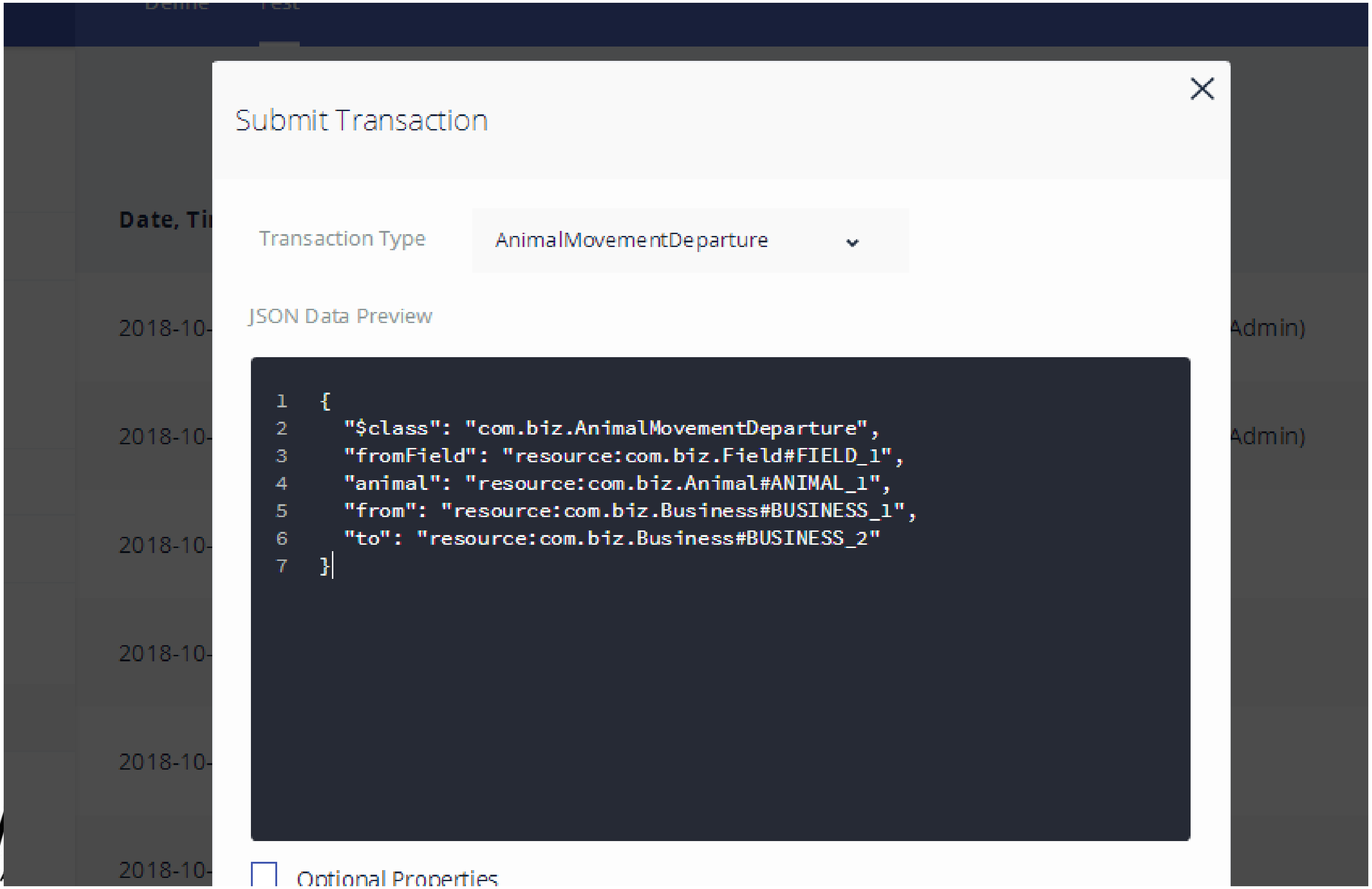
☐ Optional Properties

Just need quick test data? Generate Random Data

联盟链智能合约

Hyperledger Composer

动物离开农场



Submit Transaction

Transaction Type: AnimalMovementDeparture

JSON Data Preview

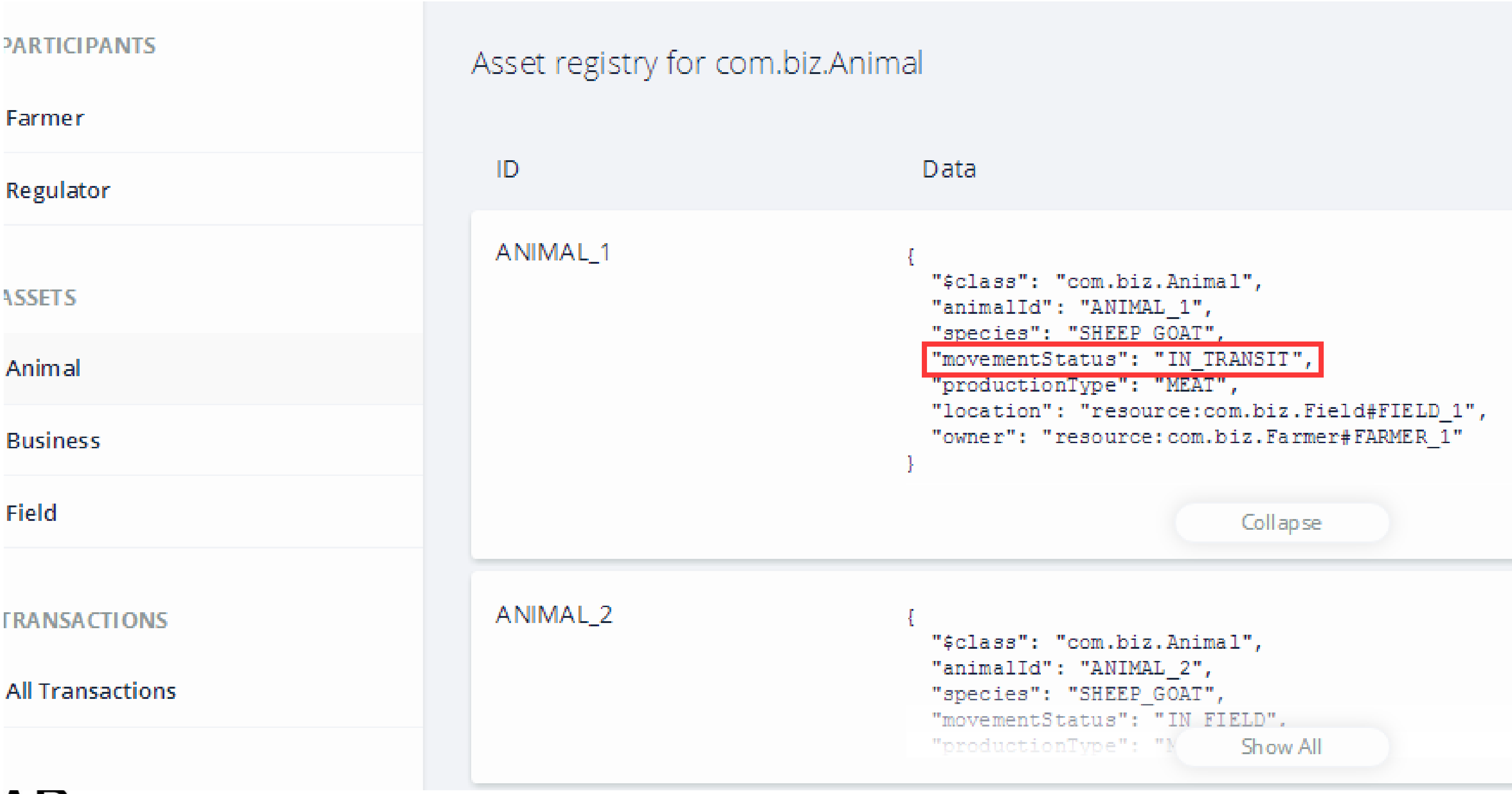
```
{
  "$class": "com.biz.AnimalMovementDeparture",
  "fromField": "resource:com.biz.Field#FIELD_1",
  "animal": "resource:com.biz.Animal#ANIMAL_1",
  "from": "resource:com.biz.Business#BUSINESS_1",
  "to": "resource:com.biz.Business#BUSINESS_2"
}
```

