Implementation of Smart Contracts in Ethereum

Department of Computer Science and Engineering

Indian Institute of Technology, Jodhpur

Correctness of Blockchain

- Algorithm for Blockchain
- Logically or Mathematically Proof
- Implementation based the Programming Language
- Tools

Different tools provide different functionality

	Tools / Activities	Remix	Ganache	MyEtherWallet	Geth
1	Configure the Blockchain	-	-	-	+
2	Deploy the Blockchain	Not Persistent	+	-	+
3	Develop the contract	+	-	-	+
4	Compile the contract	+	-	-	+
5	Create user account	+	+	+	+
6	Deploy the contract	+	-	+	+
7	Create the UI for interacting	+	-	+	+
8	Run the client	+	-	+	+
9	Interact with the contract & have fun	+	-	+	+
10	Monitor the execution	-	+	-	+

References

- https://remix.ethereum.org/
- http://truffleframework.com/ganache/
- https://github.com/kvhnuke/etherwallet/releases/tag/v3.21.06

Use which tool for what purpose?

- ❖ Use Geth for everything?
  - Powerful but command-line only

- ❖ What should I use?
  - As a starting point for developing contracts – mostly Remix

- ❖ What cannot Remix do?
  - Configure the blockchain
  - Create real (non-test) user accounts and transfer funds between user accounts
  - Monitor the execution
  - Other advanced operations

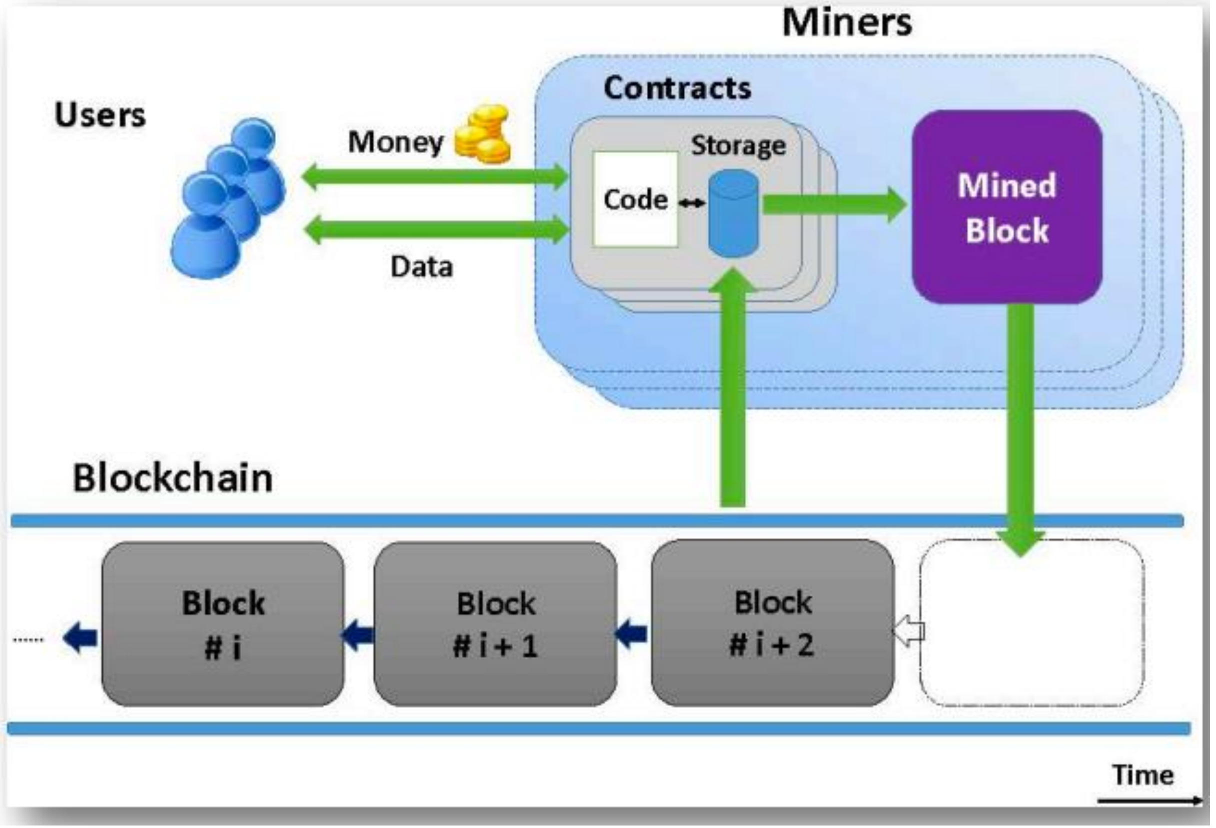
Use which tool for what purpose?

- Why use Ganache?
  - To inspect and monitor the execution
  - To visualize certain elements in a better way

- Why use MyEtherWallet?
- To create a personal wallet (real user account), transfer funds between user accounts, and interact with contracts
- Metamask as another alternative

