Smart Contracts 101

Crafting the InfuyToken

AGENDA

1

Smart Contracts and Solidity

Quick introduction into Smart Contracts and Solidity Language specification

2

Development Environment

Setup using Ganache & Truffle framework.
Web3.js

3

Crafting the InfuyToken

Creation of a non standard Token

4

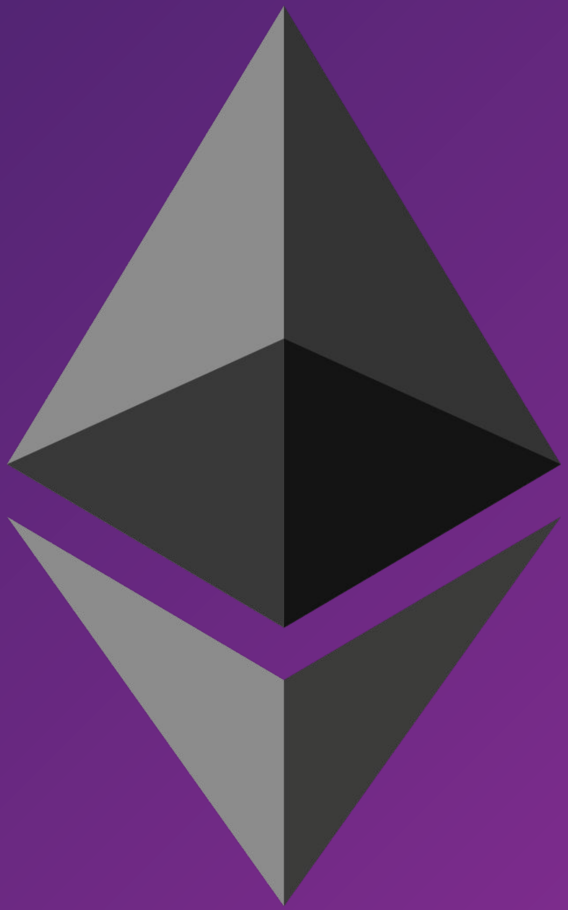
InfuyToken on TestNet

Contract deploy and Wallet interaction

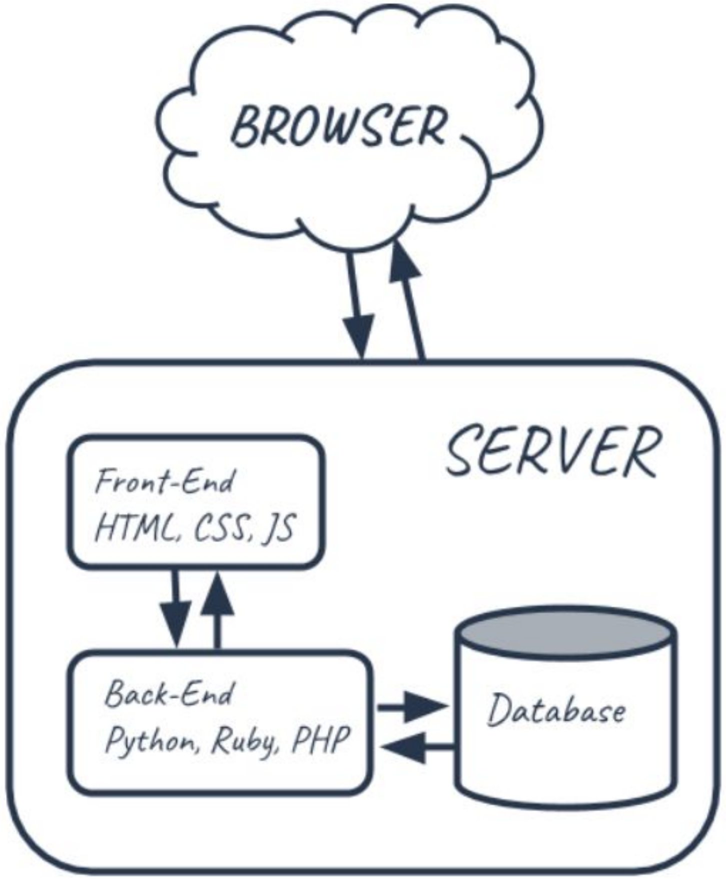
01

Smart contracts & Solidity

Ethereum



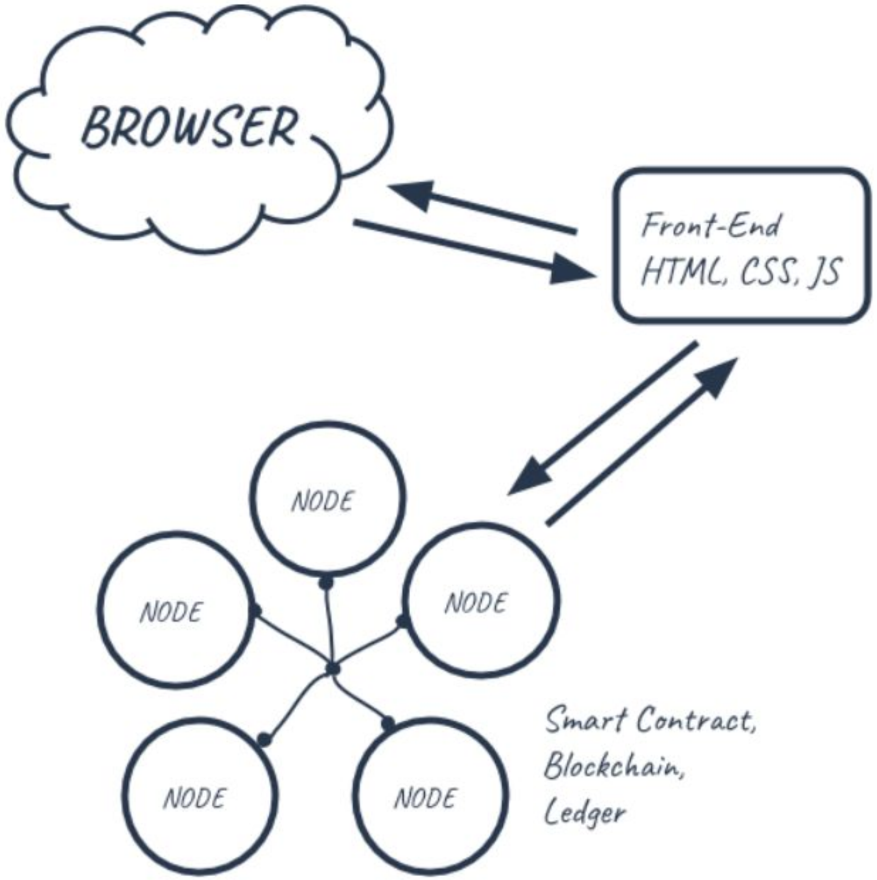
Why Ethereum?



Voting webapp issues

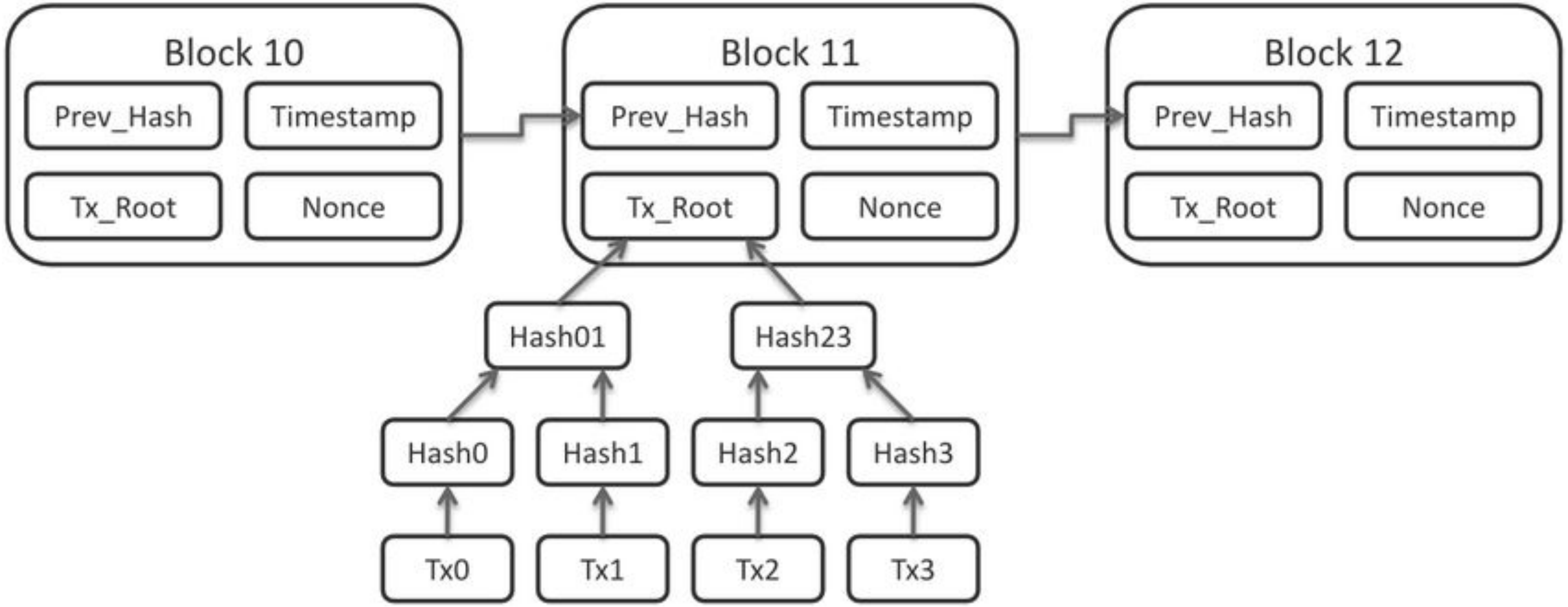
1. Data can be changed or lost
2. Votes can be counted twice
3. Source code can be modified at any time
4. Availability and Downtimes
5. Non-repudiation and transparency

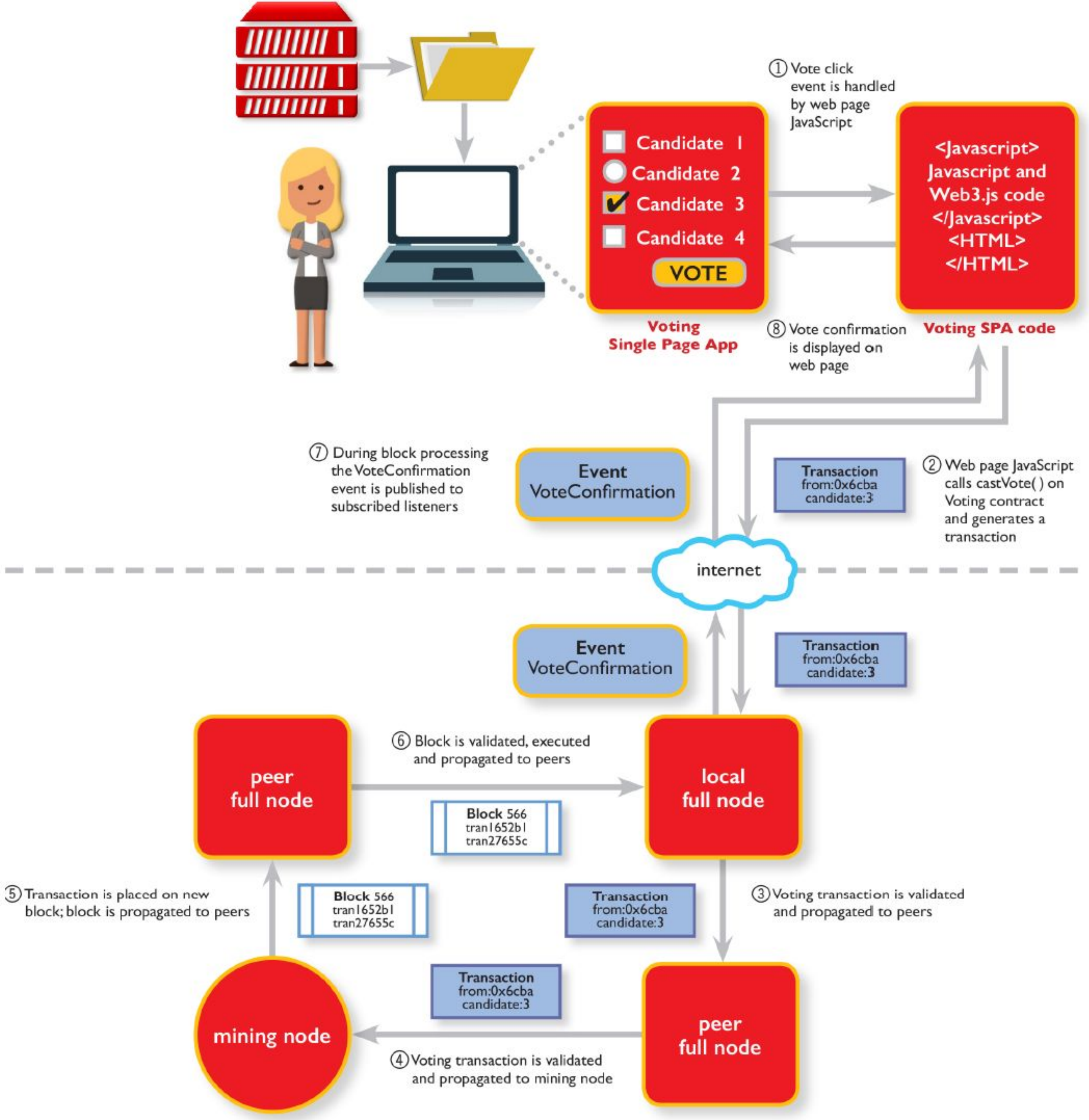
Why Ethereum?



How Ethereum works?