□ Optional Properties

联盟链智能合约

Hyperledger Composer

动物离开农场



PARTICIPANTS

Farmer

Regulator

ASSETS

Animal

Business

Field

TRANSACTIONS

All Transactions

Asset registry for com.biz.Animal

ID	Data
ANIMAL_1	see below

ANIMAL_1

```
{
  "$class": "com.biz.Animal",
  "animalId": "ANIMAL_1",
  "species": "SHEEP_GOAT",
  "movementStatus": "IN_TRANSIT",
  "productionType": "MEAT",
  "location": "resource:com.biz.Field#FIELD_1",
  "owner": "resource:com.biz.Farmer#FARMER_1"
}
```

Collapse

ANIMAL_2

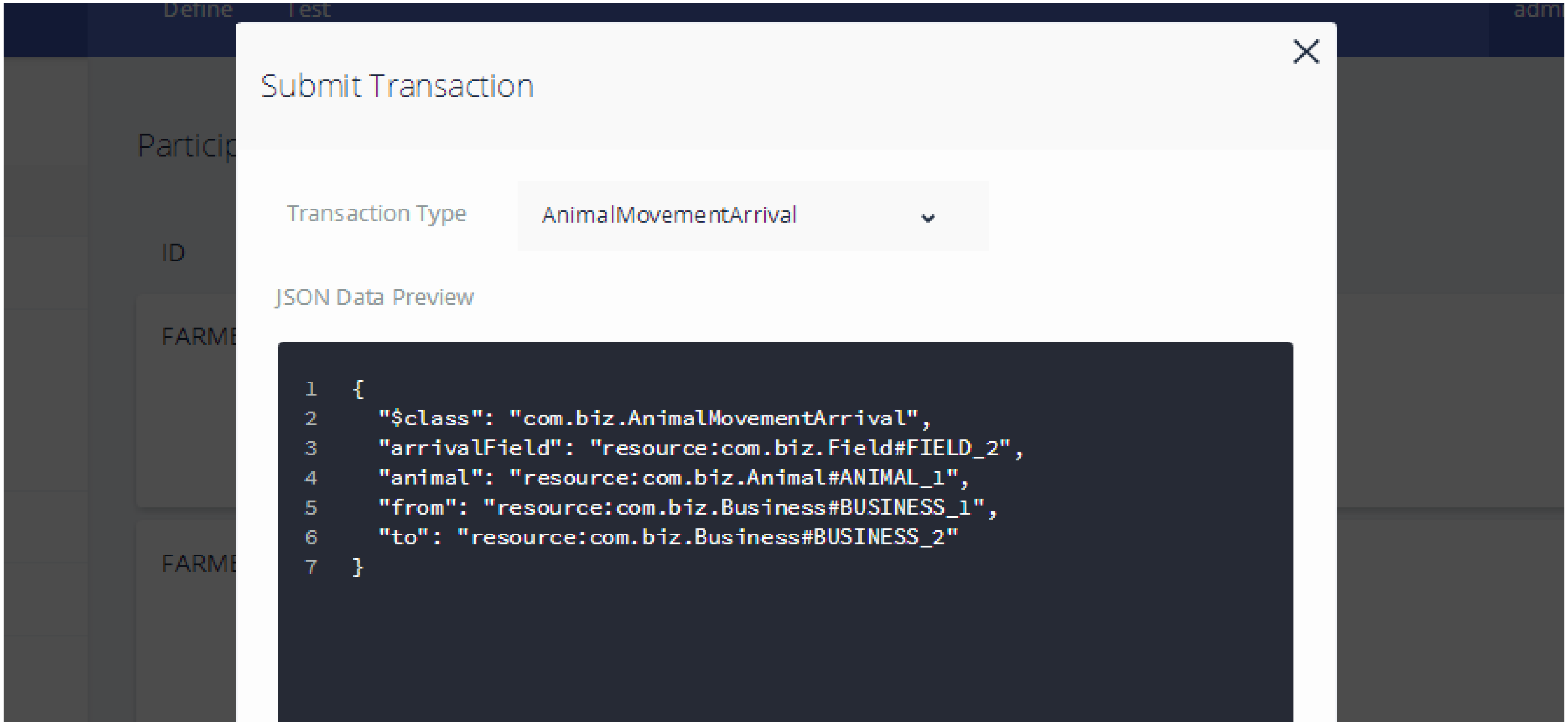
```
{
  "$class": "com.biz.Animal",
  "animalId": "ANIMAL_2",
  "species": "SHEEP_GOAT",
  "movementStatus": "IN_FIELD",
  "productionType": "M
```

Show All

联盟链智能合约

Hyperledger Composer

动物到达目的地



Submit Transaction

Transaction Type: AnimalMovementArrival

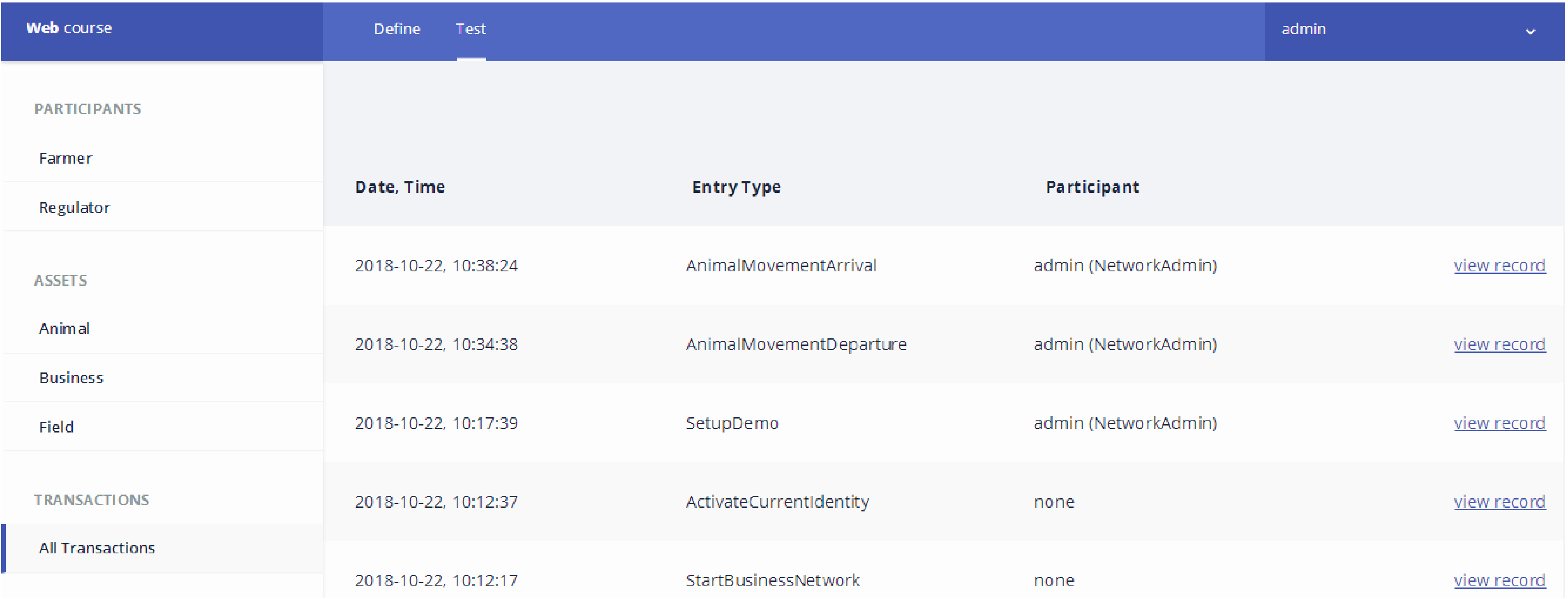
JSON Data Preview

```
{
  "$class": "com.biz.AnimalMovementArrival",
  "arrivalField": "resource:com.biz.Field#FIELD_2",
  "animal": "resource:com.biz.Animal#ANIMAL_1",
  "from": "resource:com.biz.Business#BUSINESS_1",
  "to": "resource:com.biz.Business#BUSINESS_2"
}
```