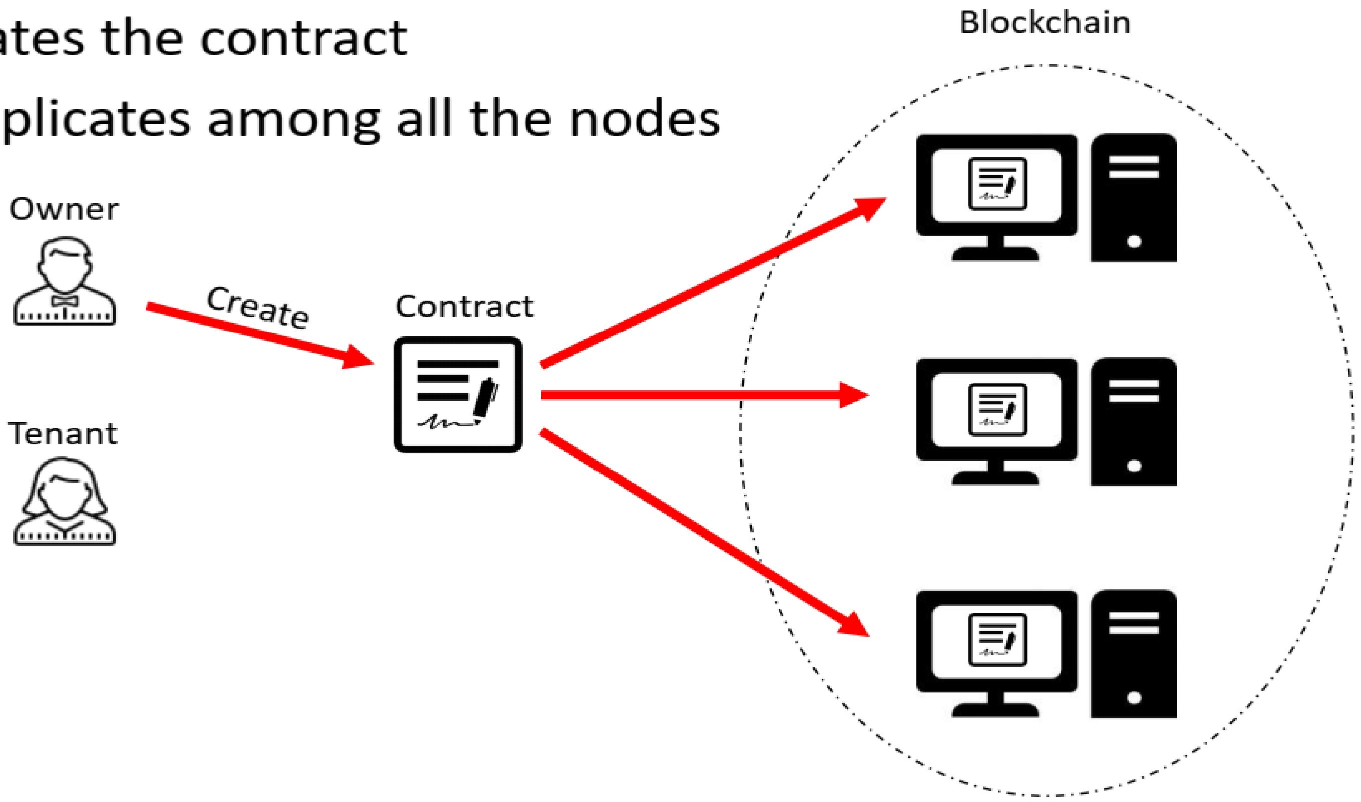
Smart Contracts

- In the form of code
- Stored on a blockchain
- Executes under given conditions



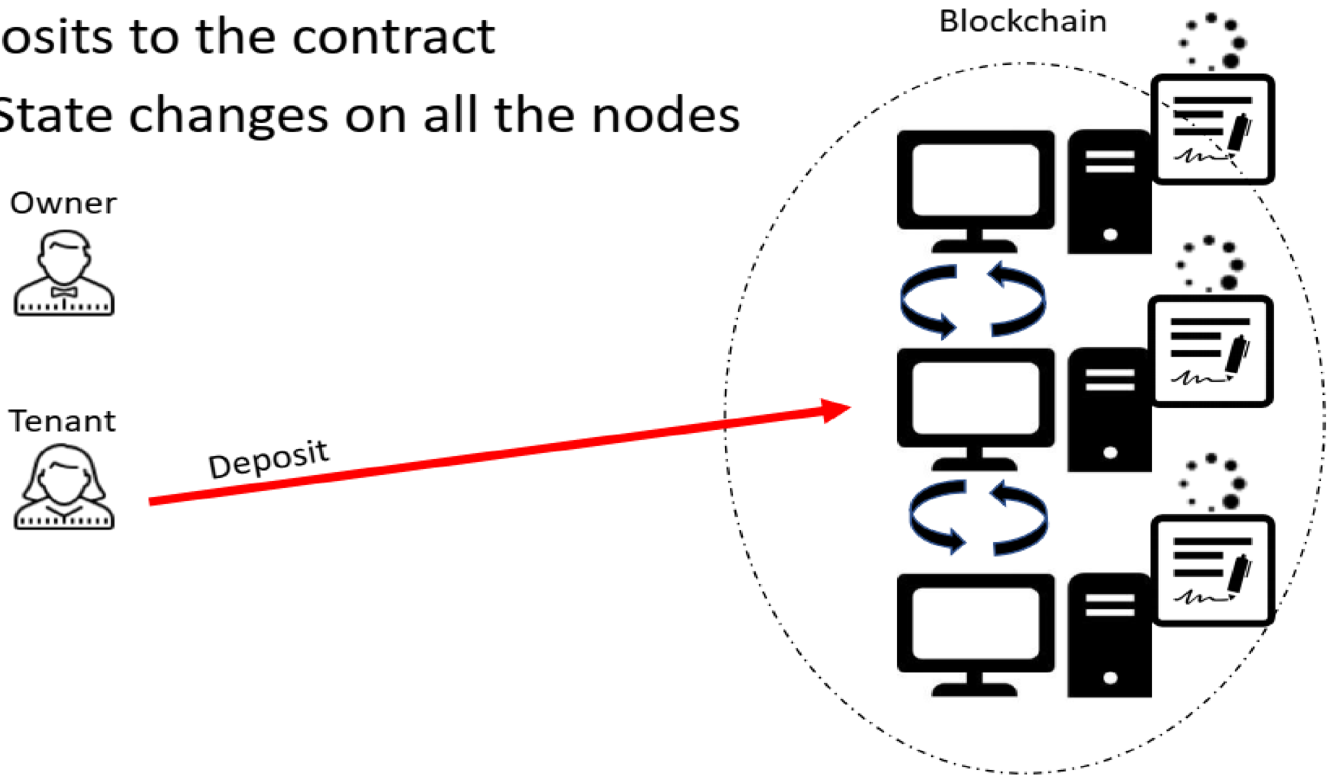
Smart Contracts Example

- Owner creates the contract
- Contract replicates among all the nodes



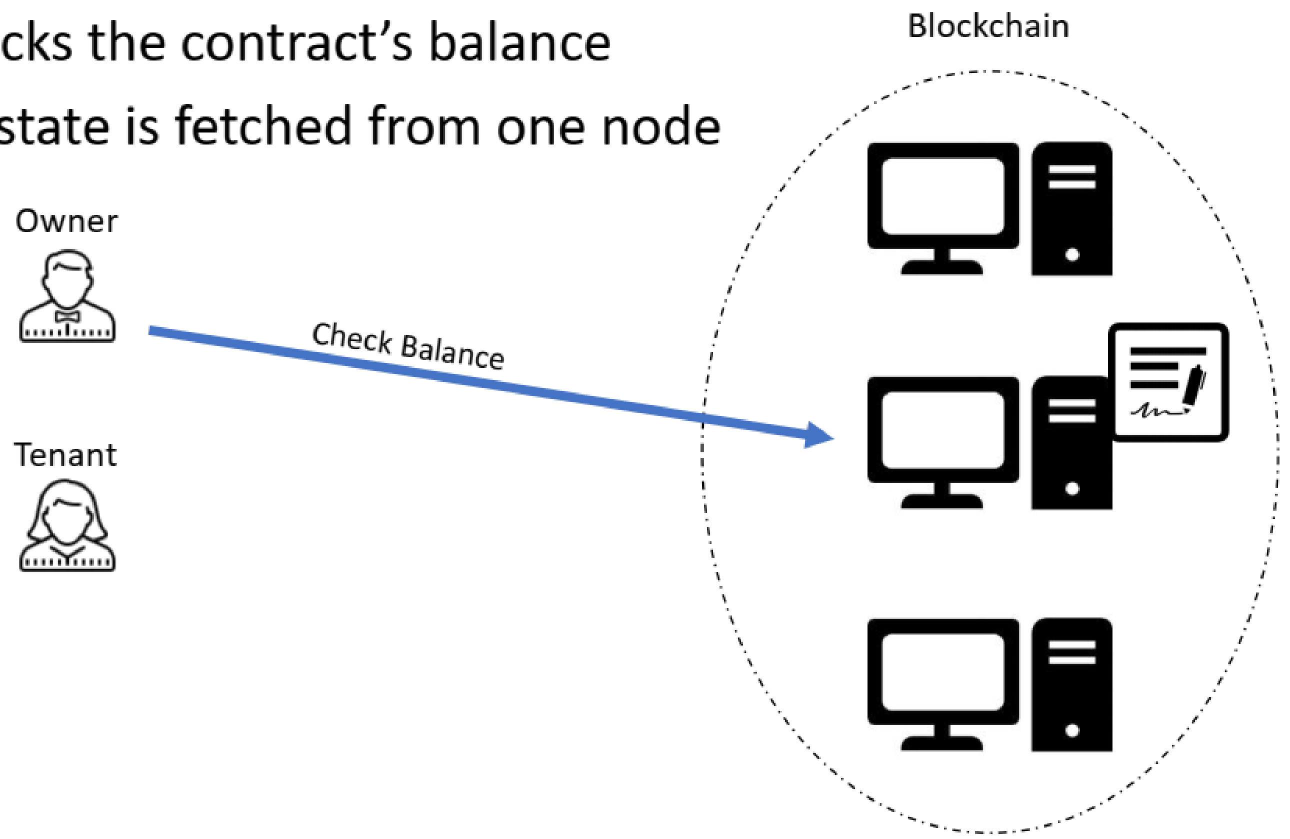
Smart Contracts Example

- Tenant deposits to the contract
- Contract's State changes on all the nodes



Smart Contracts Example

- Owner checks the contract's balance
- Contract's state is fetched from one node