1. Nodes share the data across all the network
2. Transactions (data) are signed and stored in Blocks
3. All the nodes shares the same consensus algorithm
4. Blocks are validated and chained together




□ Candidate 1
○ Candidate 2
☑ Candidate 3
□ Candidate 4
VOTE
Voting Single Page App
<Javascript> Javascript and Web3.js code </Javascript> <HTML> </HTML>
Voting SPA code
Event VoteConfirmation
Transaction from:0x6cba candidate:3
internet
Event VoteConfirmation
Transaction from:0x6cba candidate:3
peer full node
Block 566 tran1652b1 tran27655c
local full node
Block 566 tran1652b1 tran27655c
Transaction from:0x6cba candidate:3
Transaction from:0x6cba candidate:3
mining node
peer full node

① Vote click event is handled by web page JavaScript

② Web page JavaScript calls castVote( ) on Voting contract and generates a transaction

③ Voting transaction is validated and propagated to peers

④ Voting transaction is validated and propagated to mining node

⑤ Transaction is placed on new block; block is propagated to peers

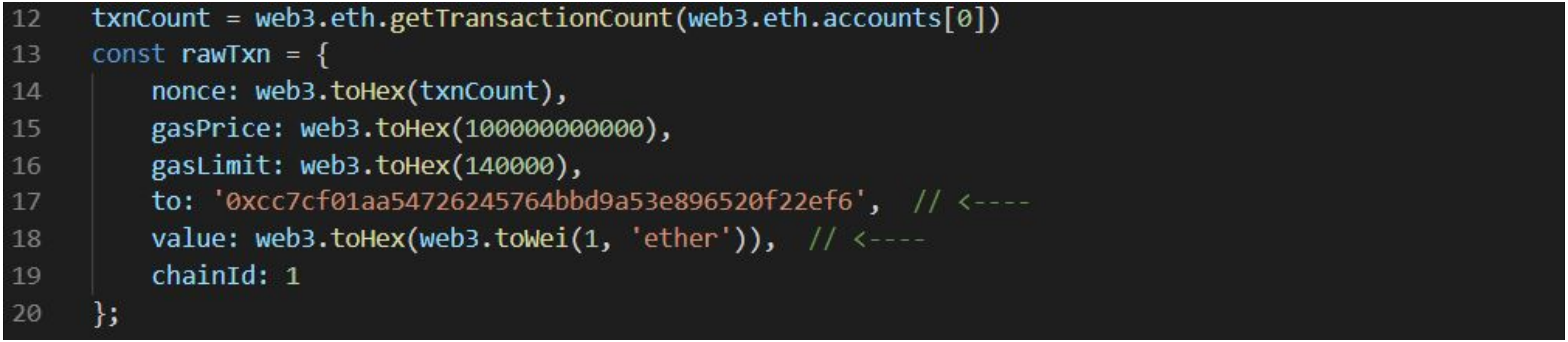
⑥ Block is validated, executed and propagated to peers

⑦ During block processing the VoteConfirmation event is published to subscribed listeners

⑧ Vote confirmation is displayed on web page

Ethereum Transactions

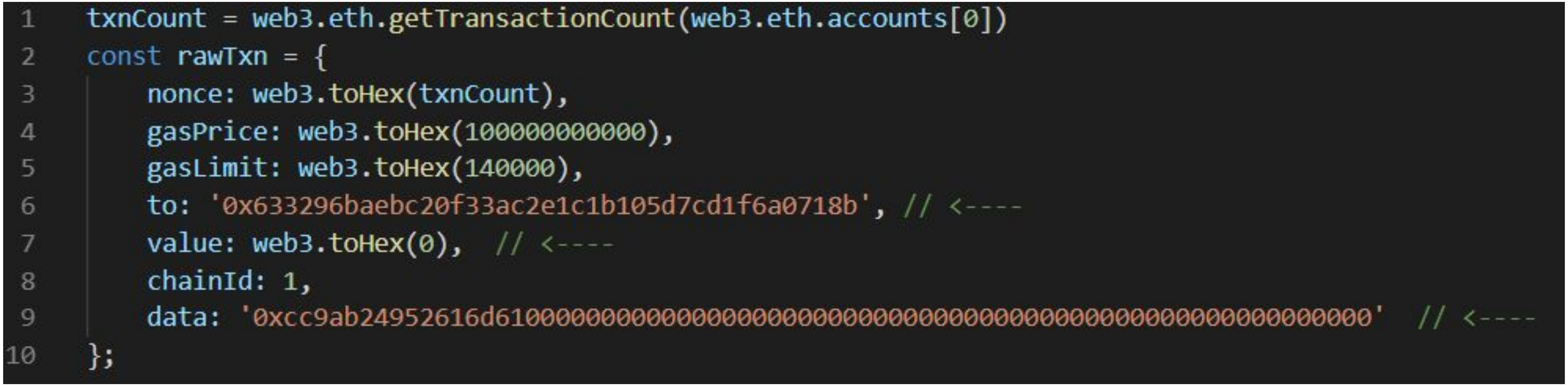
Ether sending



```
txnCount = web3.eth.getTransactionCount(web3.eth.accounts[0])
const rawTxn = {
    nonce: web3.toHex(txnCount),
    gasPrice: web3.toHex(100000000000),
    gasLimit: web3.toHex(140000),
    to: '0xcc7cf01aa54726245764bbd9a53e896520f22ef6',  // <----
    value: web3.toHex(web3.toWei(1, 'ether')),  // <----
    chainId: 1
};
```

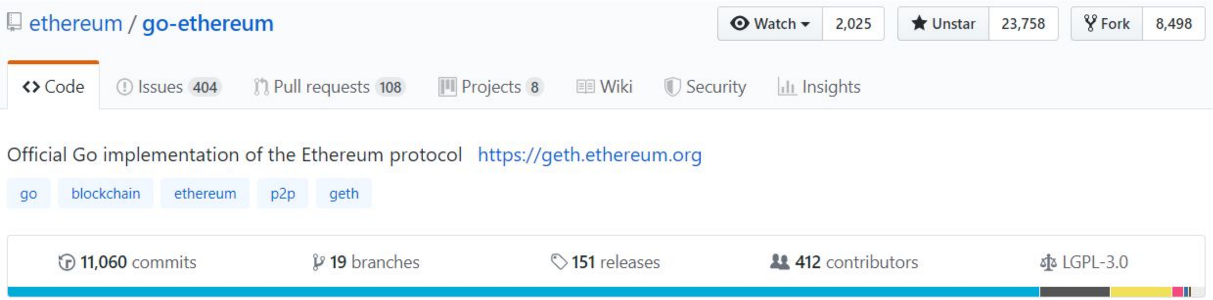
Ethereum Transactions

Contract execution



```
txnCount = web3.eth.getTransactionCount(web3.eth.accounts[0])
const rawTxn = {
    nonce: web3.toHex(txnCount),
    gasPrice: web3.toHex(100000000000),
    gasLimit: web3.toHex(140000),
    to: '0x633296baebc20f33ac2e1c1b105d7cd1f6a0718b', // <----
    value: web3.toHex(0),  // <----
    chainId: 1,
    data: '0xcc9ab24952616d6100000000000000000000000000000000000000000000000000000000'  // <----
};
```

Ethereum implementations

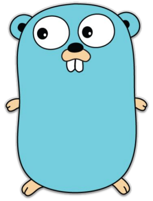


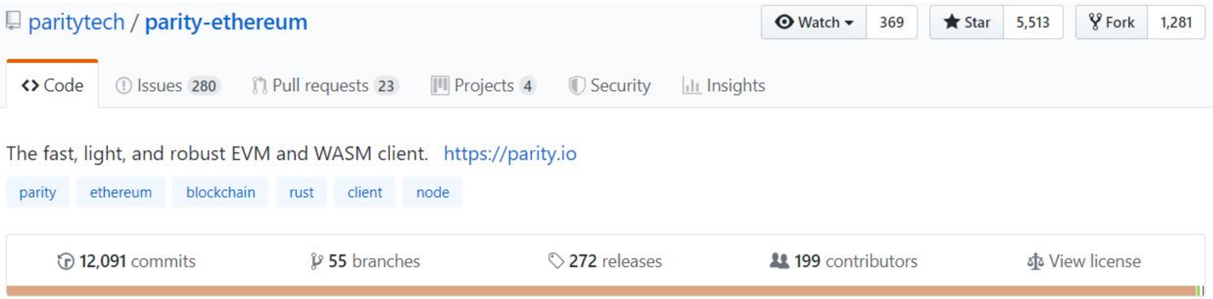
ethereum / go-ethereum

Official Go implementation of the Ethereum protocol https://geth.ethereum.org

go blockchain ethereum p2p geth

11,060 commits | 19 branches | 151 releases | 412 contributors | LGPL-3.0



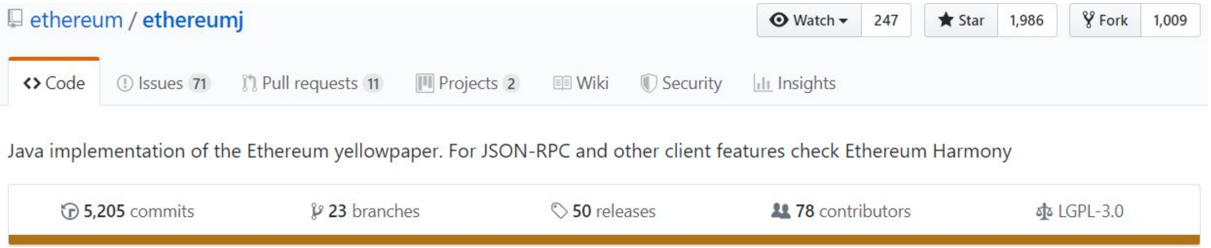


paritytech / parity-ethereum

The fast, light, and robust EVM and WASM client. https://parity.io

parity ethereum blockchain rust client node

12,091 commits | 55 branches | 272 releases | 199 contributors | View license



ethereum / ethereumj

Java implementation of the Ethereum yellowpaper. For JSON-RPC and other client features check Ethereum Harmony

5,205 commits | 23 branches | 50 releases | 78 contributors | LGPL-3.0

Smart Contracts

Smart contracts

- First used by Nick Szabo (CS, law scholar and cryptographer)
- 1997, before Bitcoin creation
- Digitalize real life contracts and publish them into a public ledger.

What is a Smart Contract?

Immutability

Piece of code stored at the blockchain

Removes third parties

Defines conditions which using parties agrees

Autonomous

If required conditions are met certain actions are executed

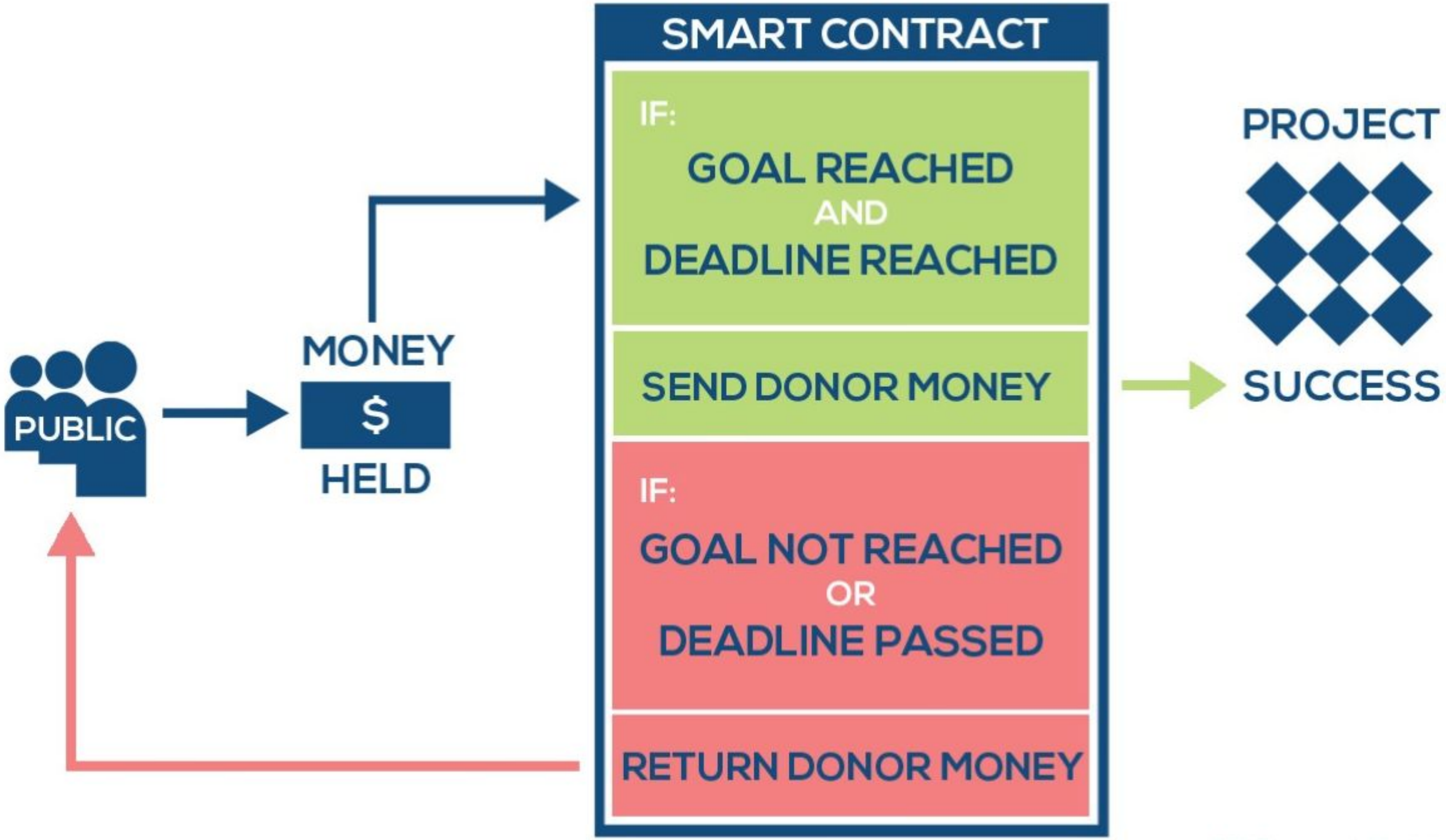
Trusted decentralization

Validated for each blockchain node

Example: crowdfunding

-

Example: crowdfunding

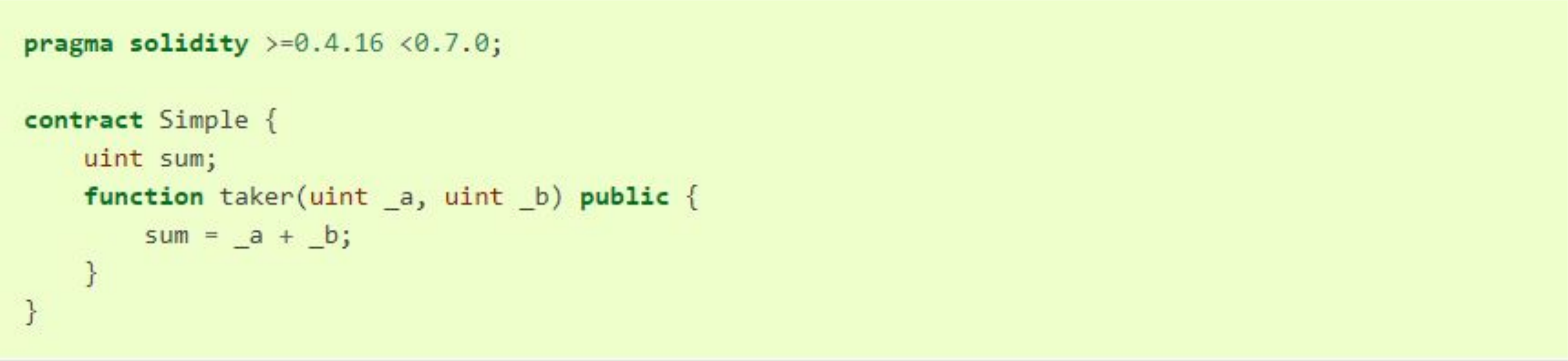


Solidity

What is Solidity?