联盟链智能合约

Hyperledger Composer

查看交易情况

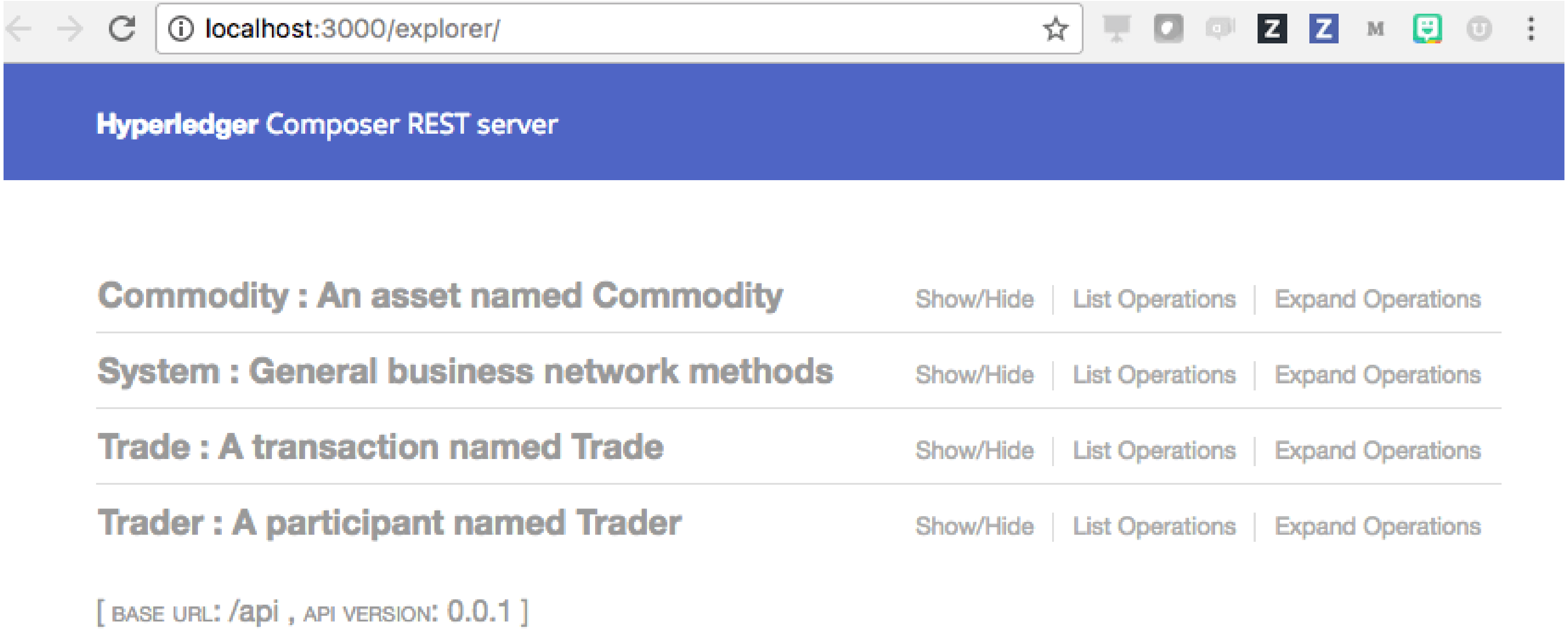


Date, Time	Entry Type	Participant	
2018-10-22, 10:38:24	AnimalMovementArrival	admin (NetworkAdmin)	view record
2018-10-22, 10:34:38	AnimalMovementDeparture	admin (NetworkAdmin)	view record
2018-10-22, 10:17:39	SetupDemo	admin (NetworkAdmin)	view record
2018-10-22, 10:12:37	ActivateCurrentIdentity	none	view record
2018-10-22, 10:12:17	StartBusinessNetwork	none	view record

联盟链智能合约

Hyperledger Composer

自动生成REST API



localhost:3000/explorer/

Hyperledger Composer REST server

Commodity : An asset named Commodity	Show/Hide	List Operations	Expand Operations
System : General business network methods	Show/Hide	List Operations	Expand Operations
Trade : A transaction named Trade	Show/Hide	List Operations	Expand Operations
Trader : A participant named Trader	Show/Hide	List Operations	Expand Operations

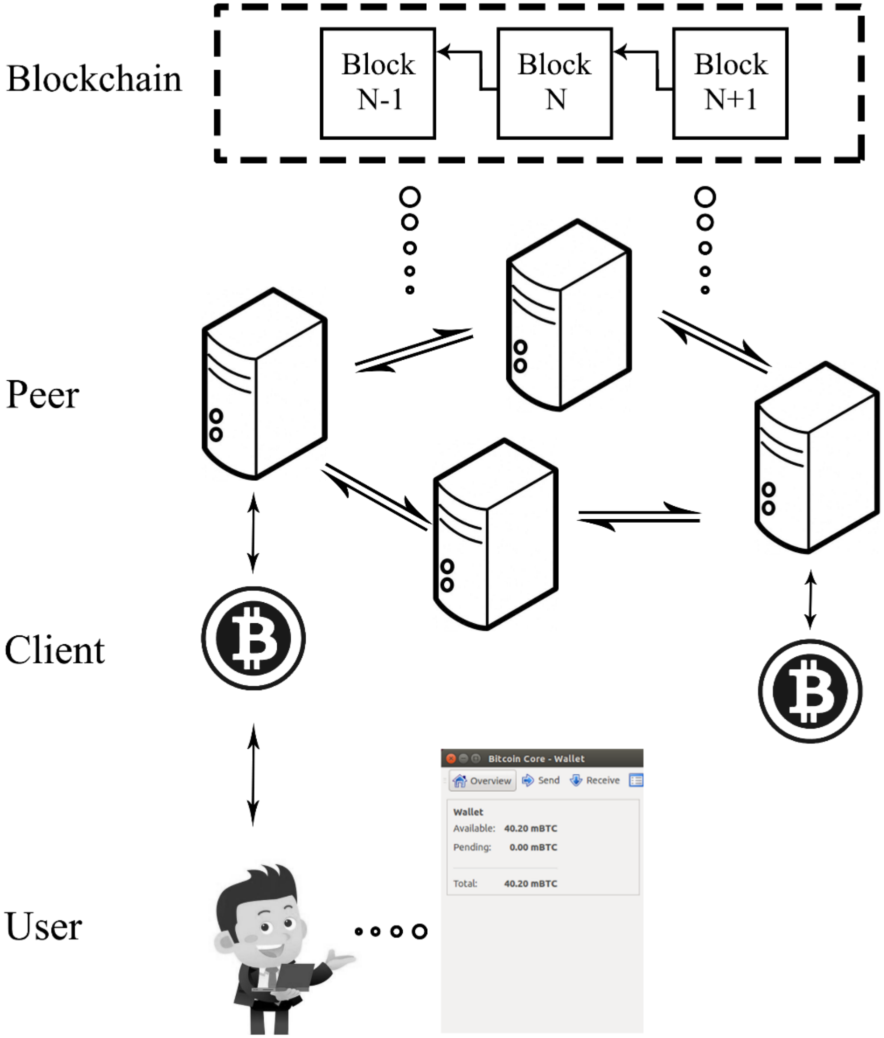
[BASE URL: /api , API VERSION: 0.0.1]