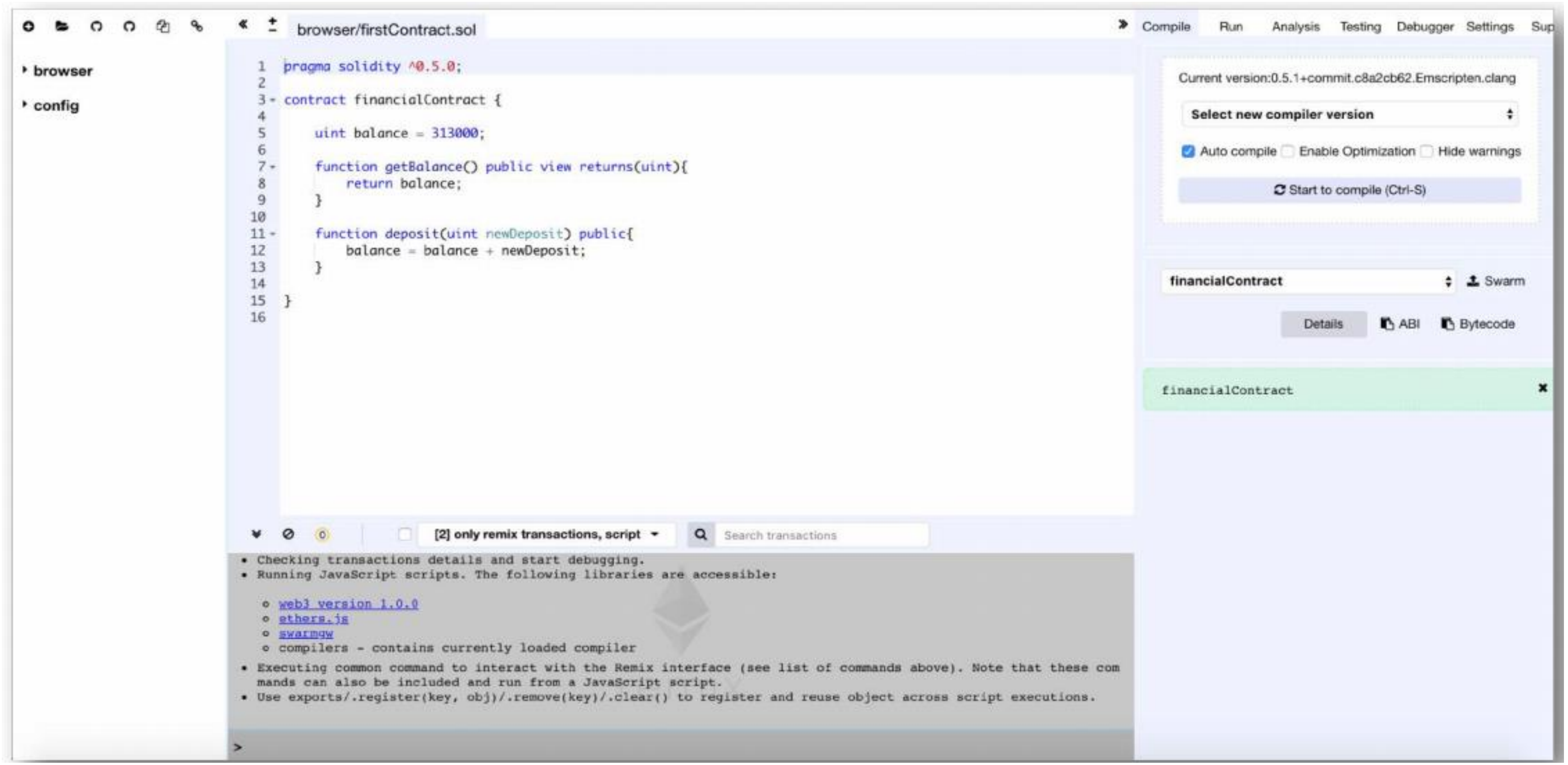


Smart Contracts

1. Developing a simple contract
2. Compiling the contract
3. Deploying the contract
4. Interacting with the contract
5. Adding more functions to our code to make it more practical

Open Remix : remix.ethereum.org

- An open source tool for writing, compiling and testing Solidity contracts



```
pragma solidity ^0.5.0;

contract financialContract {

    uint balance = 313000;

    function getBalance() public view returns(uint){
        return balance;
    }

    function deposit(uint newDeposit) public{
        balance = balance + newDeposit;
    }

}
```

Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

```
• Checking transactions details and start debugging.
• Running JavaScript scripts. The following libraries are accessible:
    ○ web3 version 1.0.0
    ○ ethers.js
    ○ swarmgw
    ○ compilers - contains currently loaded compiler
• Executing common command to interact with the Remix interface (see list of commands above). Note that these com
  mands can also be included and run from a JavaScript script.
• Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.
>
```

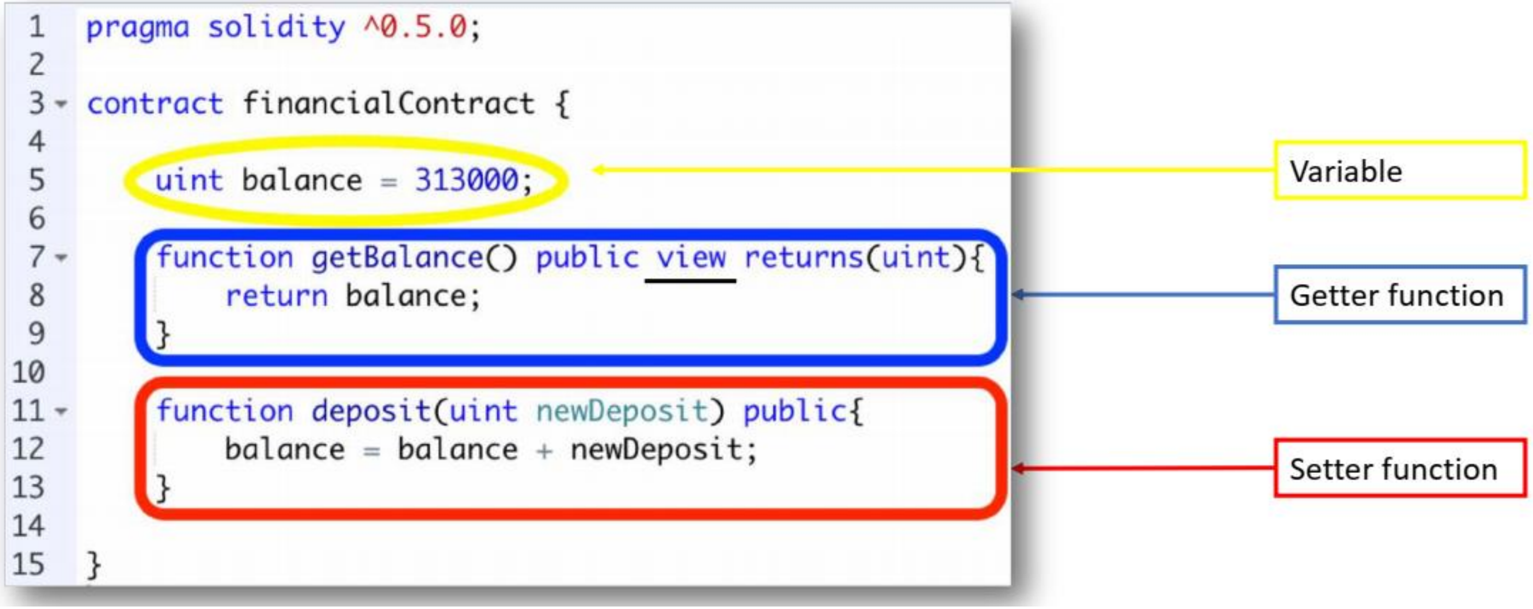
Solidity

- Object-oriented
- Contract-oriented
- High-level language
- Influenced by C++, Python, and JavaScript
- Target Ethereum Virtual Machine (EVM)