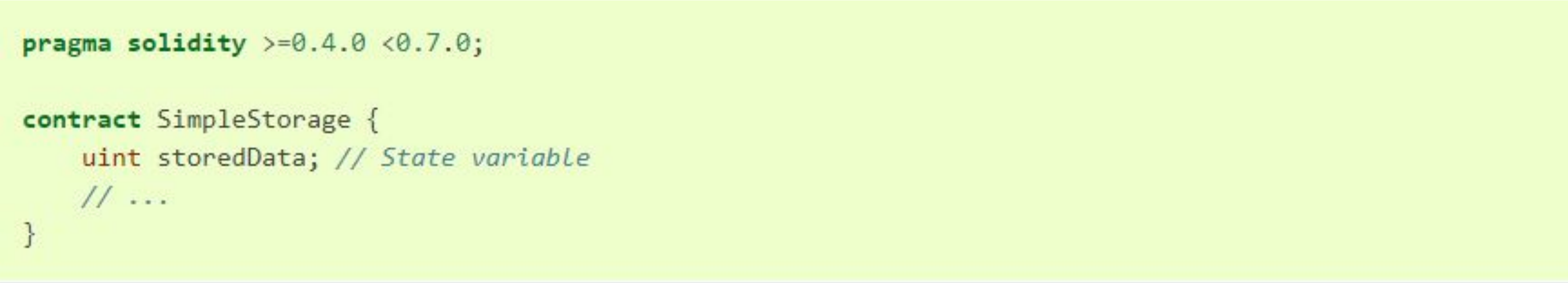
- Object oriented language for smart contracts development
- Influenced by C++, Python and JavaScript
- Designed to target the Ethereum Virtual Machine (EVM)
- Static typing (during compilation phase)
- Supports inheritance and composition
- Supports user-defined types

Solidity Contracts



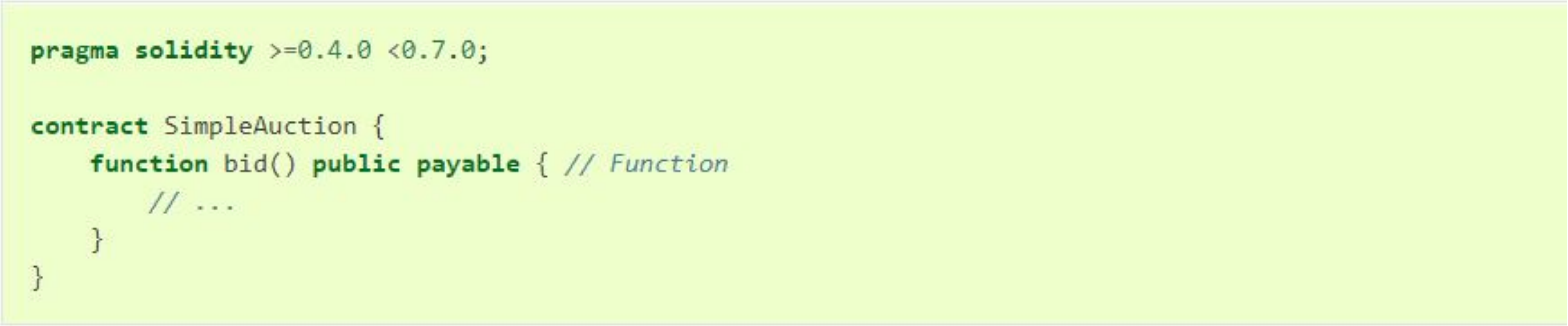
```
pragma solidity >=0.4.16 <0.7.0;

contract Simple {
    uint sum;
    function taker(uint _a, uint _b) public {
        sum = _a + _b;
    }
}
```

State Variables



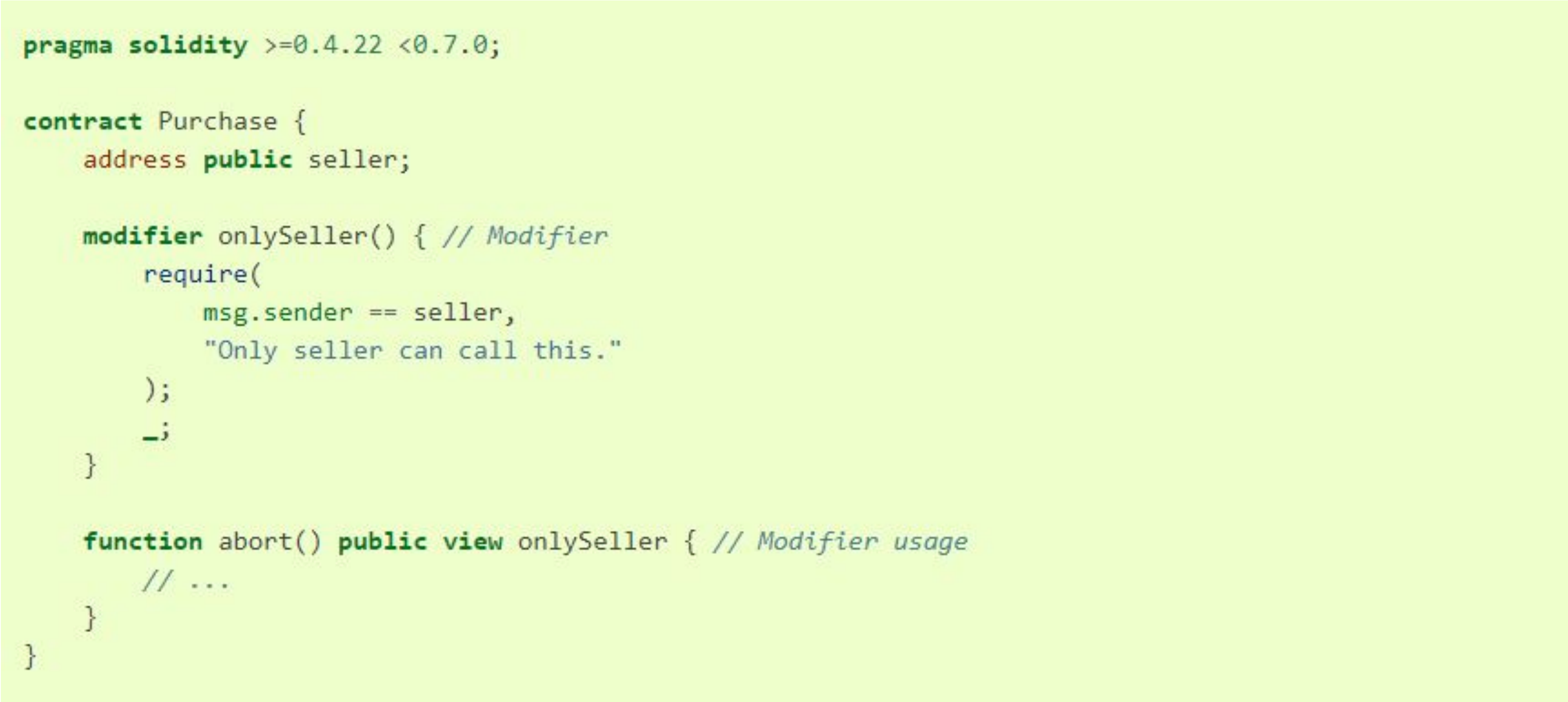
```
pragma solidity >=0.4.0 <0.7.0;

contract SimpleStorage {
    uint storedData; // State variable
    // ...
}
```

Functions



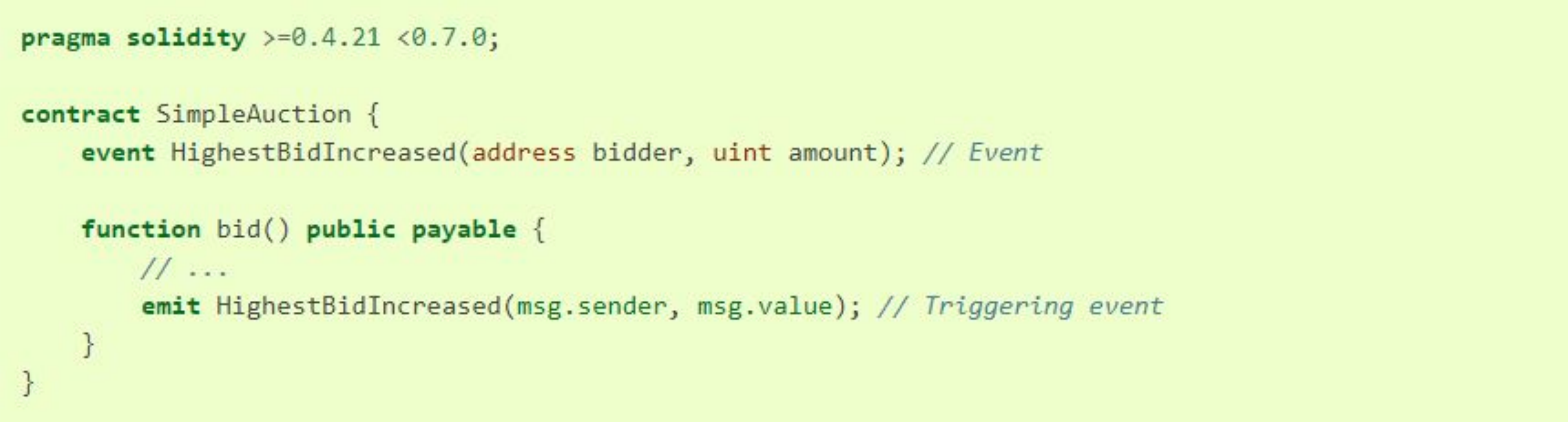
```
pragma solidity >=0.4.0 <0.7.0;

contract SimpleAuction {
    function bid() public payable { // Function
        // ...
    }
}
```

Function Modifiers



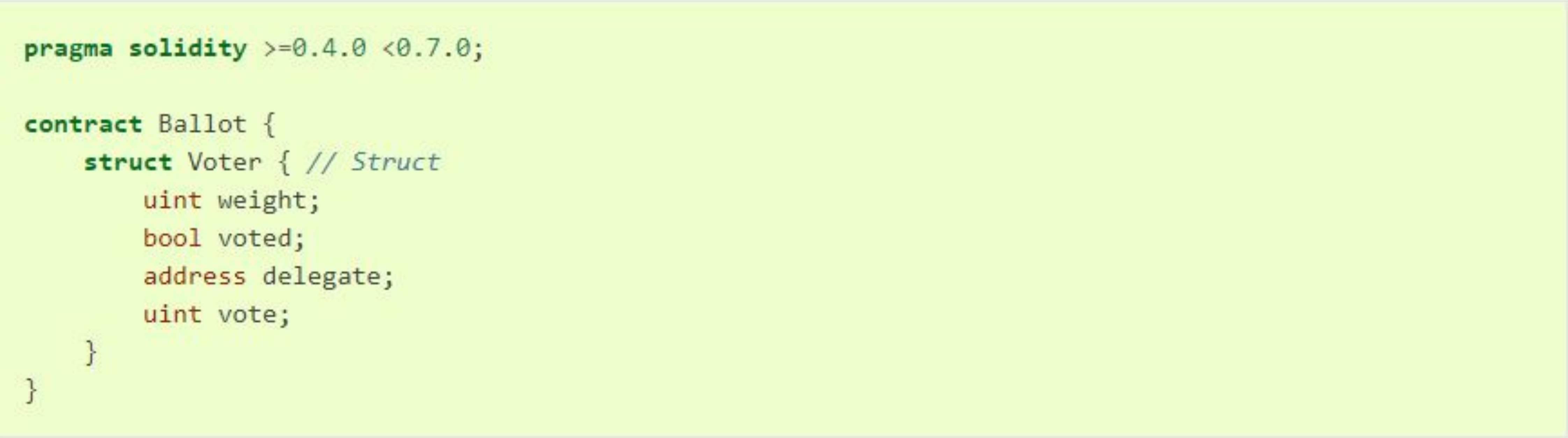
```
pragma solidity >=0.4.22 <0.7.0;

contract Purchase {
    address public seller;

    modifier onlySeller() { // Modifier
        require(
            msg.sender == seller,
            "Only seller can call this."
        );
        _;
    }

    function abort() public view onlySeller { // Modifier usage
        // ...
    }
}
```

Events



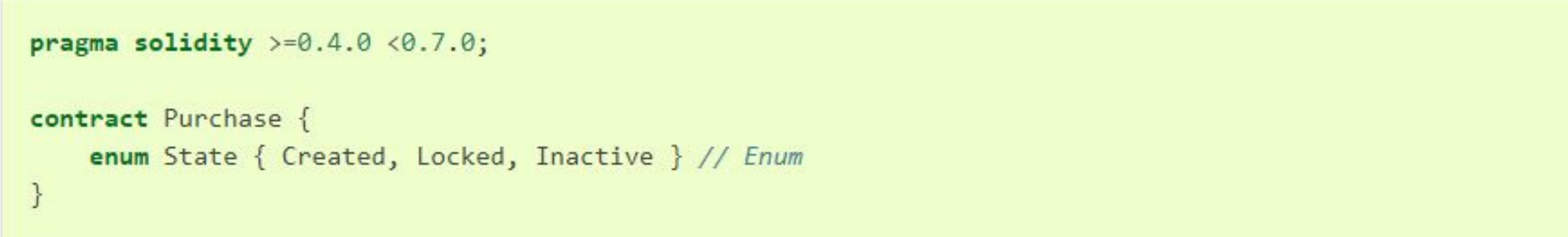
```
pragma solidity >=0.4.21 <0.7.0;

contract SimpleAuction {
    event HighestBidIncreased(address bidder, uint amount); // Event

    function bid() public payable {
        // ...
        emit HighestBidIncreased(msg.sender, msg.value); // Triggering event
    }
}
```