目录

1. 智能合约及平台简介
2. 以太坊基本操作及原理
3. Solidity语言
4. 联盟链智能合约
5. 为智能合约构造图形交互

为智能合约构造图形交互

- 原生写一个的区块链客户端



Bitcoin Core - Wallet

Wallet

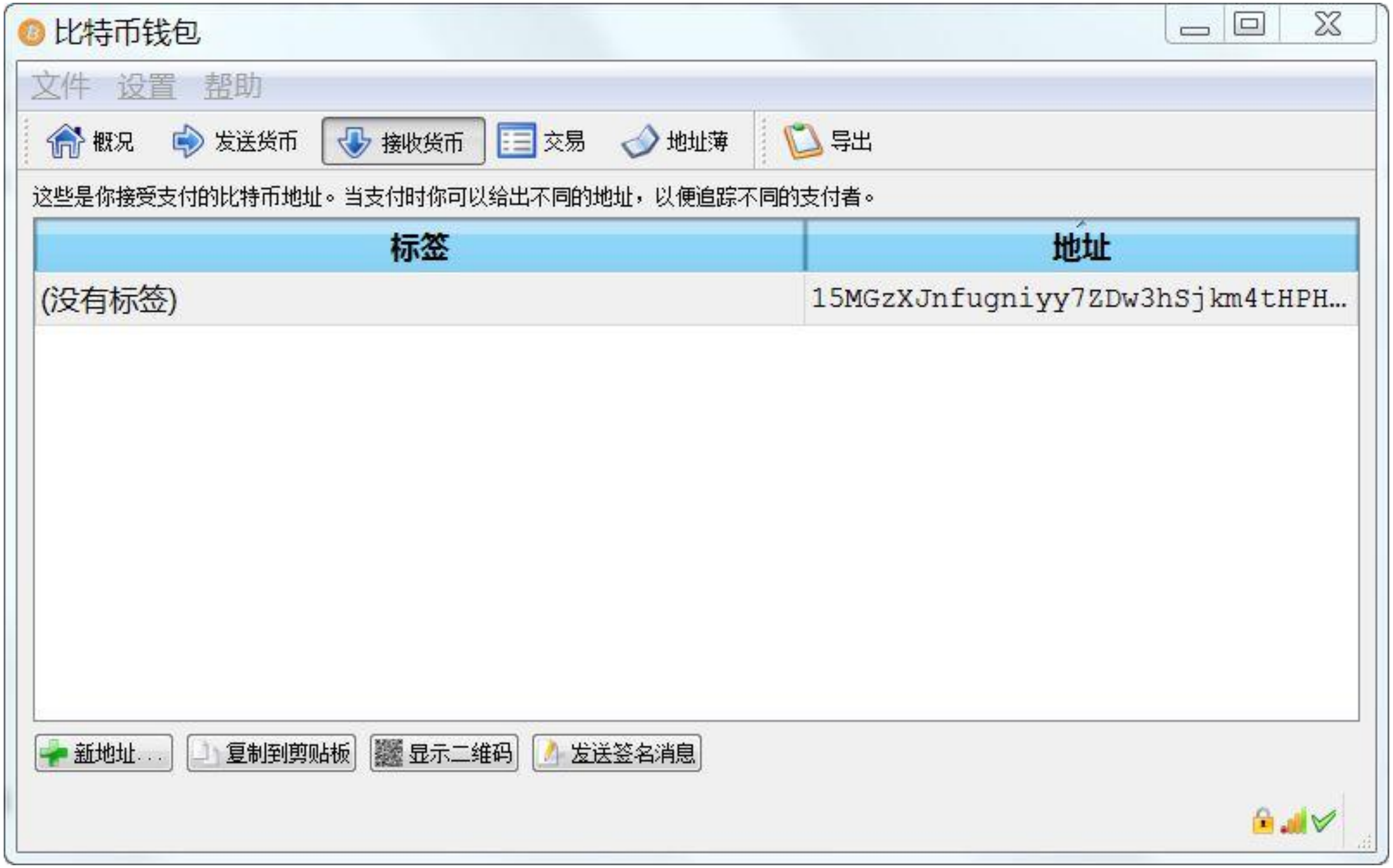
Available: 40.20 mBTC

Pending: 0.00 mBTC

Total: 40.20 mBTC

为智能合约构造图形交互

■ 原生写一个的区块链客户端

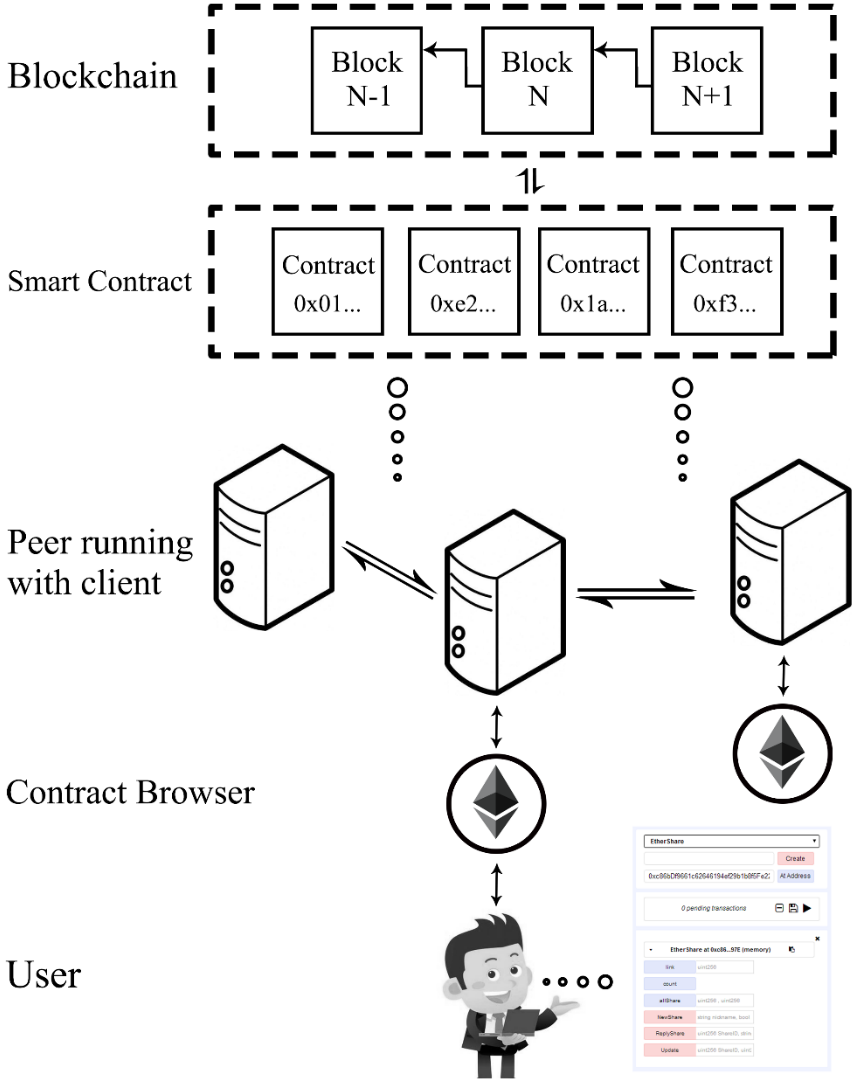


这些是你接受支付的比特币地址。当支付时你可以给出不同的地址，以便追踪不同的支付者。

标签	地址
(没有标签)	15MGzXJnfugniyy7ZDw3hSjkm4tHPH...