Serpent as an Alternative?
- Low-level language
- Complex compiler

Start Coding

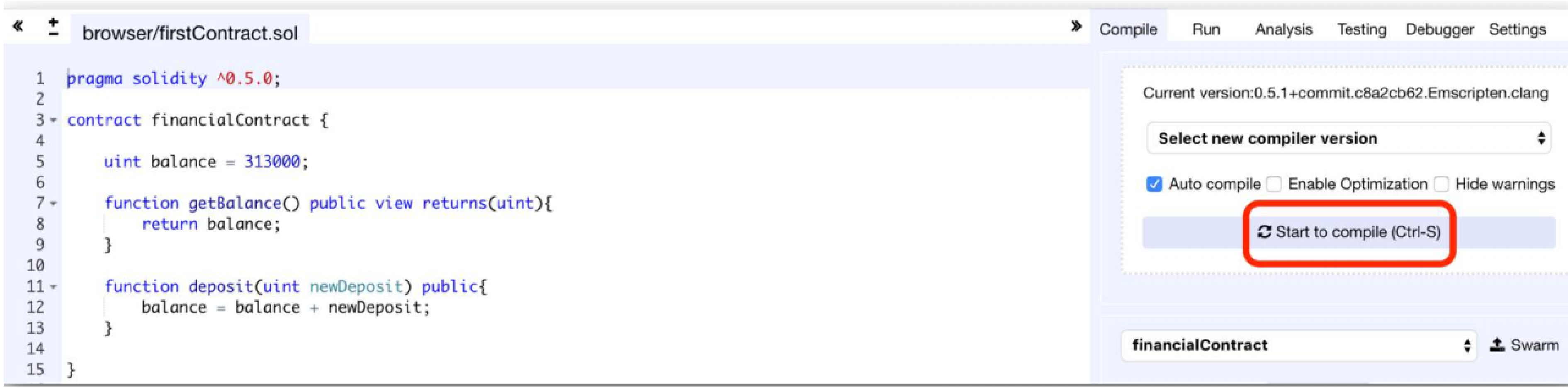
- Setter and Getter: Set and get the information



```
pragma solidity ^0.5.0;

contract financialContract {

    uint balance = 313000;

    function getBalance() public view returns(uint){
        return balance;
    }

    function deposit(uint newDeposit) public{
        balance = balance + newDeposit;
    }

}
```

Compile the Contract

- Compile tab: Start to compile button


browser/firstContract.sol
Compile
Run
Analysis
Testing
Debugger
Settings
Select new compiler version
Start to compile (Ctrl-S)
Swarm

```
pragma solidity ^0.5.0;

contract financialContract {

    uint balance = 313000;

    function getBalance() public view returns(uint){
        return balance;
    }

    function deposit(uint newDeposit) public{
        balance = balance + newDeposit;
    }

}
```

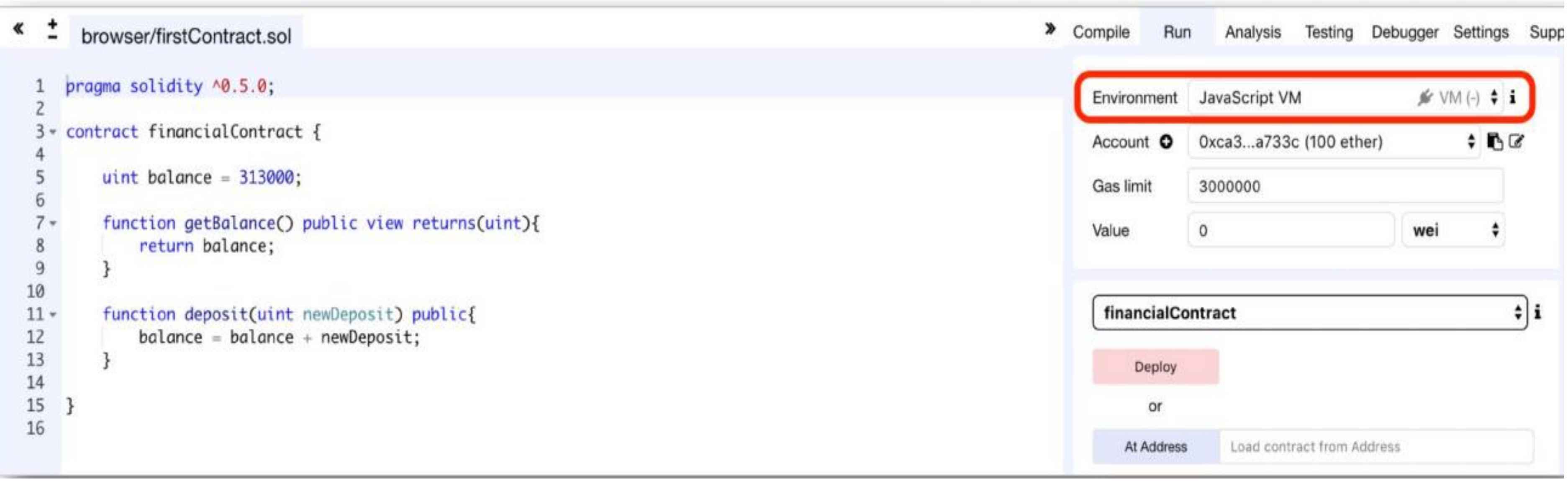
Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

[✓] Auto compile [ ] Enable Optimization [ ] Hide warnings

financialContract

Set Deployment Parameters

- Run tab: Environment = JavaScript VM



```
pragma solidity ^0.5.0;

contract financialContract {

    uint balance = 313000;

    function getBalance() public view returns(uint){
        return balance;
    }

    function deposit(uint newDeposit) public{
        balance = balance + newDeposit;
    }

}
```

Environment: JavaScript VM — VM (-)
Account: 0xca3...a733c (100 ether)
Gas limit: 3000000
Value: 0 wei
financialContract