Structs



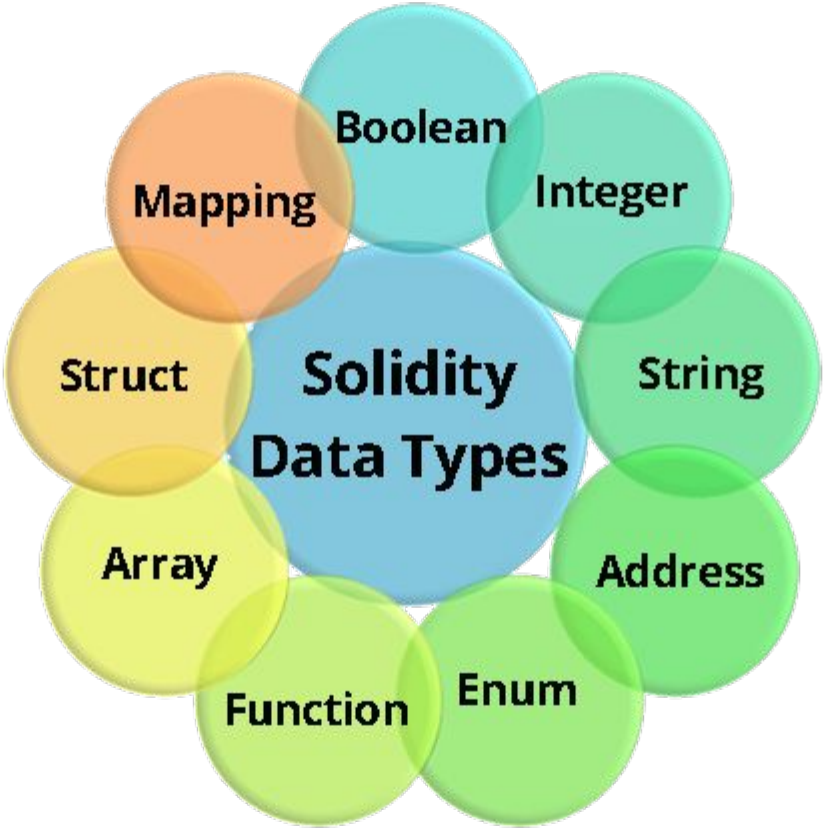
```
pragma solidity >=0.4.0 <0.7.0;

contract Ballot {
    struct Voter { // Struct
        uint weight;
        bool voted;
        address delegate;
        uint vote;
    }
}
```

Enums

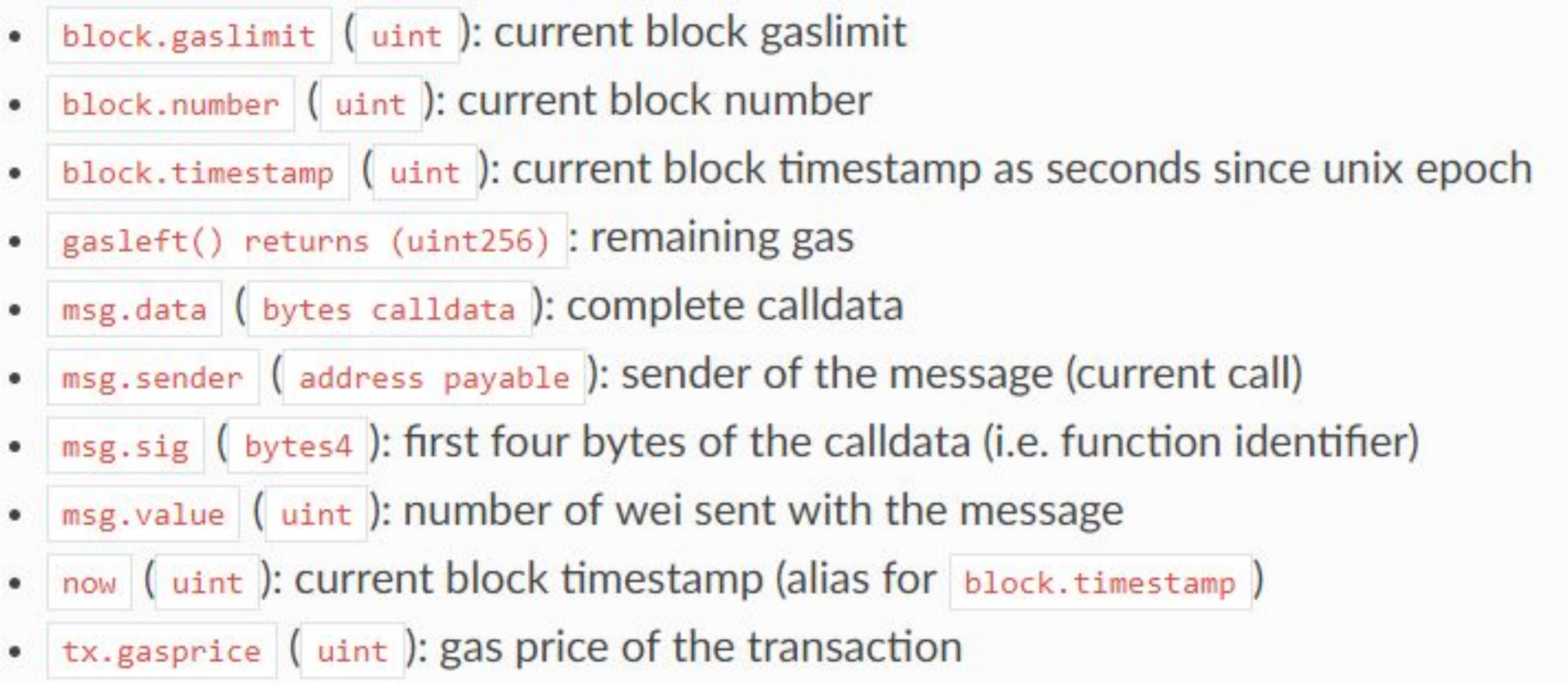


```
pragma solidity >=0.4.0 <0.7.0;

contract Purchase {
    enum State { Created, Locked, Inactive } // Enum
}
```

Types

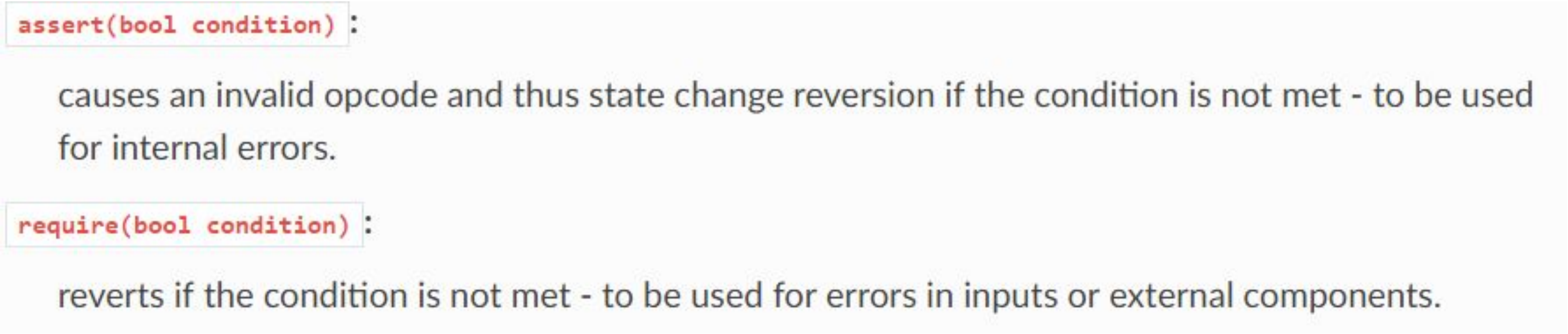

Solidity Data Types
Boolean
Integer
String
Address
Enum
Function
Array
Struct
Mapping

Globally Available Variables



- `block.gaslimit` (`uint`): current block gaslimit
- `block.number` (`uint`): current block number
- `block.timestamp` (`uint`): current block timestamp as seconds since unix epoch
- `gasleft() returns (uint256)` : remaining gas
- `msg.data` (`bytes calldata`): complete calldata
- `msg.sender` (`address payable`): sender of the message (current call)
- `msg.sig` (`bytes4`): first four bytes of the calldata (i.e. function identifier)
- `msg.value` (`uint`): number of wei sent with the message
- `now` (`uint`): current block timestamp (alias for `block.timestamp`)
- `tx.gasprice` (`uint`): gas price of the transaction

Error handling



`assert(bool condition)`:

causes an invalid opcode and thus state change reversion if the condition is not met - to be used for internal errors.

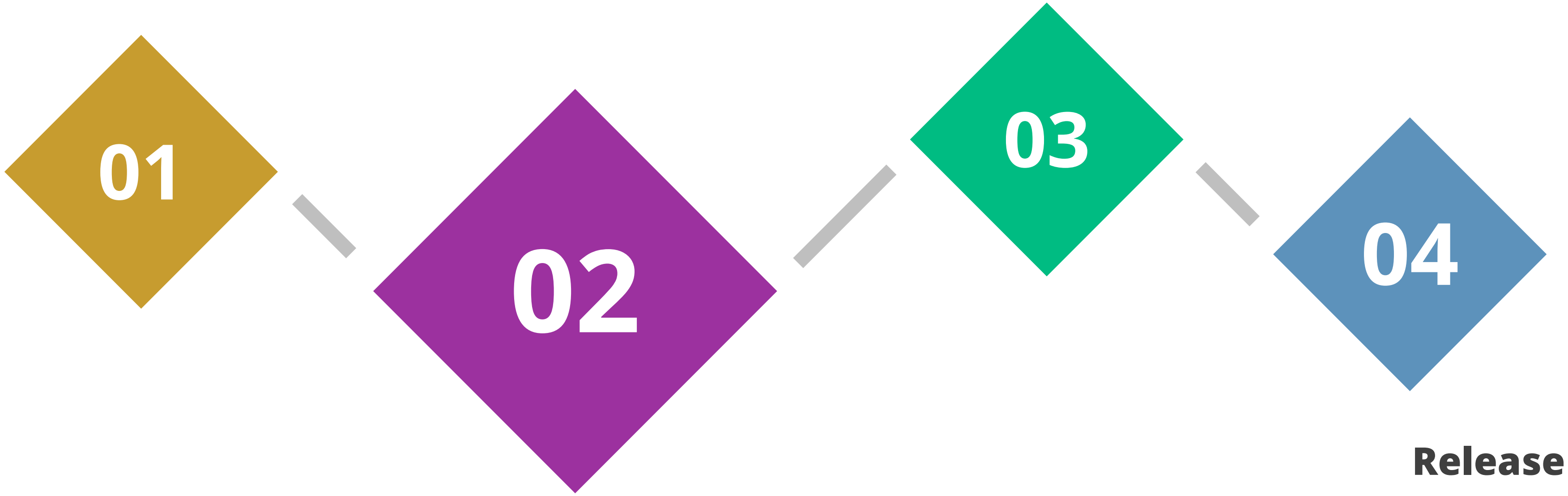
`require(bool condition)`:

reverts if the condition is not met - to be used for errors in inputs or external components.

02

Development Environment

DEVELOPMENT PROCESS



Create the Smart Contract

Test it on RegTest (DEV)

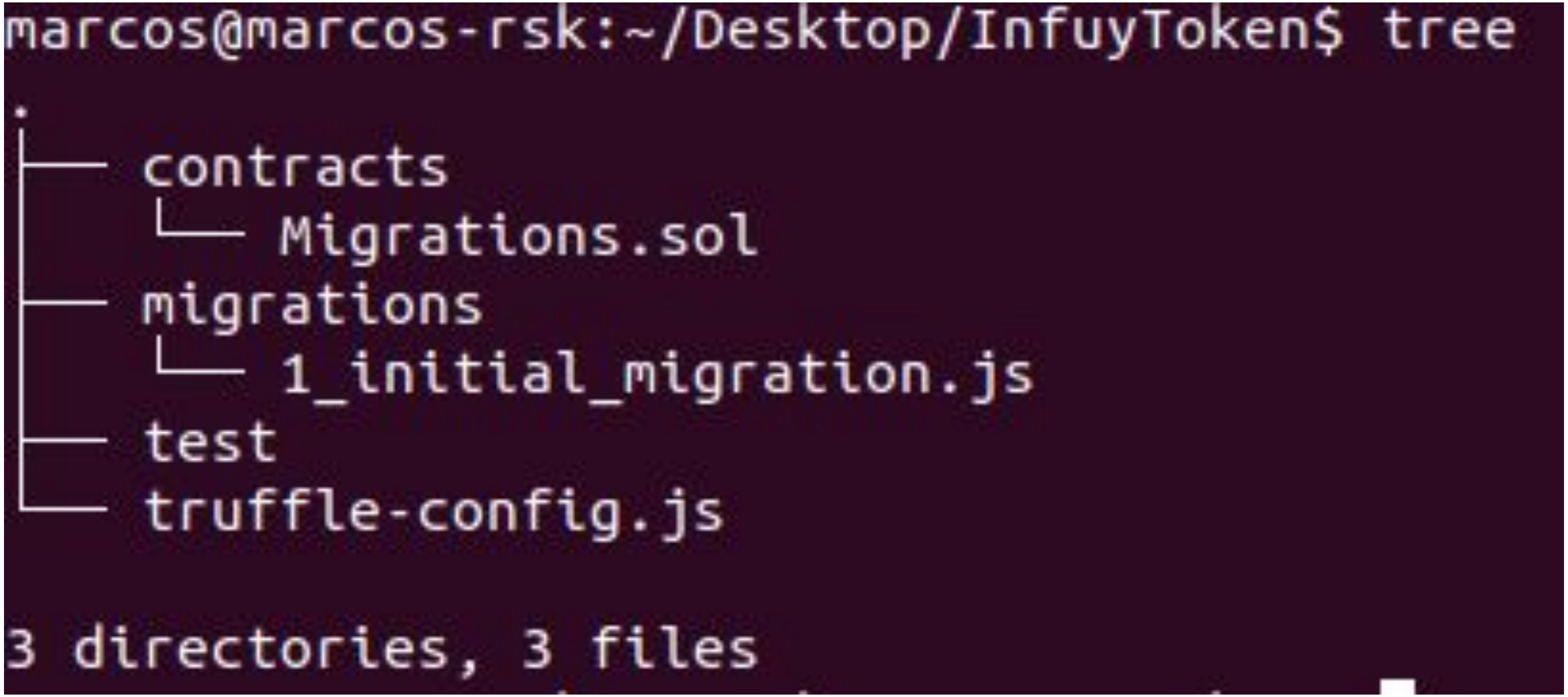
Test it on TestNet (STAGING)