为智能合约构造图形交互

- 通过通用的智能合约浏览器进行交互



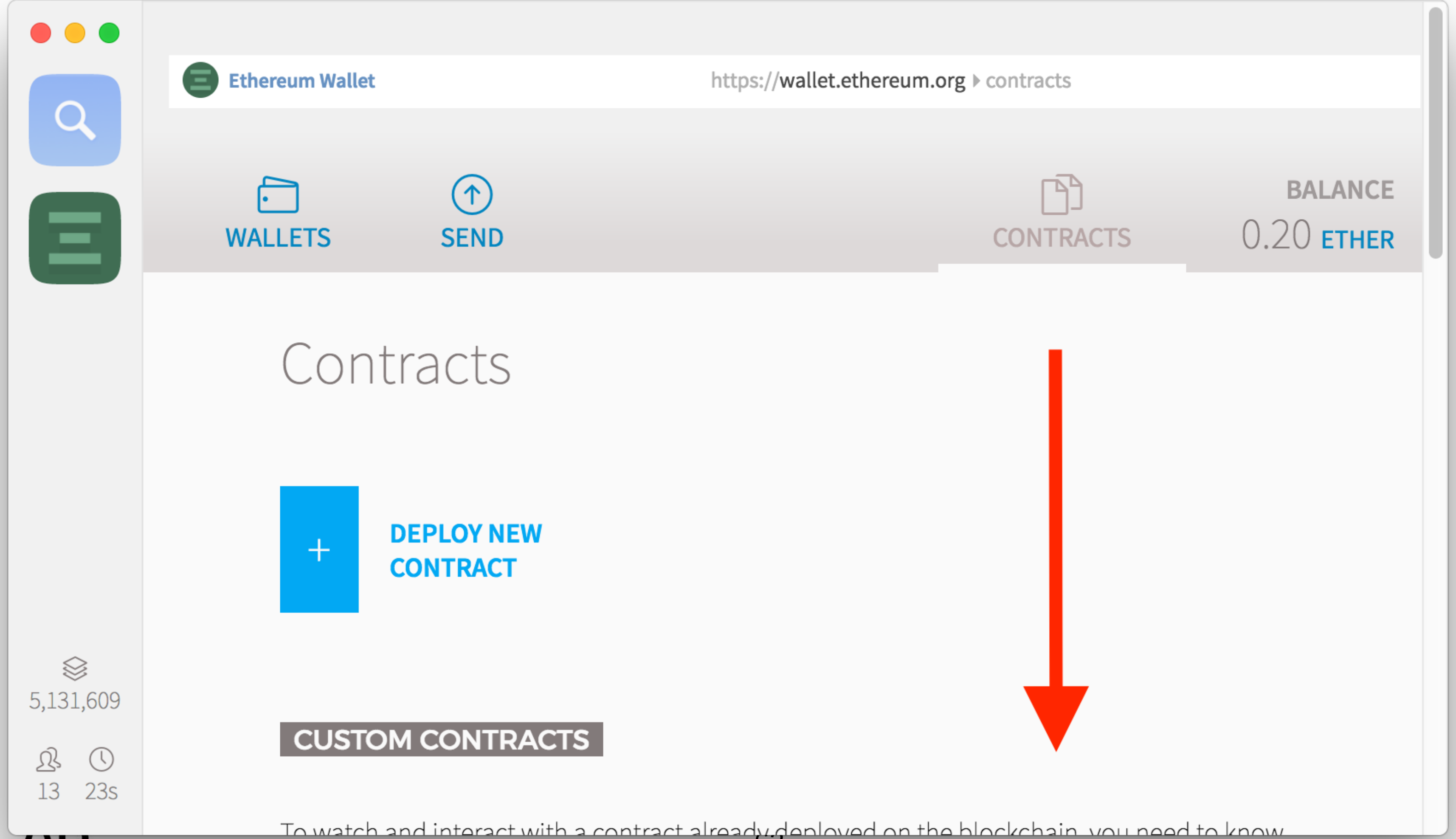
0xc86bDf9661c62646194ef29b1b8f5Fe2

EtherShare at 0xc86...97E (memory)

link	uint256
count	
allShare	uint256 , uint256
NewShare	string nickname, bool
ReplyShare	uint256 ShareID, strin
Update	uint256 ShareID, uint

为智能合约构造图形交互

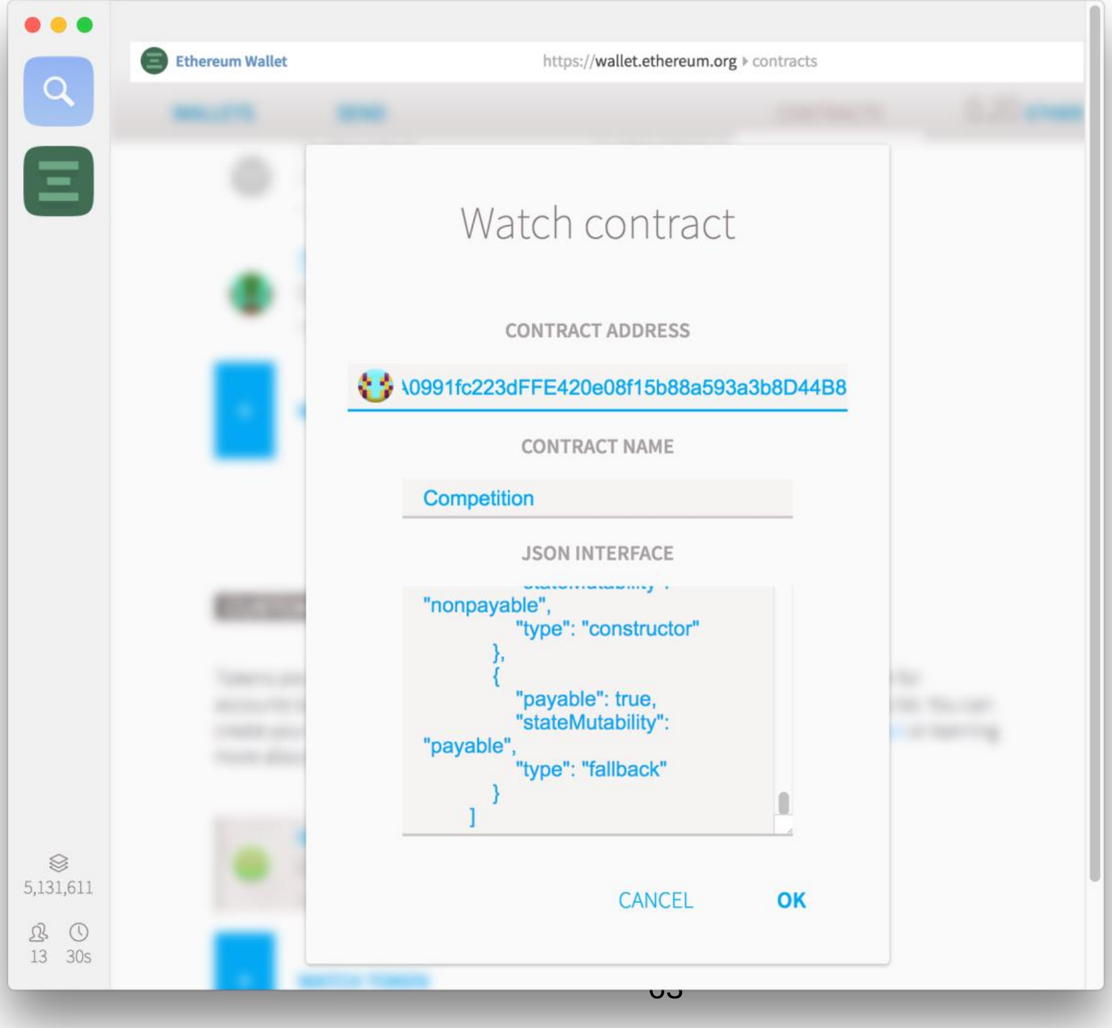
- 通过通用的智能合约浏览器进行交互



To watch and interact with a contract already deployed on the blockchain, you need to know