Set Deployment Parameters

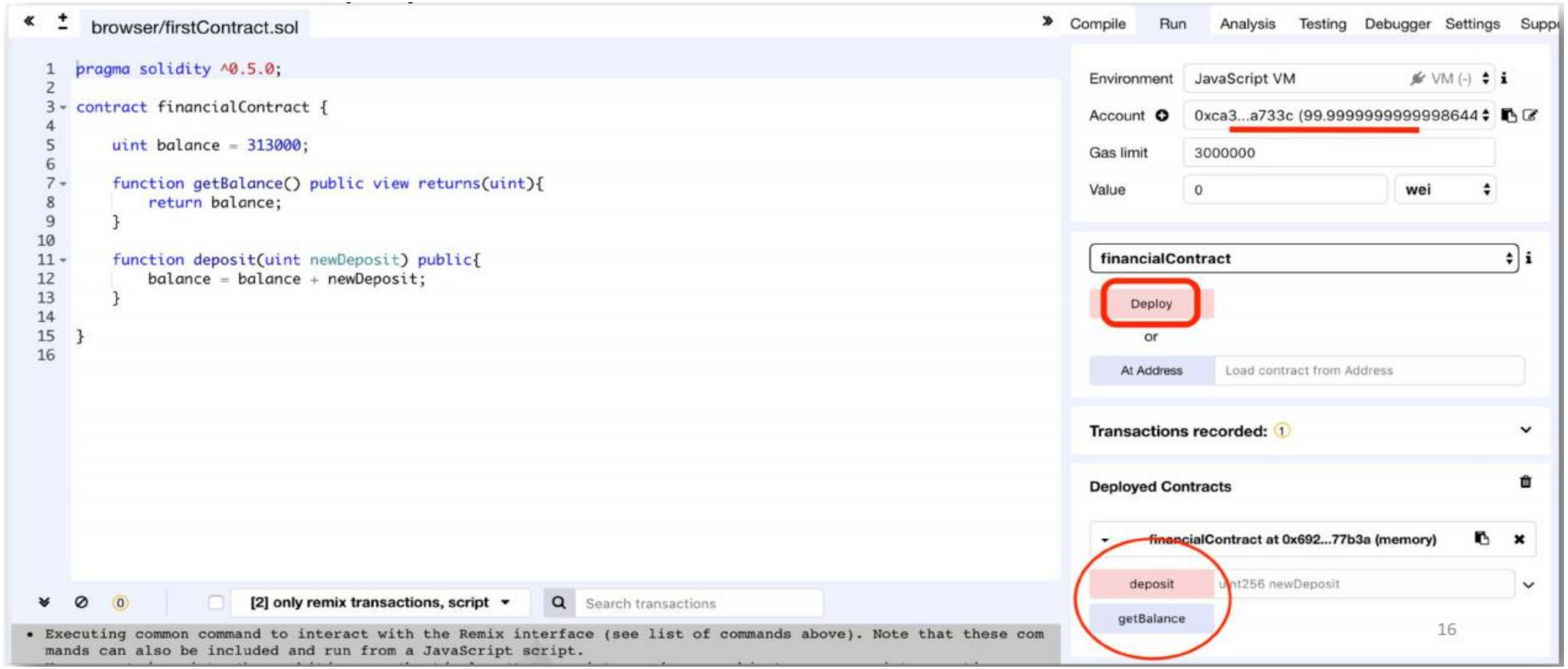
- **JavaScript VM:** All the transactions will be executed in a sandbox blockchain in the browser. Nothing will be persisted and a page reload will restart a new blockchain from scratch, the old one will not be saved.
- **Injected Provider:** Remix will connect to an injected web3 provider. Mist and Metamask are example of providers that inject web3, thus they can be used with this option.
- **Web3 Provider:** Remix will connect to a remote node. You will need to provide the URL address to the selected provider: geth, parity or any
- **Gas Limit:** The maximum amount of gas that can be set for all the instructions of a contract.
- **Value:** Input some ether with the next created transaction (wei = 10-18 of ether).

Types of Blockchain Deployment

- **Private: e.g.**, Ganache sets a personal Ethereum blockchain for running tests, executing commands, and inspecting the state while controlling how the chain operates.
- **Public Test (Testnet):** Like Ropsten, Kovan and Rinkeby which are existing public blockchains used for testing and which do not use real funds. Use faucet for receiving initial virtual funds.
- **Public Real (Mainnet):** Like Bitcoin and Ethereum which are used for real and which available for everybody to join.

Deploy the Contract on the Private Blockchain of Remix

- Run tab: Deploy button



```
pragma solidity ^0.5.0;

contract financialContract {

    uint balance = 313000;

    function getBalance() public view returns(uint){
        return balance;
    }

    function deposit(uint newDeposit) public{
        balance = balance + newDeposit;
    }

}
```

Environment: JavaScript VM

Account: 0xca3...a733c (99.9999999999998644

Gas limit: 3000000

Value: 0 wei

financialContract

Transactions recorded: 1

Deployed Contracts

financialContract at 0x692...77b3a (memory)

deposit: uint256 newDeposit

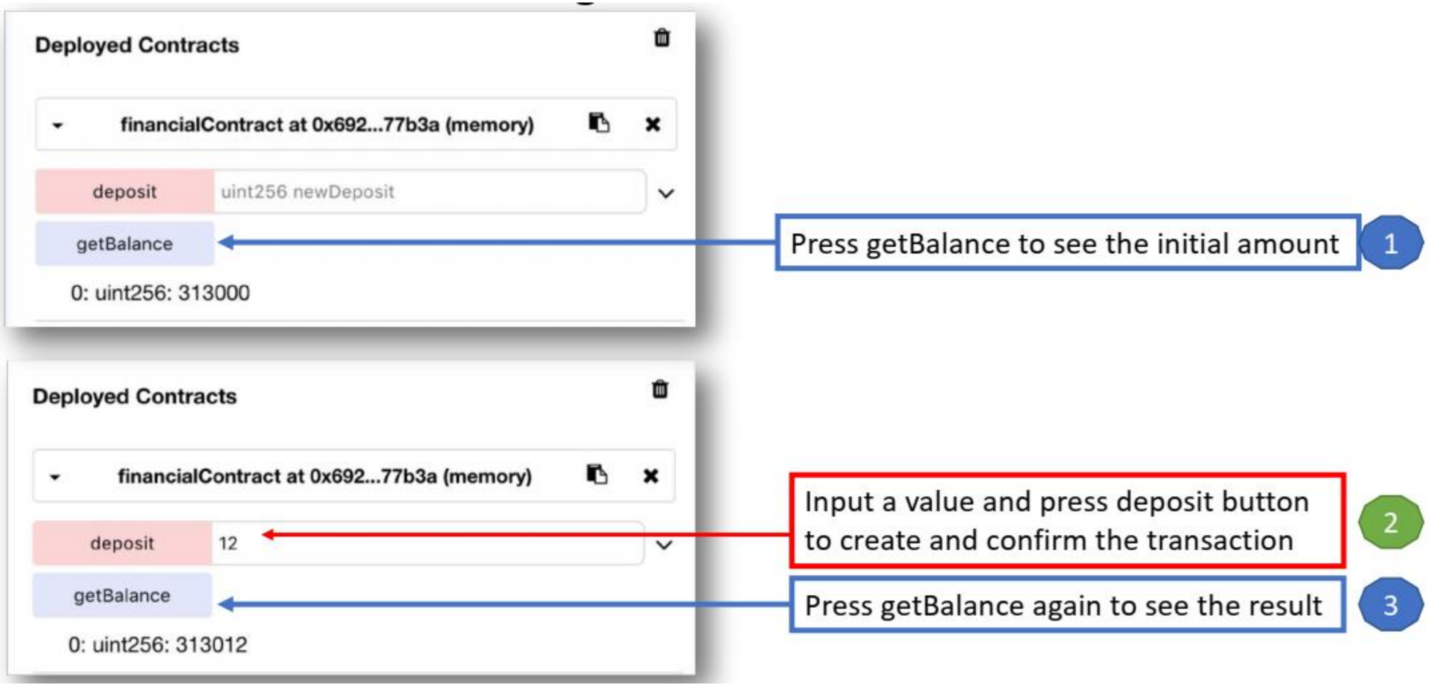
[2] only remix transactions, script

Search transactions

```
• Executing common command to interact with the Remix interface (see list of commands above). Note that these com
  mands can also be included and run from a JavaScript script.
```