Release to MainNet (PRODUCTION)

Truffle

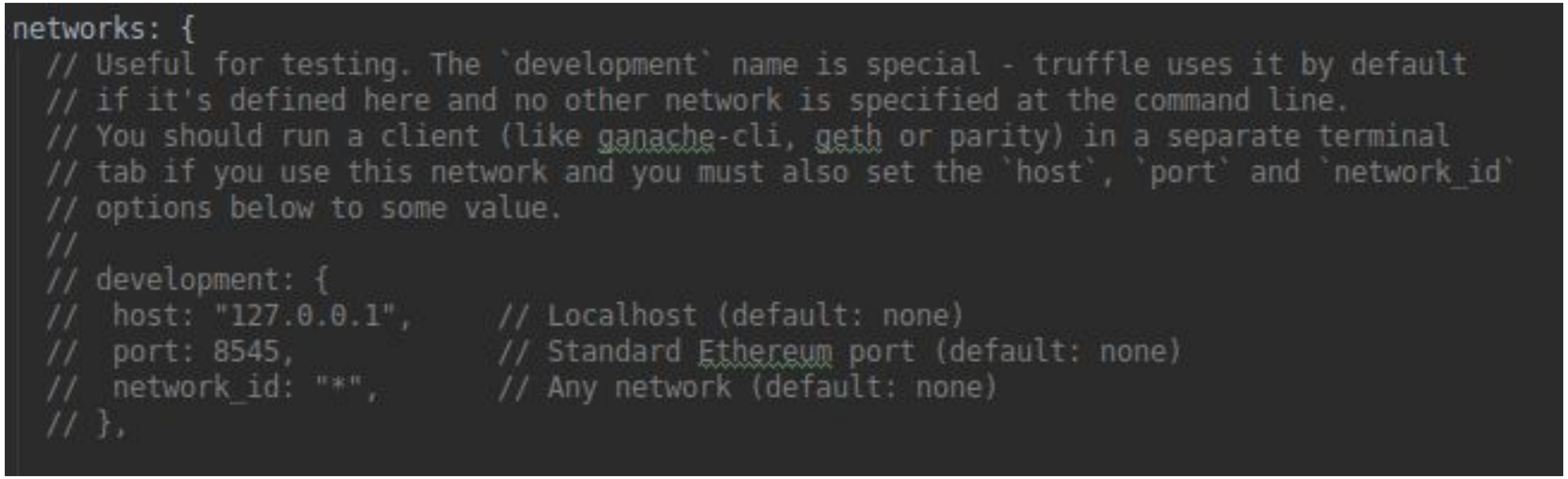
```
mkdir InfuyToken
cd InfuyToken
truffle init
```

Truffle project structure

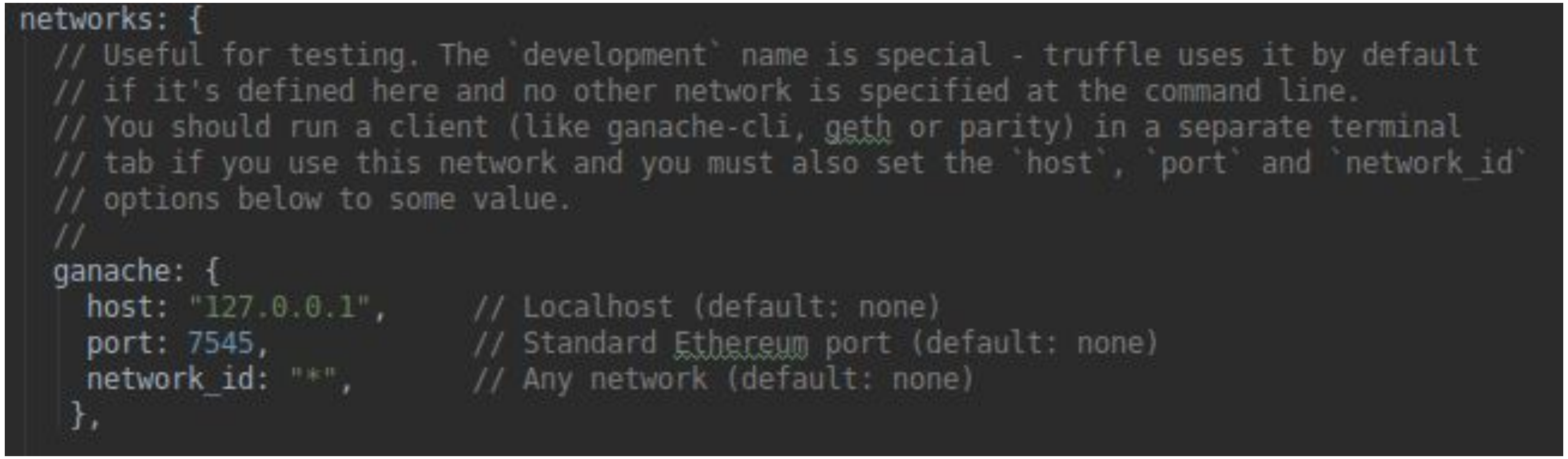


```
marcos@marcos-rsk:~/Desktop/InfuyToken$ tree
.
├── contracts
│   └── Migrations.sol
├── migrations
│   └── 1_initial_migration.js
├── test
└── truffle-config.js

3 directories, 3 files
```

truffle-config.js



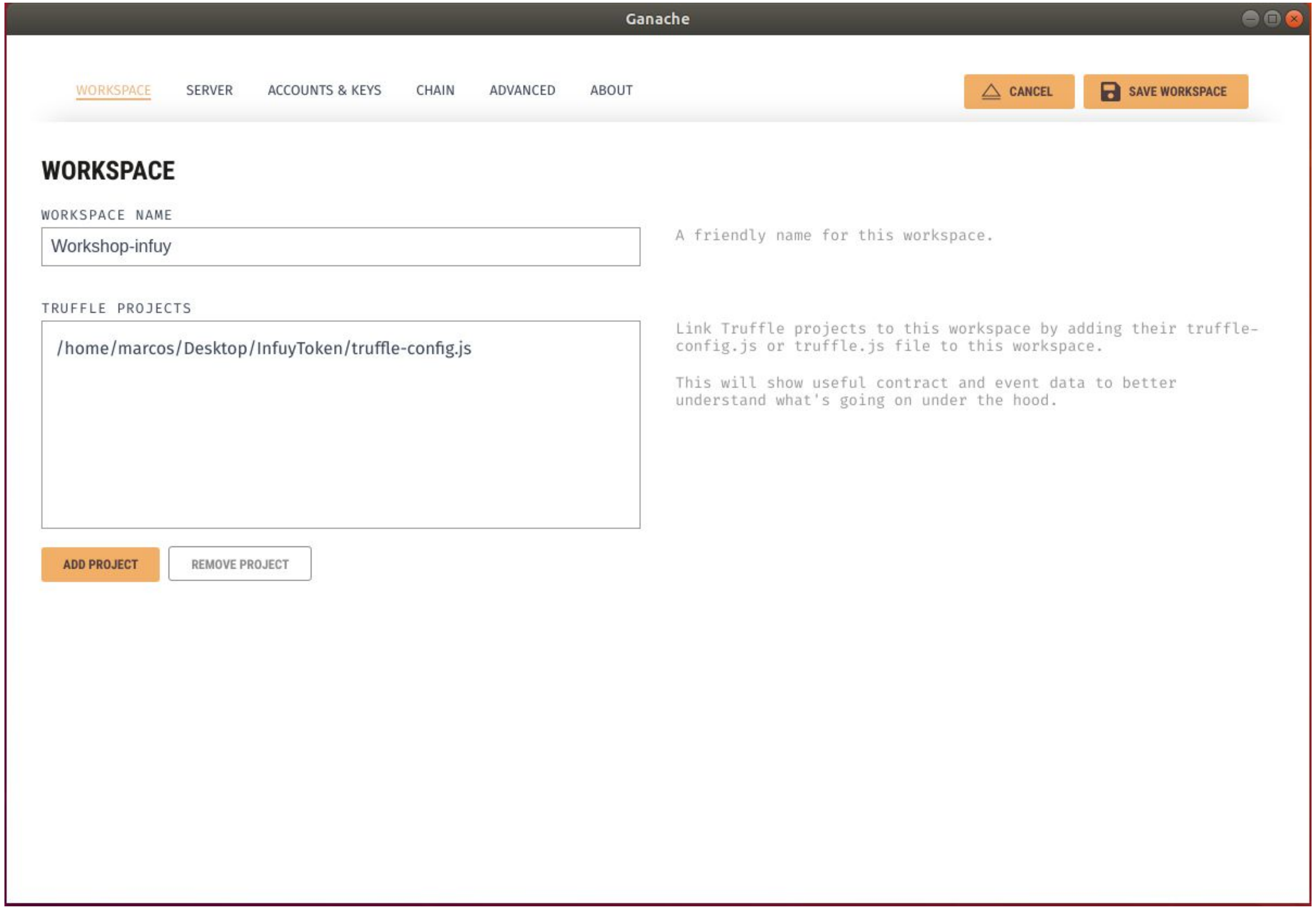
```
networks: {
  // Useful for testing. The `development` name is special - truffle uses it by default
  // if it's defined here and no other network is specified at the command line.
  // You should run a client (like ganache-cli, geth or parity) in a separate terminal
  // tab if you use this network and you must also set the `host`, `port` and `network_id`
  // options below to some value.
  //
  // development: {
  //  host: "127.0.0.1",     // Localhost (default: none)
  //  port: 8545,            // Standard Ethereum port (default: none)
  //  network_id: "*",       // Any network (default: none)
  // },
```



```
networks: {
  // Useful for testing. The `development` name is special - truffle uses it by default
  // if it's defined here and no other network is specified at the command line.
  // You should run a client (like ganache-cli, geth or parity) in a separate terminal
  // tab if you use this network and you must also set the `host`, `port` and `network_id`
  // options below to some value.
  //
  ganache: {
    host: "127.0.0.1",     // Localhost (default: none)
    port: 7545,            // Standard Ethereum port (default: none)
    network_id: "*",       // Any network (default: none)
  },
```

Ganache

Ganache Setup (1/2)



WORKSPACE

WORKSPACE NAME

Workshop-infuy

A friendly name for this workspace.

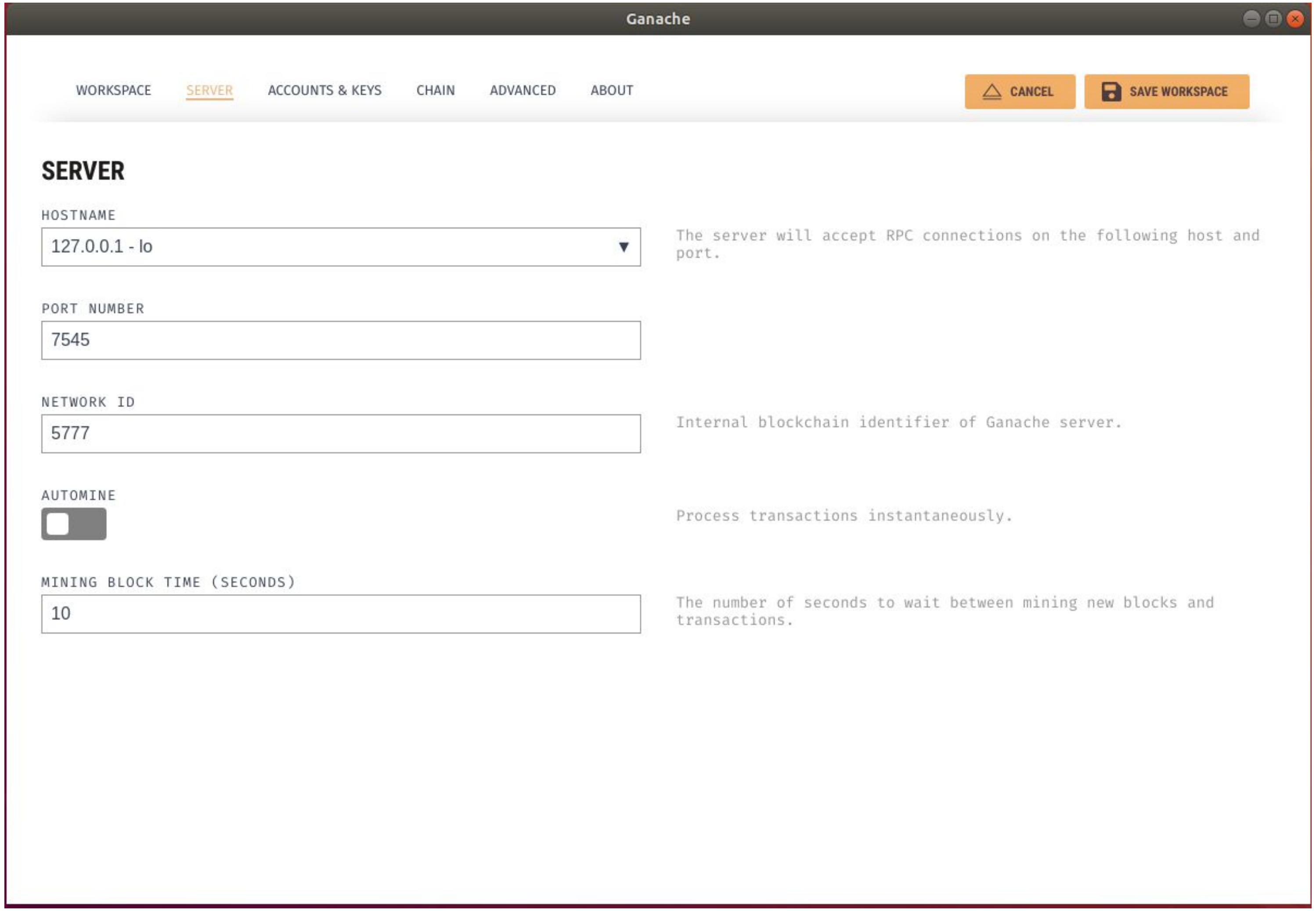
TRUFFLE PROJECTS

/home/marcos/Desktop/InfuyToken/truffle-config.js

Link Truffle projects to this workspace by adding their truffle-config.js or truffle.js file to this workspace.

This will show useful contract and event data to better understand what's going on under the hood.

Ganache Setup (2/2)



SERVER

HOSTNAME

127.0.0.1 - lo ▼

The server will accept RPC connections on the following host and port.

PORT NUMBER

7545

NETWORK ID

5777

Internal blockchain identifier of Ganache server.

AUTOMINE

○

Process transactions instantaneously.