为智能合约构造图形交互

- 通过通用的智能合约浏览器进行交互



Watch contract

CONTRACT ADDRESS

\0991fc223dFFE420e08f15b88a593a3b8D44B8

CONTRACT NAME

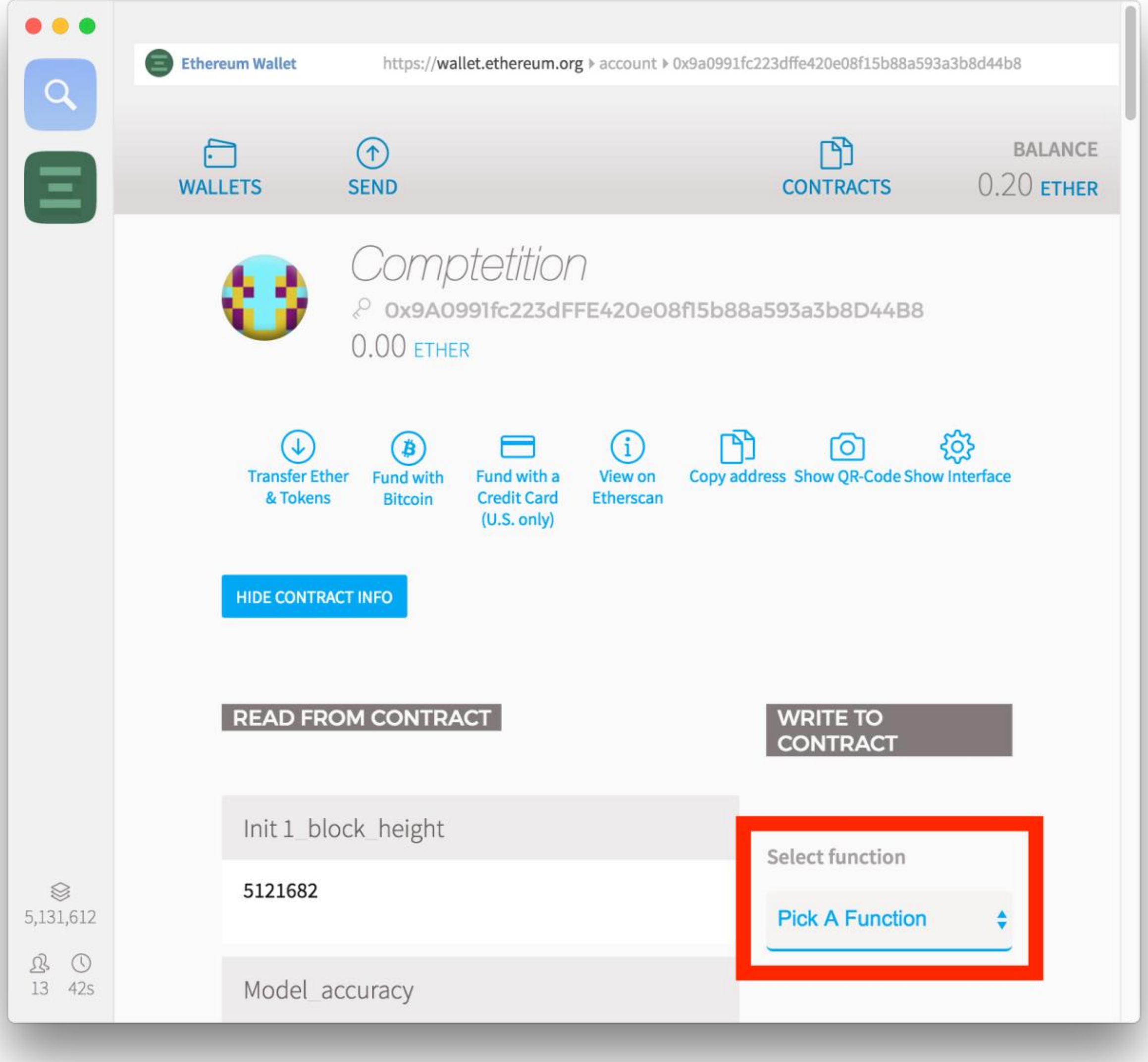
Competition

JSON INTERFACE

```
"nonpayable",
            "type": "constructor"
        },
        {
            "payable": true,
            "stateMutability":
"payable",
            "type": "fallback"
        }
    ]
```

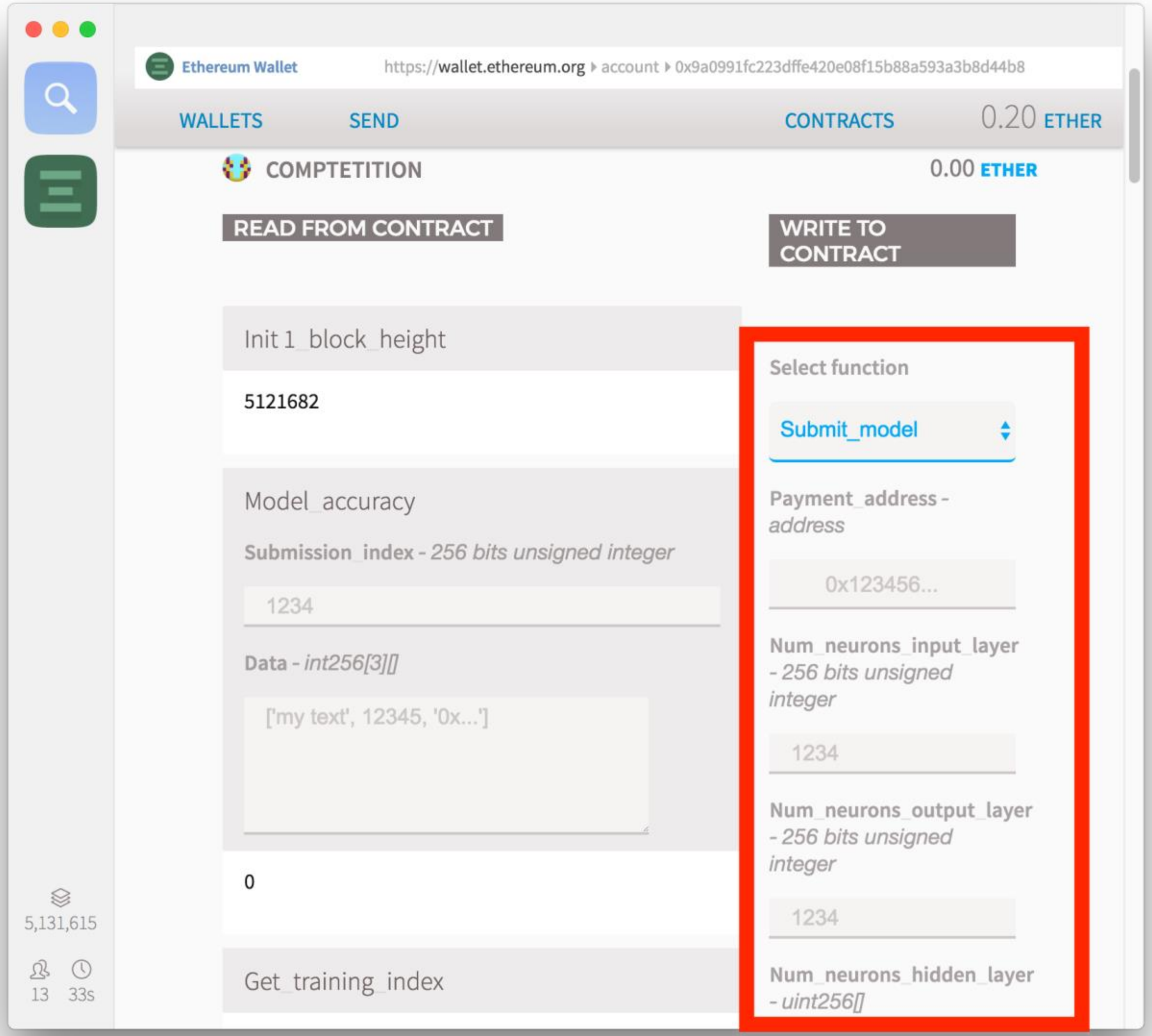
为智能合约构造图形交互

- 通过通用的智能合约浏览器进行交互



为智能合约构造图形交互

- 通过通用的智能合约浏览器进行交互



https://wallet.ethereum.org ▸ account ▸ 0x9a0991fc223dffe420e08f15b88a593a3b8d44b8

COMPTETITION

0.00 ETHER

READ FROM CONTRACT

WRITE TO CONTRACT

Init 1_block_height

5121682

Model_accuracy

Submission_index - *256 bits unsigned integer*

1234

Data - *int256[3][]*

['my text', 12345, '0x...']