Interact with the Contract

- Setter = Red Button: Creates transaction
- Getter= Blue Button: Just gives information



Deployed Contracts

financialContract at 0x692...77b3a (memory)

```
0: uint256: 313000
```

Press getBalance to see the initial amount

Deployed Contracts

financialContract at 0x692...77b3a (memory)

```
0: uint256: 313012
```

Input a value and press deposit button to create and confirm the transaction

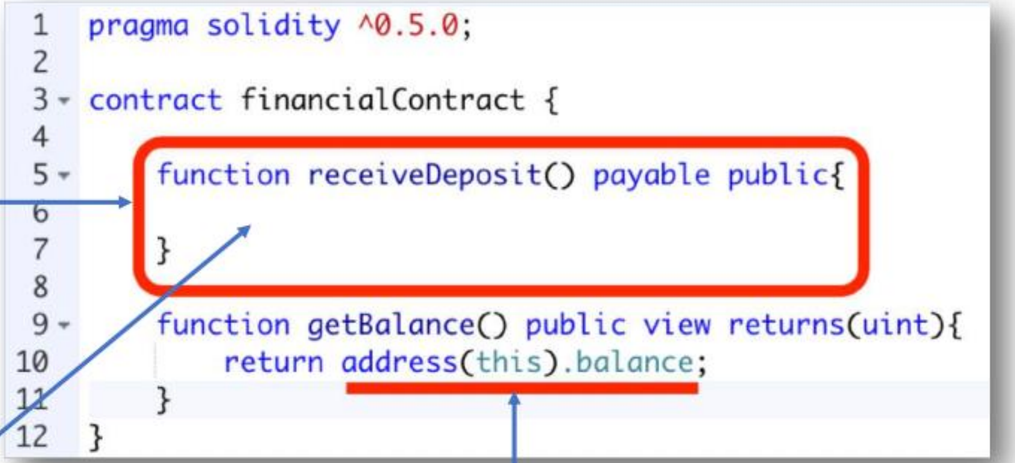
Press getBalance again to see the result

Additional features

- Transferring funds from an account to the contract
- Saving the address of the contract creator
- Limiting the users’ access to functions
- Withdrawing funds from the contract to an account

Receive ether

- Transfer money to the contract



```
pragma solidity ^0.5.0;

contract financialContract {

    function receiveDeposit() payable public{

    }

    function getBalance() public view returns(uint){
        return address(this).balance;
    }
}
```

Payable keyword allows receiving ether

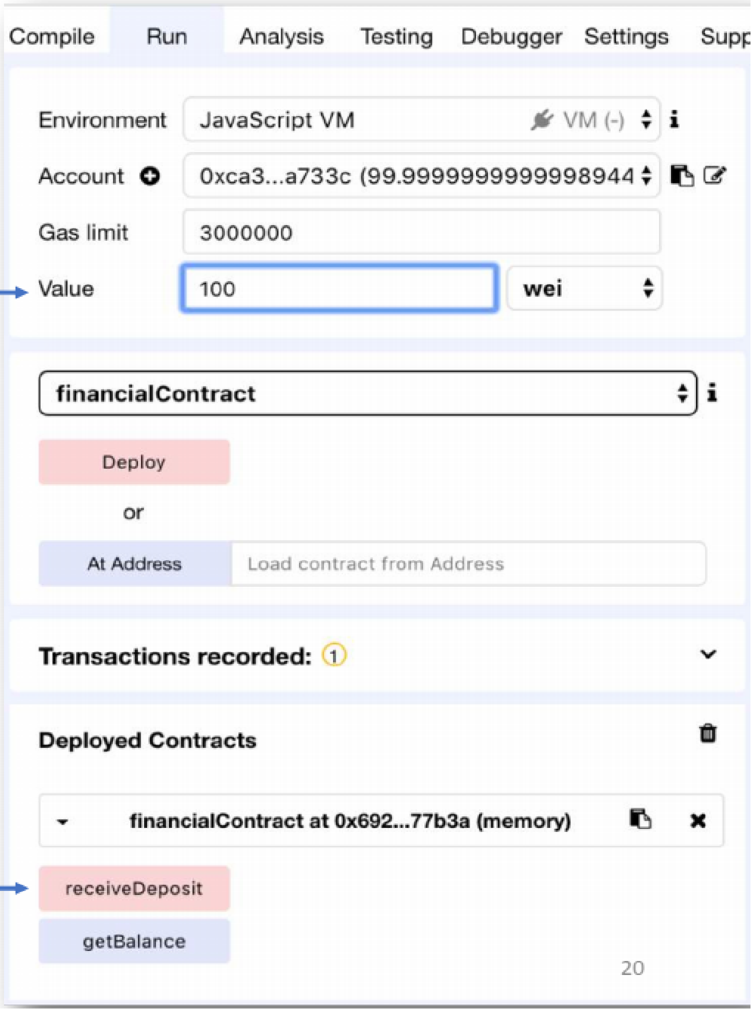
Hidden Code:
Address(this).balance += msg.value;

We can get the balance of the contract

Receive ether (2/2)

1 Input the value as wei (10^{-18} of ether)

2 Click the receiveDeposit button to transfer the money to the contract



Environment JavaScript VM VM (-)

Account 0xca3...a733c (99.9999999999998944

Gas limit 3000000

Value 100 wei

financialContract

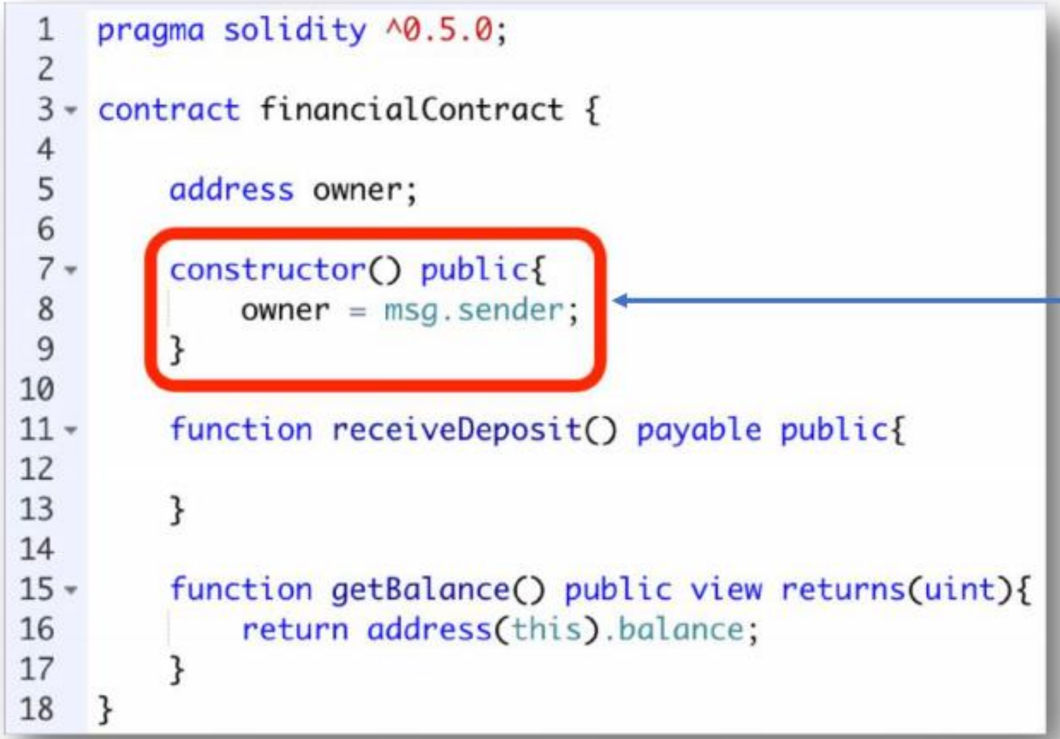
Transactions recorded: 1

Deployed Contracts

financialContract at 0x692...77b3a (memory)