MINING BLOCK TIME (SECONDS)

10

The number of seconds to wait between mining new blocks and transactions.

Web3.js

03

Crafting the InfuyToken

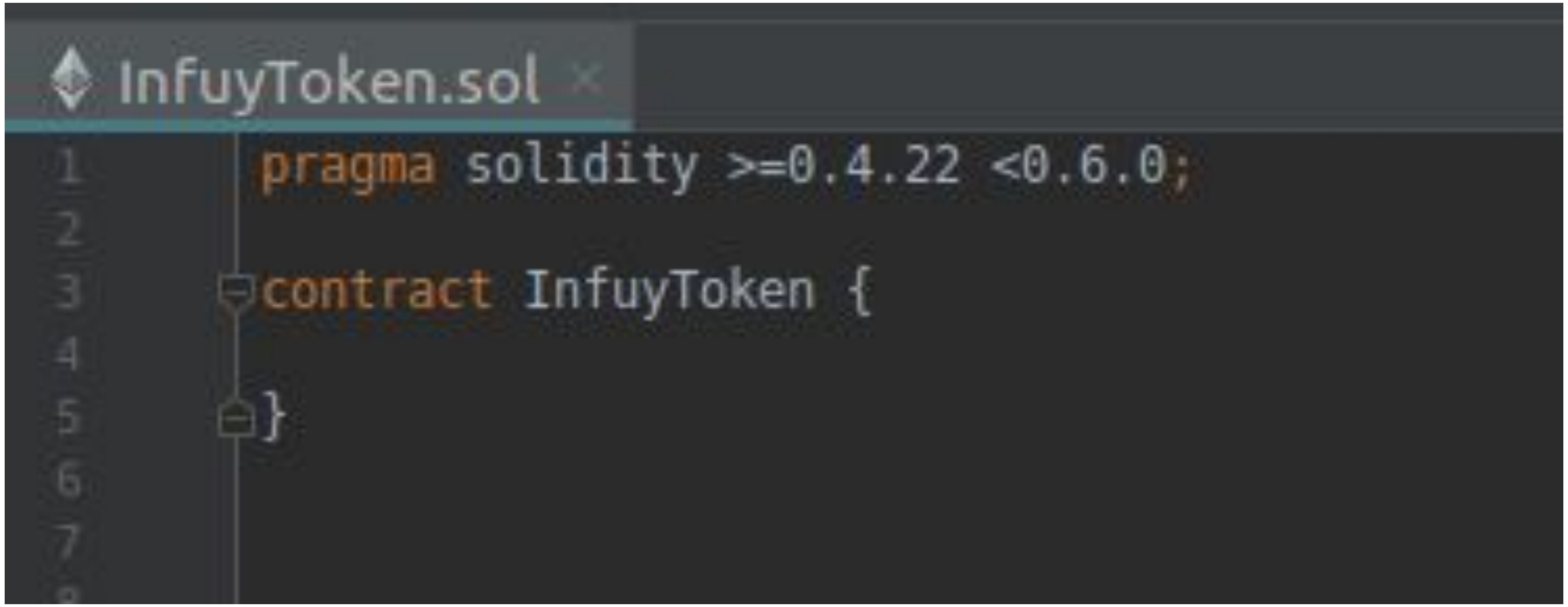
Lets code! 💪

InfuyToken requirements

1. The InfuyToken must be a Smart Contract
2. The smart contract must store the balances of the accounts
3. The owner of the Smart Contract must have 100 InfuyTokens
4. Anybody can query the balance of an account
5. Must provide a way to send balance to other accounts
6. An user can only send tokens if has enough balance
7. Must emit an event when transfer succeeds (for dapps)

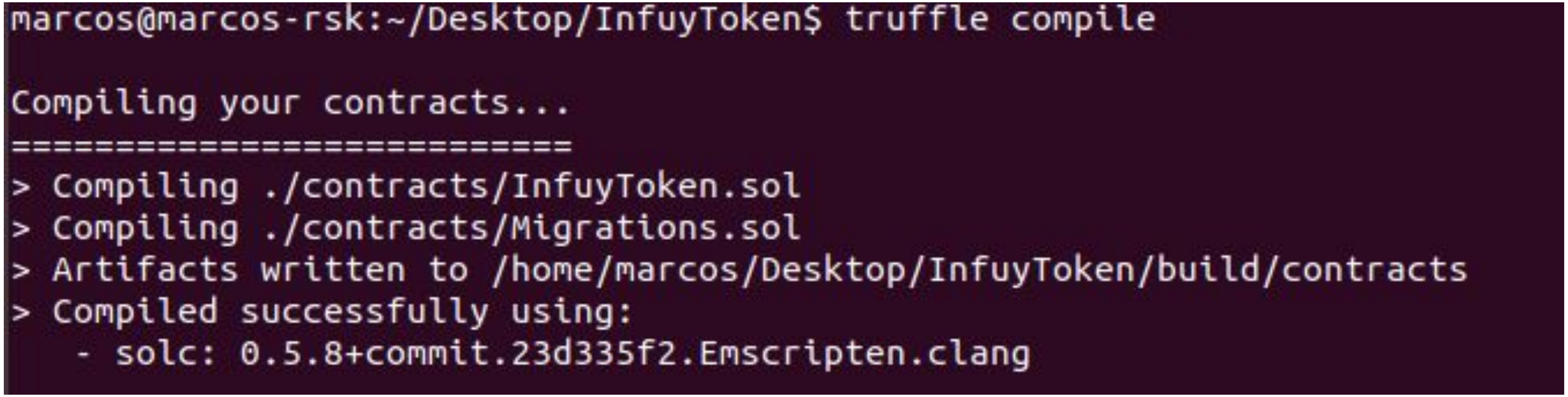
1. The InfuyToken must be a smart contract

1. The InfuyToken must be a smart contract



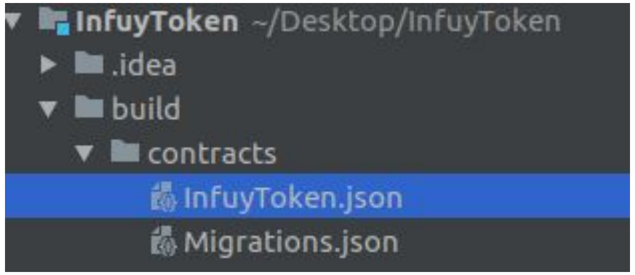
```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

}
```

1. The InfuyToken must be a smart contract

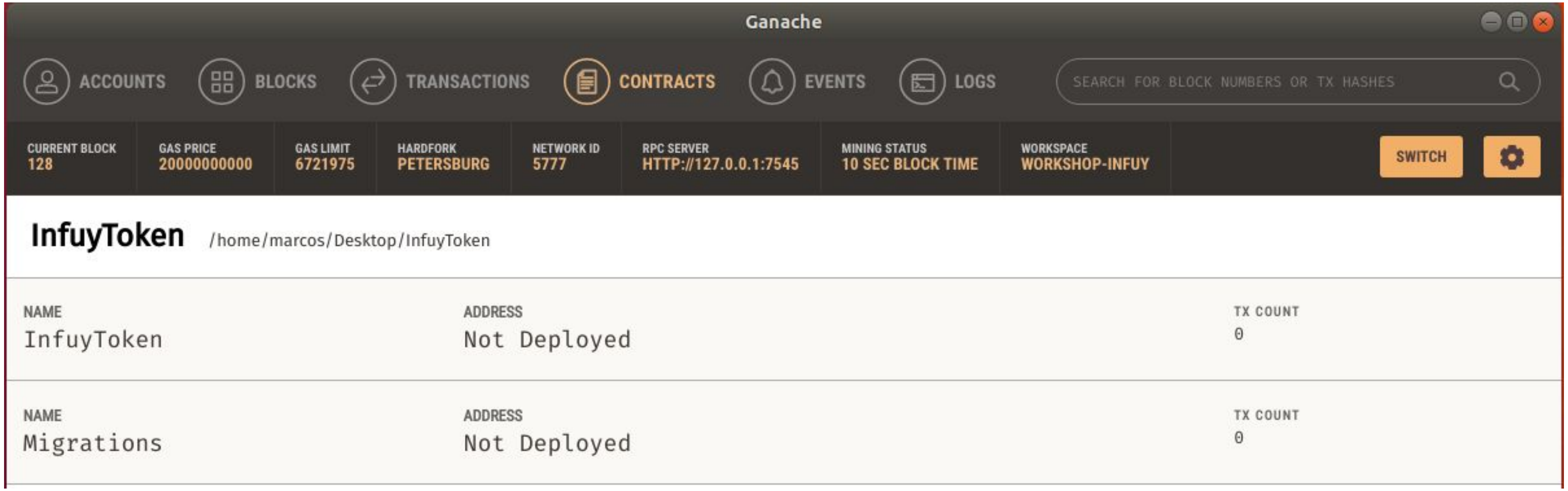


```
marcos@marcos-rsk:~/Desktop/InfuyToken$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/InfuyToken.sol
> Compiling ./contracts/Migrations.sol
> Artifacts written to /home/marcos/Desktop/InfuyToken/build/contracts
> Compiled successfully using:
   - solc: 0.5.8+commit.23d335f2.Emscripten.clang
```

1. The InfuyToken must be a smart contract



```
▼ InfuyToken ~/Desktop/InfuyToken
  ▶ .idea
  ▼ build
    ▼ contracts
        InfuyToken.json
        Migrations.json
```



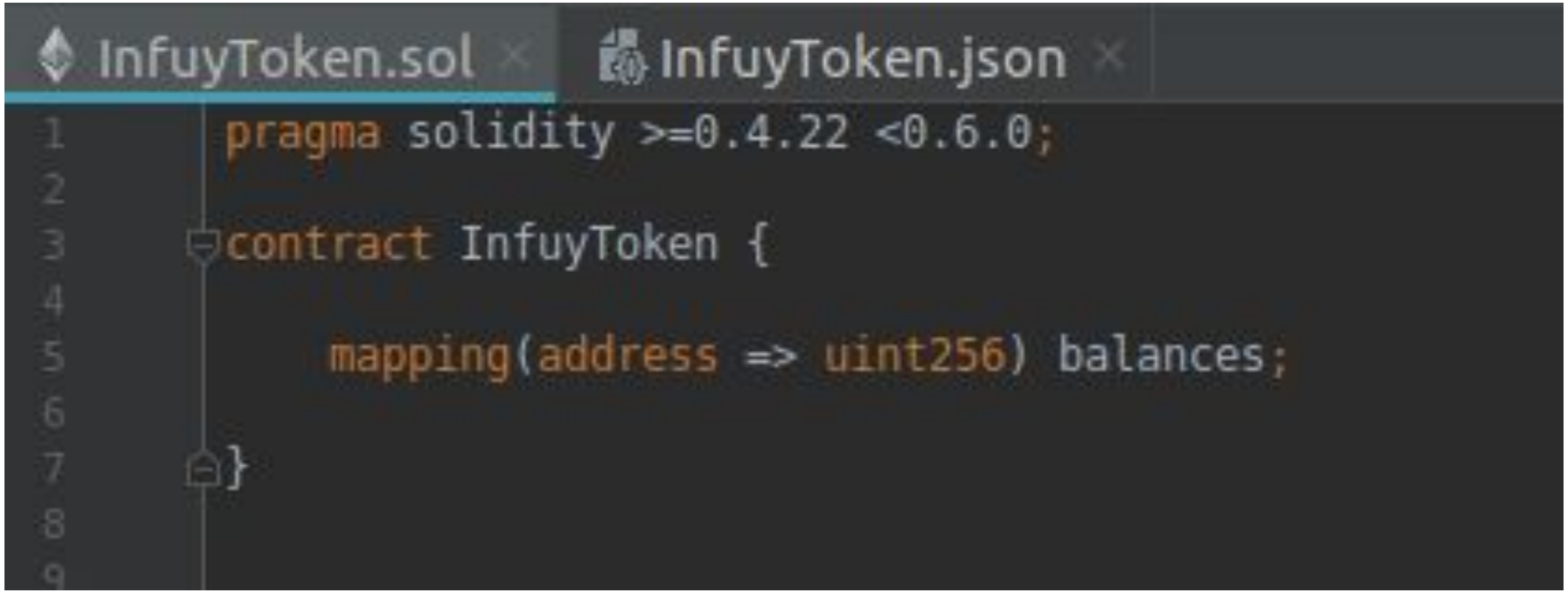
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
128	20000000000	6721975	PETERSBURG	5777	HTTP://127.0.0.1:7545	10 SEC BLOCK TIME	WORKSHOP-INFUY

InfuyToken /home/marcos/Desktop/InfuyToken

NAME	ADDRESS	TX COUNT
InfuyToken	Not Deployed	0
Migrations	Not Deployed	0

2. Must store the balances of the accounts

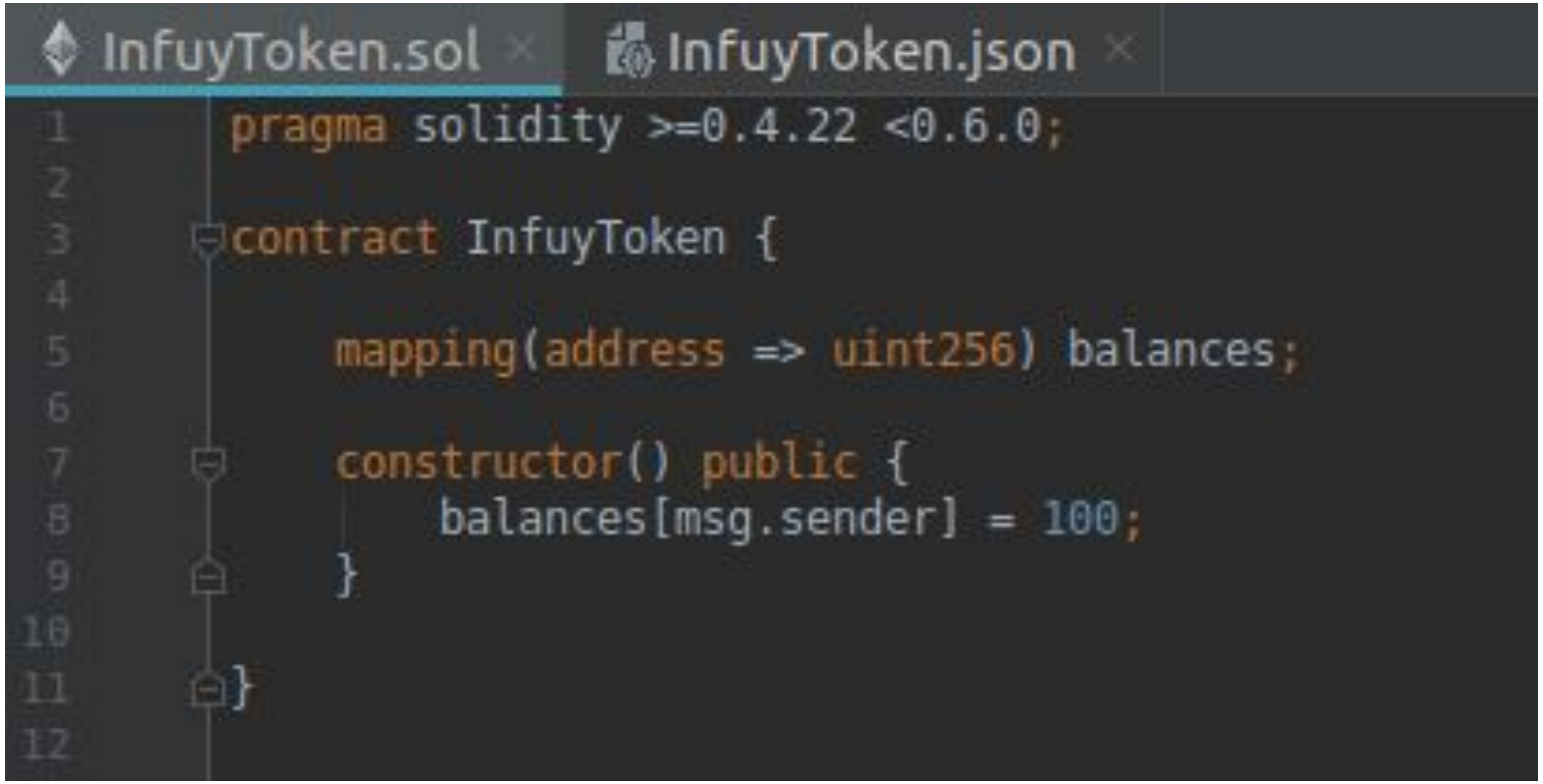
2. Must store the balances of the accounts



```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

    mapping(address => uint256) balances;

}
```