0

Get_training_index

Select function

Submit_model

Payment_address - *address*

0x123456...

Num_neurons_input_layer - *256 bits unsigned integer*

1234

Num_neurons_output_layer - *256 bits unsigned integer*

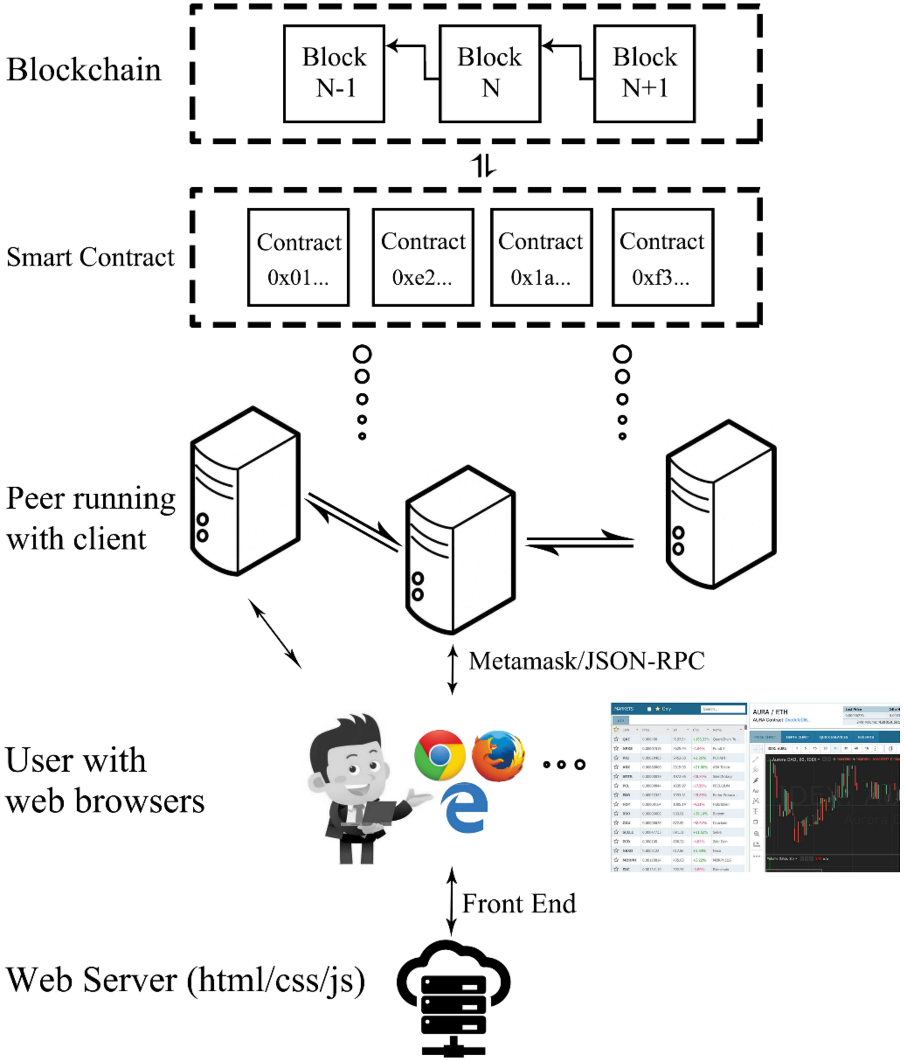
1234

Num_neurons_hidden_layer - *uint256[]*

LAB WWW.INPLUSLAB.COM

为智能合约构造图形交互

- 通过区块链平台的接口，结合GUI，进行交互



Blockchain

Block N-1

Block N

Block N+1

Smart Contract

Contract 0x01...

Contract 0xe2...

Contract 0x1a...

Contract 0xf3...

Peer running with client

Metamask/JSON-RPC

User with web browsers

Front End

Web Server (html/css/js)

为智能合约构造图形交互

■ 通过区块链平台的接口，结合GUI，进行交互

以网页调用web3.js为例：

- ➢ 设置节点地址

  web3 = new Web3(new Web3.providers.HttpProvider("https://mainnet.infura.io/"));

- ➢ 载入合约

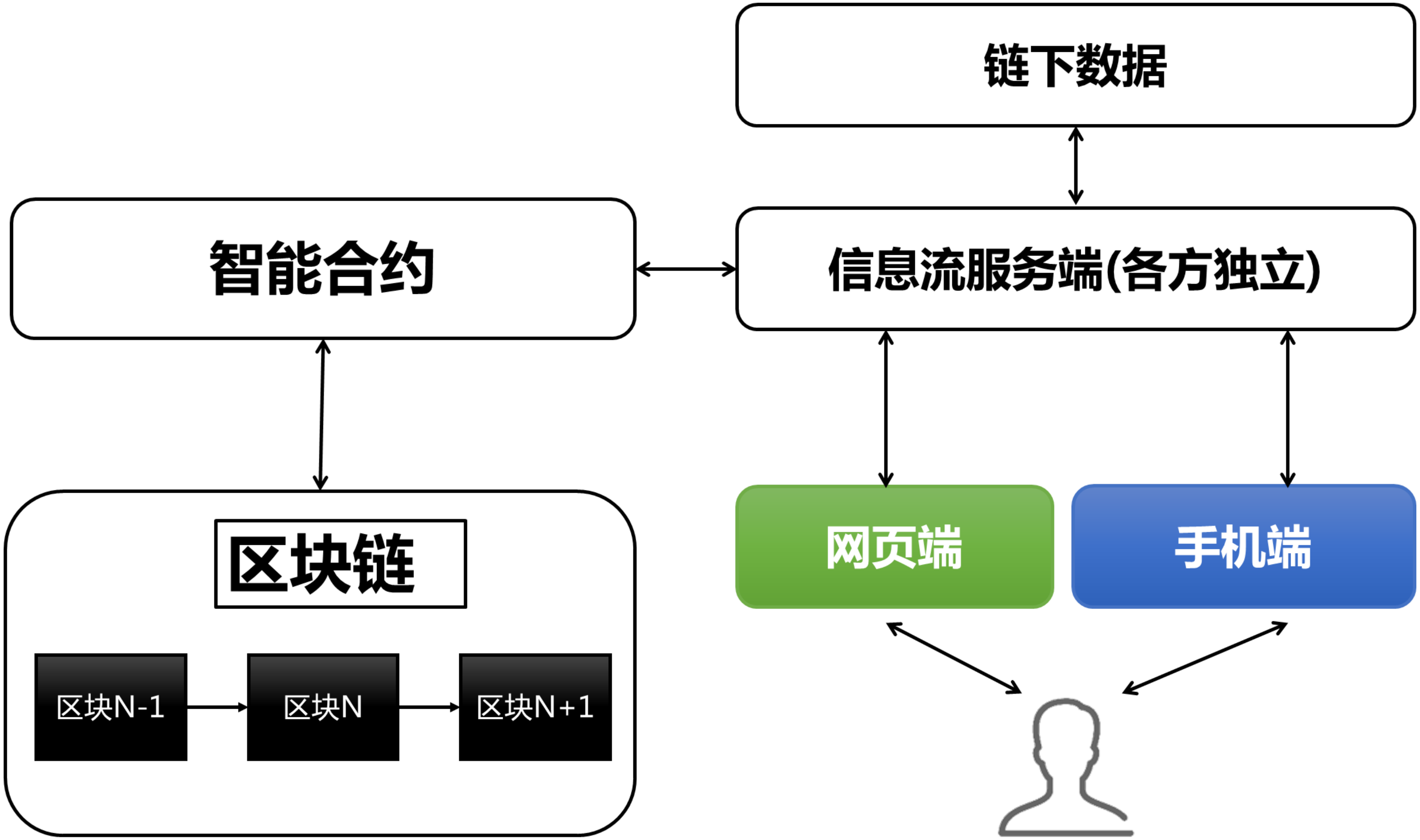
  合约名字 = web3.eth.contract(合约ABI).at(带双引号的合约地址);