Constructor

- Will be called at the creation of the instance of the contract

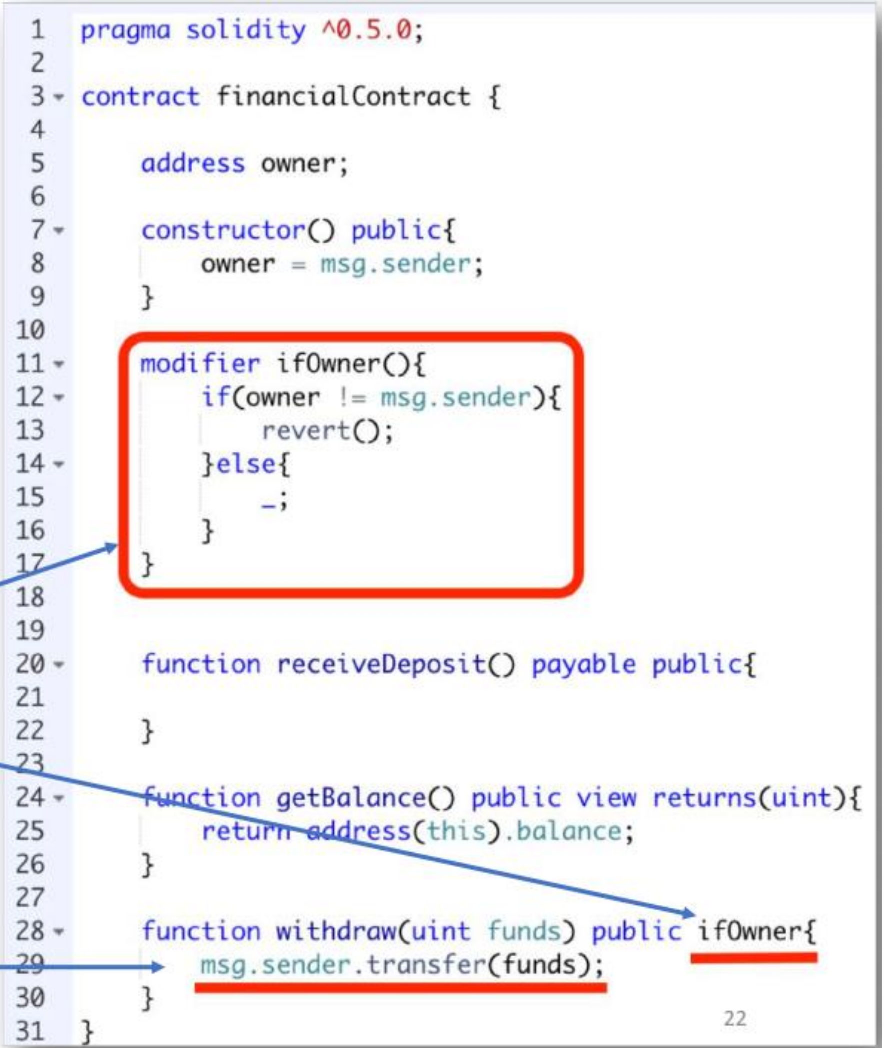


```
pragma solidity ^0.5.0;

contract financialContract {

    address owner;

    constructor() public{
        owner = msg.sender;
    }

    function receiveDeposit() payable public{

    }

    function getBalance() public view returns(uint){
        return address(this).balance;
    }
}
```

We want to save the address of the contract creator

Withdraw funds

- Modifier: Conditions you want to test in other functions
- First the modifier will execute, then the invoked function



```
pragma solidity ^0.5.0;

contract financialContract {

    address owner;

    constructor() public{
        owner = msg.sender;
    }

    modifier ifOwner(){
        if(owner != msg.sender){
            revert();
        }else{
            _;
        }
    }


    function receiveDeposit() payable public{

    }

    function getBalance() public view returns(uint){
        return address(this).balance;
    }

    function withdraw(uint funds) public ifOwner{
        msg.sender.transfer(funds);
    }
}
```

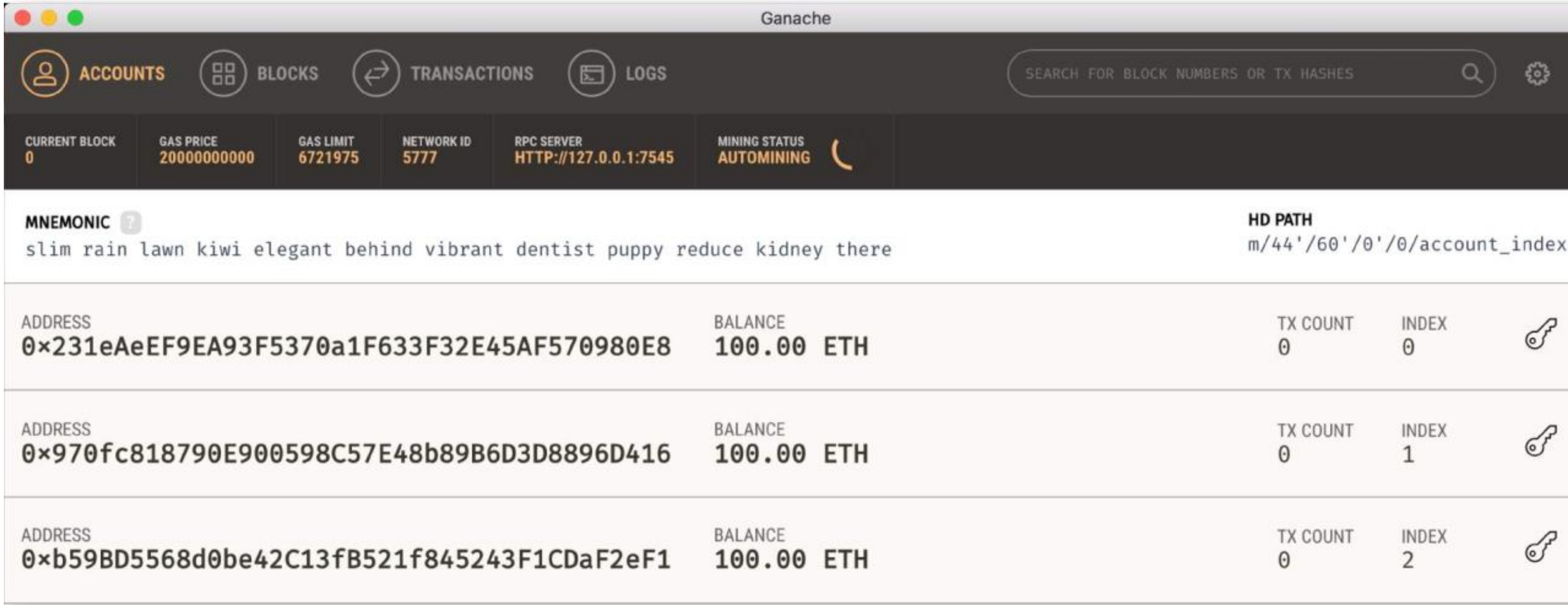
Only the contract's creator is permitted to withdraw