3. The owner must have 100 InfuyTokens

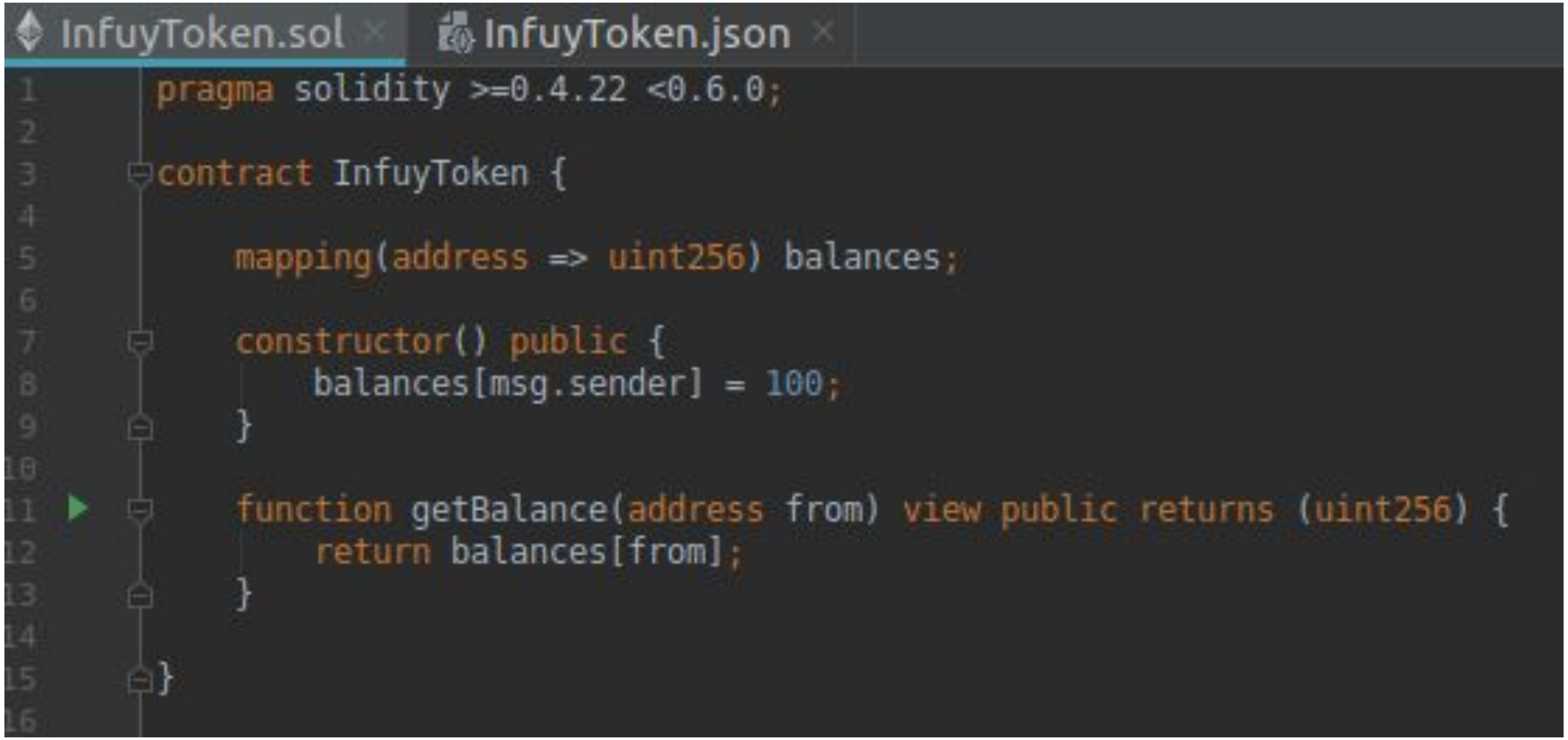
3. The owner must have 100 InfuyTokens



```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

    mapping(address => uint256) balances;

    constructor() public {
        balances[msg.sender] = 100;
    }

}

```

4. Anybody can query the balance of an account

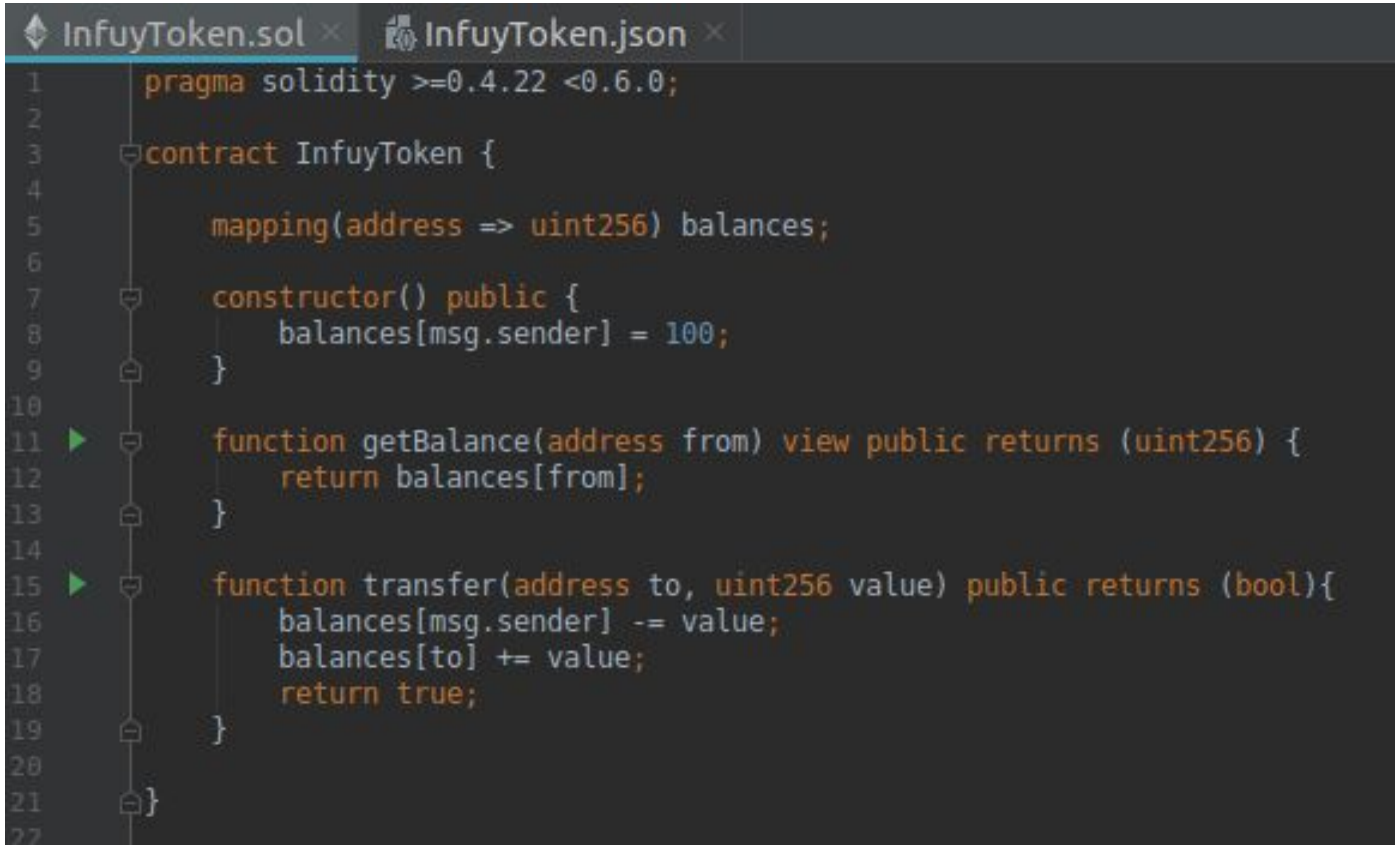
4. Anybody can query the balance of an account



```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

    mapping(address => uint256) balances;

    constructor() public {
        balances[msg.sender] = 100;
    }

    function getBalance(address from) view public returns (uint256) {
        return balances[from];
    }

}
```

5. Provide a way to send balance to others

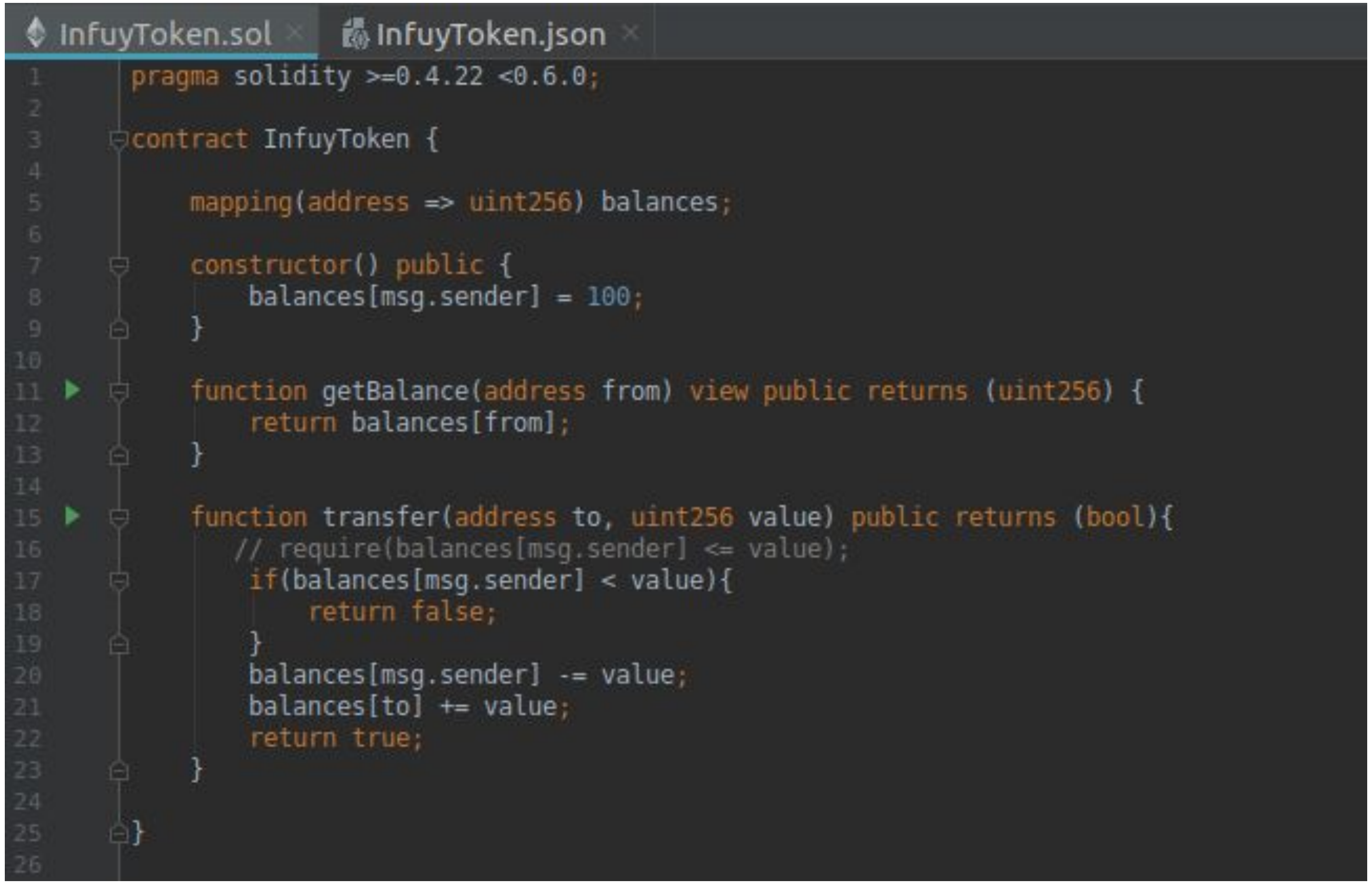
5. Provide a way to send balance to others



```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

    mapping(address => uint256) balances;

    constructor() public {
        balances[msg.sender] = 100;
    }

    function getBalance(address from) view public returns (uint256) {
        return balances[from];
    }

    function transfer(address to, uint256 value) public returns (bool){
        balances[msg.sender] -= value;
        balances[to] += value;
        return true;
    }

}
```