- ➢ 调用合约

  合约名字.函数.call()或sendTransaction()

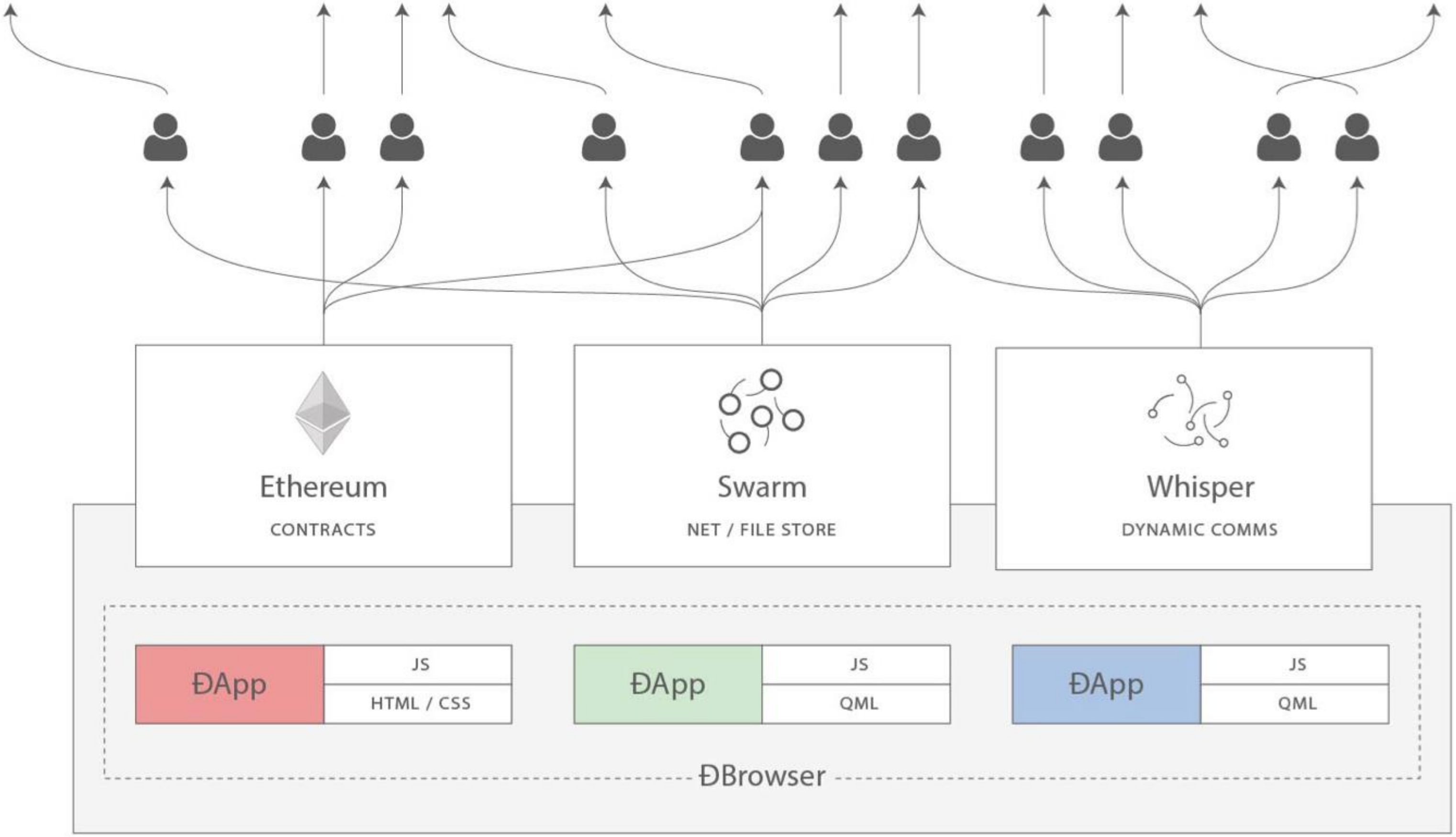
为智能合约构造图形交互

- 集成区块链接口开发服务端（联盟链场景较多）



为智能合约构造图形交互

- 如果连图形交互也去中心化?



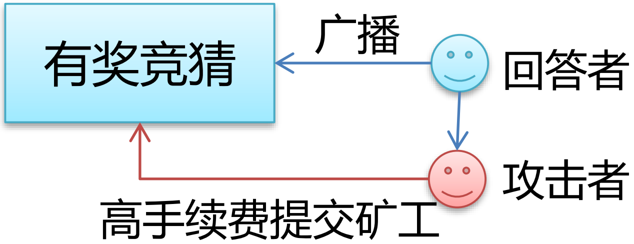
谢谢！

附录

智能合约漏洞

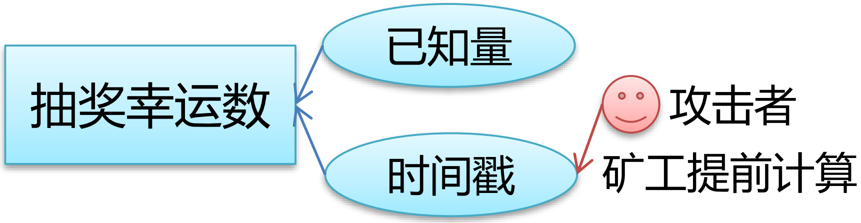
- **交易顺序依赖**

智能合约的执行随着当前交易处理的顺序不同而产生差异。



- **时间戳依赖**

智能合约的执行依赖区块时间戳，时间戳不同，执行结果也有差别

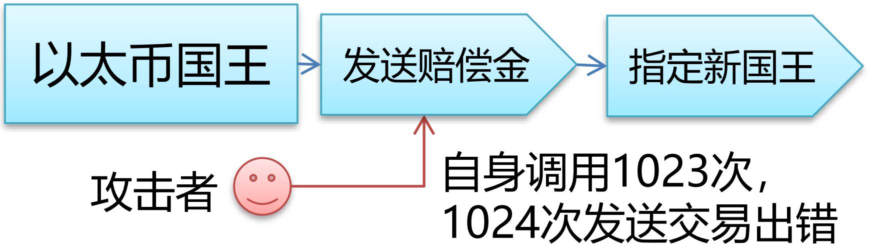


附录

智能合约漏洞

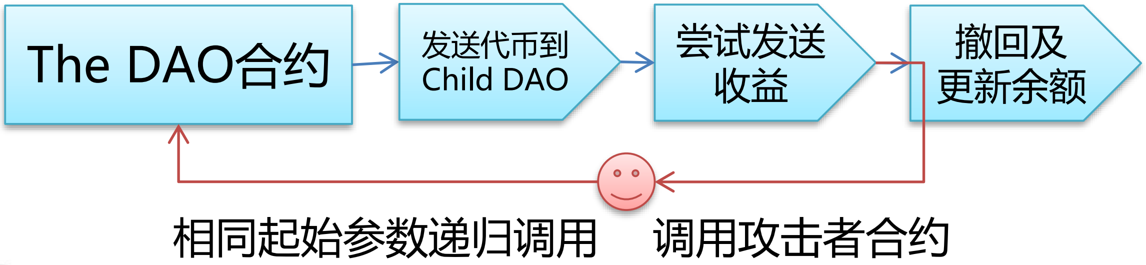
误操作异常

合约可调用另一个合约的函数，而异常可能无法很好地被调用者得知。



可重入攻击

当一个合约调用另一个合约的时候，当前操作要等到调用结束之后继续。


The DAO合约
发送代币到
Child DAO
尝试发送
收益
撤回及
更新余额
相同起始参数递归调用
调用攻击者合约