6. An user can only send tokens if has enough balance

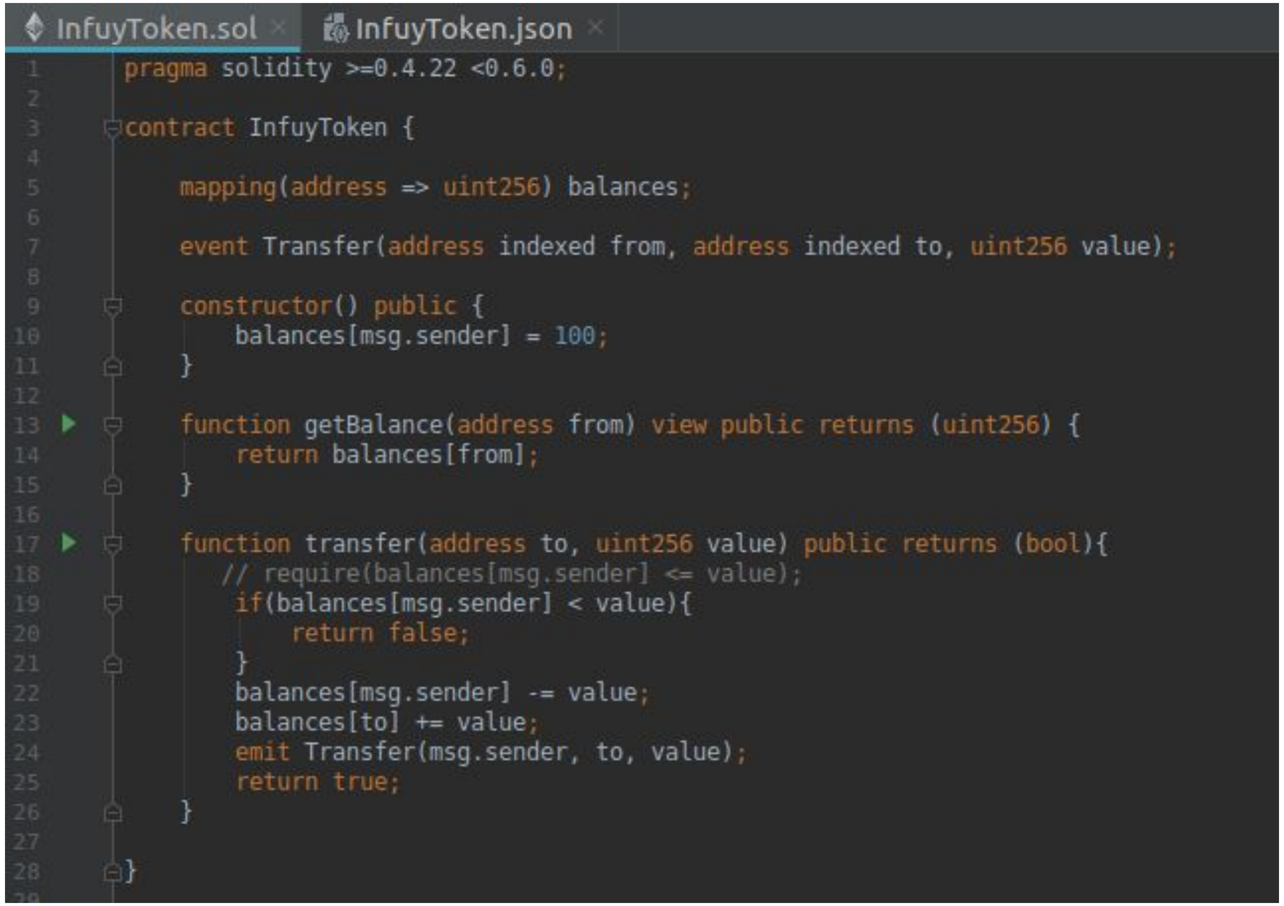
6. An user can only send tokens if has enough balance



```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

    mapping(address => uint256) balances;

    constructor() public {
        balances[msg.sender] = 100;
    }

    function getBalance(address from) view public returns (uint256) {
        return balances[from];
    }

    function transfer(address to, uint256 value) public returns (bool){
       // require(balances[msg.sender] <= value);
        if(balances[msg.sender] < value){
            return false;
        }
        balances[msg.sender] -= value;
        balances[to] += value;
        return true;
    }

}
```

7. Must emit an event when transfer succeeds

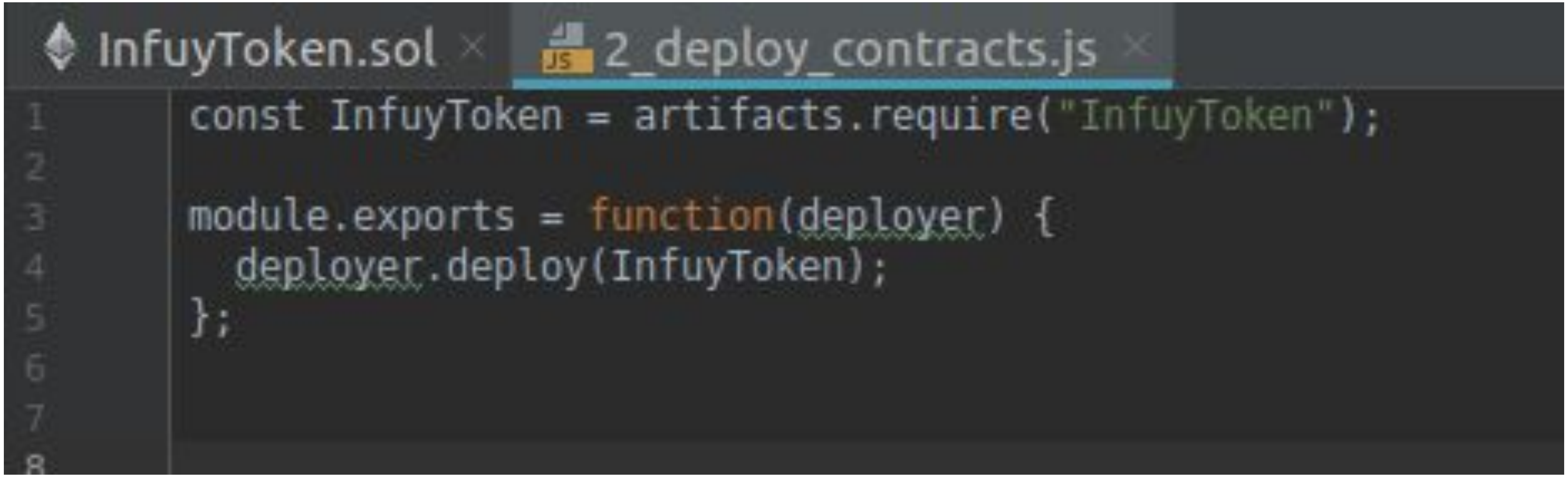
7. Must emit an event when transfer succeeds



```
pragma solidity >=0.4.22 <0.6.0;

contract InfuyToken {

    mapping(address => uint256) balances;

    event Transfer(address indexed from, address indexed to, uint256 value);

    constructor() public {
        balances[msg.sender] = 100;
    }

    function getBalance(address from) view public returns (uint256) {
        return balances[from];
    }

    function transfer(address to, uint256 value) public returns (bool){
       // require(balances[msg.sender] <= value);
        if(balances[msg.sender] < value){
            return false;
        }
        balances[msg.sender] -= value;
        balances[to] += value;
        emit Transfer(msg.sender, to, value);
        return true;
    }

}
```

Deploy!

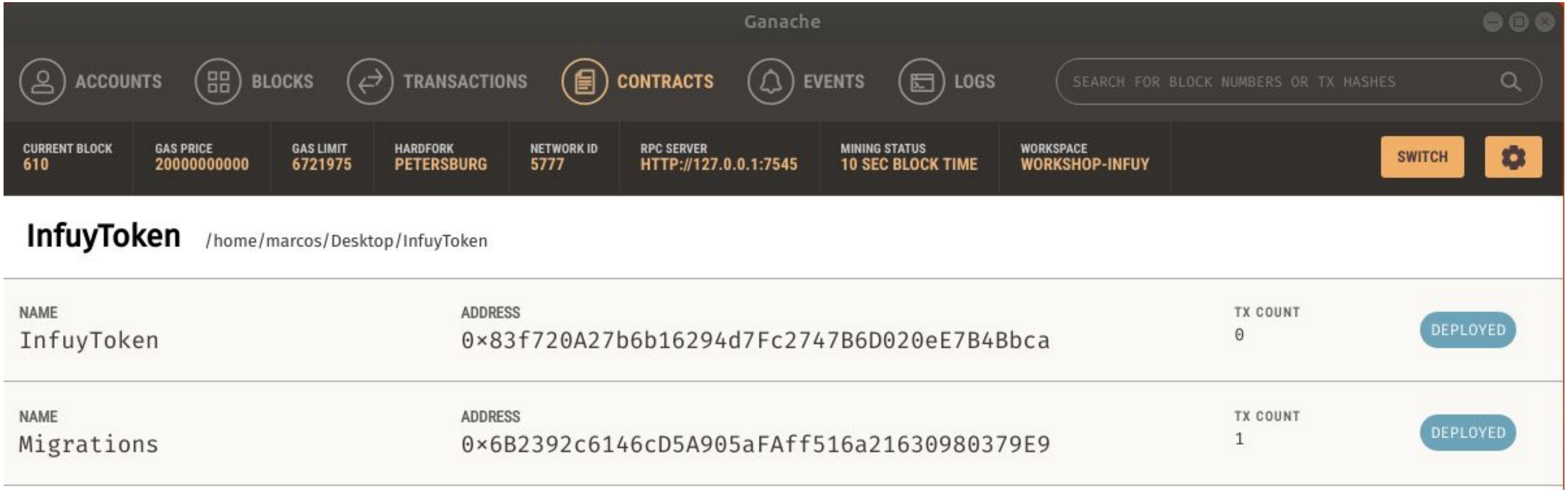
2. deploy_contracts.js



```
const InfuyToken = artifacts.require("InfuyToken");

module.exports = function(deployer) {
  deployer.deploy(InfuyToken);
};
```

```
truffle console --network ganache
compile
migrate
```

Deploy

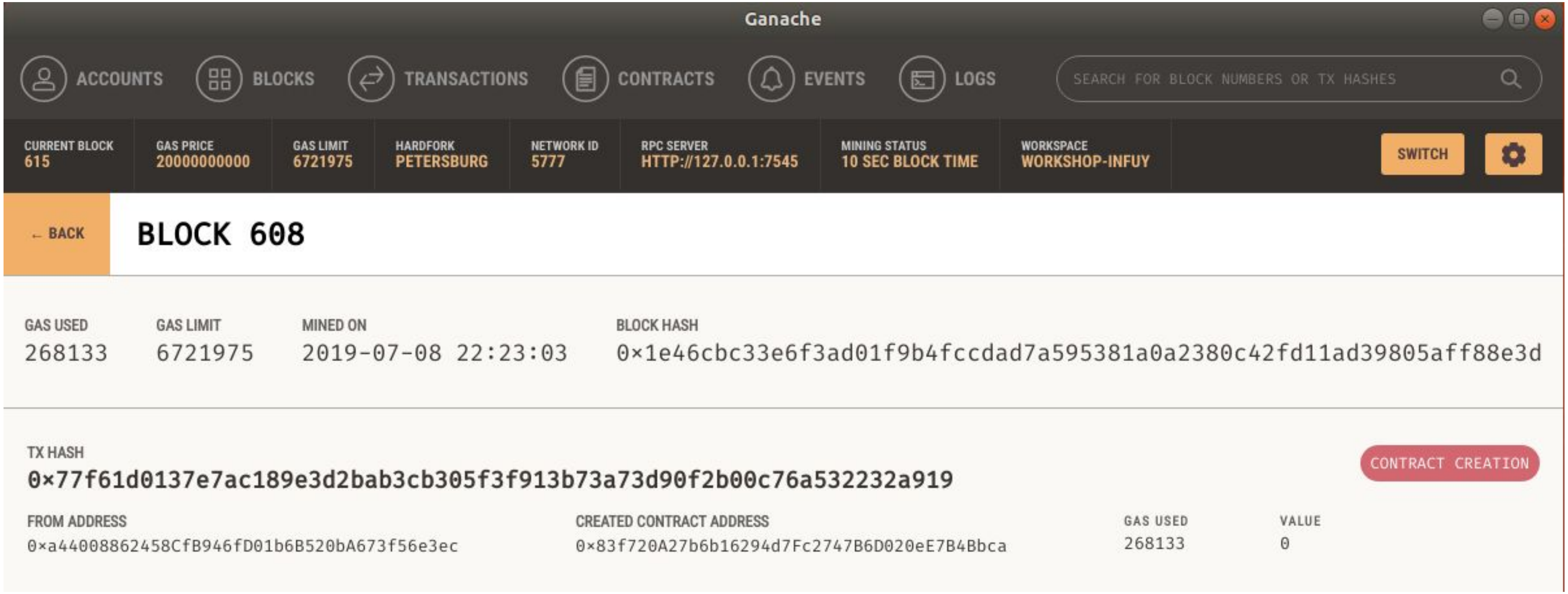


CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
610	20000000000	6721975	PETERSBURG	5777	HTTP://127.0.0.1:7545	10 SEC BLOCK TIME	WORKSHOP-INFUY

InfuyToken /home/marcos/Desktop/InfuyToken

NAME	ADDRESS	TX COUNT	
InfuyToken	0×83f720A27b6b16294d7Fc2747B6D020eE7B4Bbca	0	DEPLOYED
Migrations	0×6B2392c6146cD5A905aFAff516a21630980379E9	1	DEPLOYED

Deploy



CURRENT BLOCK 615

GAS PRICE 20000000000

GAS LIMIT 6721975

HARDFORK PETERSBURG

NETWORK ID 5777

RPC SERVER HTTP://127.0.0.1:7545

MINING STATUS 10 SEC BLOCK TIME

WORKSPACE WORKSHOP-INFUY

BLOCK 608

GAS USED 268133

GAS LIMIT 6721975

MINED ON 2019-07-08 22:23:03

BLOCK HASH 0×1e46cbc33e6f3ad01f9b4fccdad7a595381a0a2380c42fd11ad39805aff88e3d

TX HASH 0×77f61d0137e7ac189e3d2bab3cb305f3f913b73a73d90f2b00c76a532232a919

FROM ADDRESS 0×a44008862458CfB946fD01b6B520bA673f56e3ec

CREATED CONTRACT ADDRESS 0×83f720A27b6b16294d7Fc2747B6D020eE7B4Bbca

GAS USED 268133

VALUE 0

Sending Tokens!

https://github.com/marcosmartinez7/infuy-sc-workshop

04

InfuyToken on TestNet

https://myetherwallet.com

https://github.com/marcosmartinez7/infuy-sc-workshop

Thanks!

https://forms.gle/gY1E5CJZH8u7rxU66