Transfer some money from the contract's balance to the owner

Now deploying a smart contract on an external blockchain

	Tools / Activities	Remix	Ganache	MyEtherWallet	Geth
1	Configuring the Blockchain	-	-	-	+
2	Deploying the Blockchain	Not Persistent	+	-	+
3	Developing the contract	+	-	-	+
4	Compiling the contract	+	-	-	+
5	Creating user account	+	+	+	+
6	Deploying the contract	+	-	+	+
7	Creating the UI for interacting	+	-	+	+
8	Run the client	+	-	+	+
9	Interact with the contract & have fun	+	-	+	+
10	Monitoring the execution	-	+	-	+

Run Ganache



CURRENT BLOCK 0
GAS PRICE 20000000000
GAS LIMIT 6721975
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING

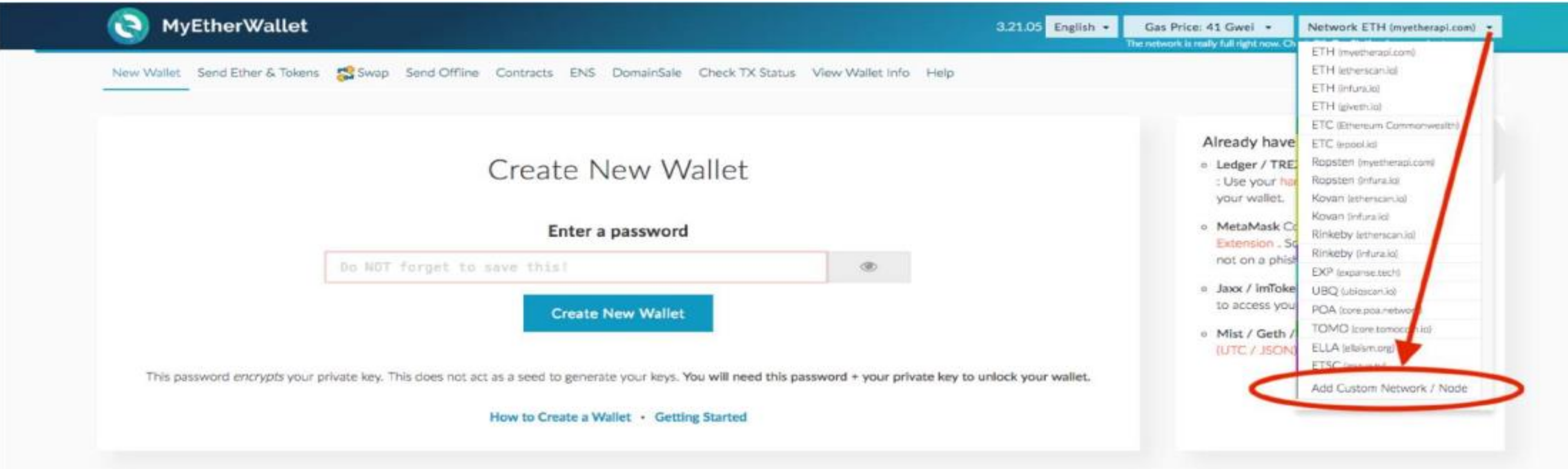
MNEMONIC
slim rain lawn kiwi elegant behind vibrant dentist puppy reduce kidney there

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0x231eAeEF9EA93F5370a1F633F32E45AF570980E8	100.00 ETH	0	0
0x970fc818790E900598C57E48b89B6D3D8896D416	100.00 ETH	0	1
0xb59BD5568d0be42C13fB521f845243F1CDaF2eF1	100.00 ETH	0	2

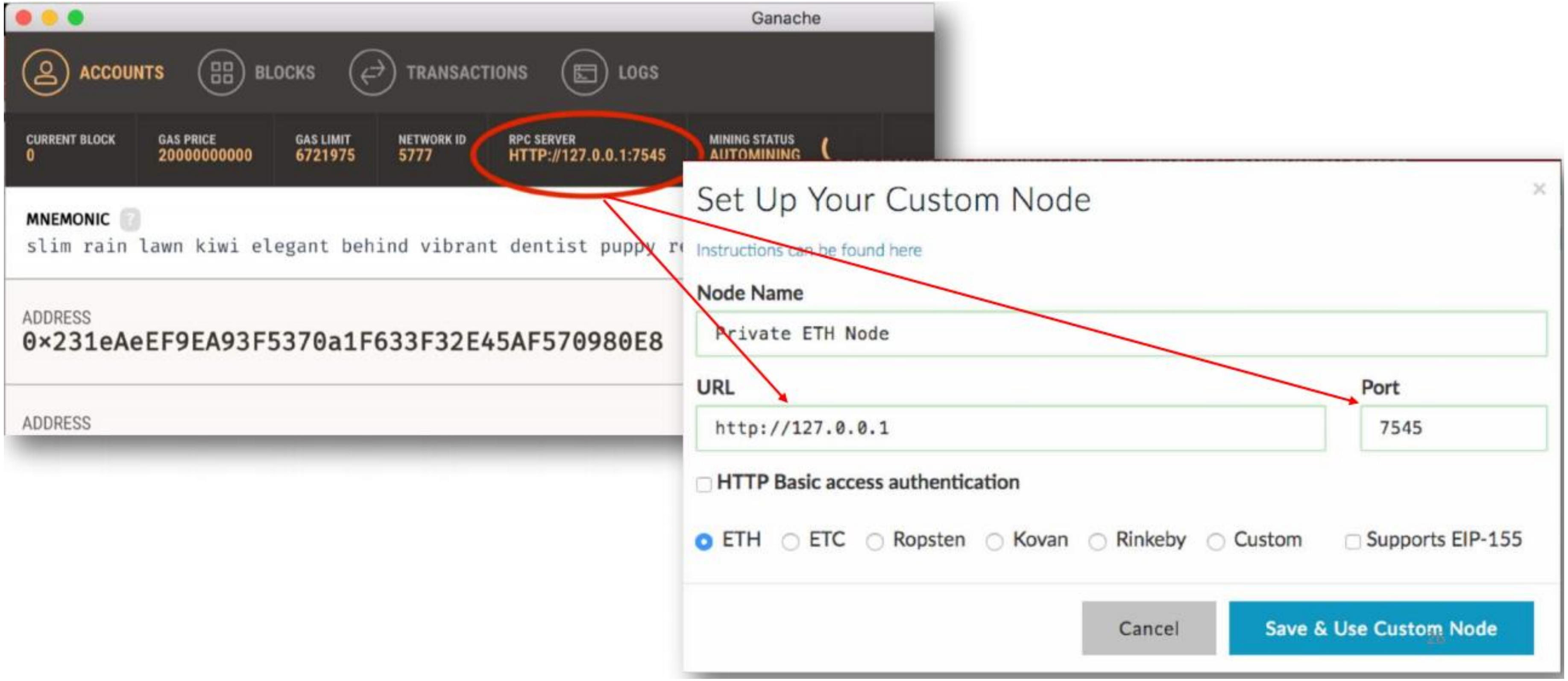
MyEtherWallet

- add your custom network that you want to test your contracts on



This password *encrypts* your private key. This does not act as a seed to generate your keys. **You will need this password + your private key to unlock your wallet.**

Import your RPC server address and the port number from Ganache to MyEtherWallet



Ganache

CURRENT BLOCK 0
GAS PRICE 20000000000
GAS LIMIT 6721975
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING

MNEMONIC
slim rain lawn kiwi elegant behind vibrant dentist puppy r

ADDRESS
0x231eAeEF9EA93F5370a1F633F32E45AF570980E8

ADDRESS

Set Up Your Custom Node

Instructions can be found here

Node Name: Private ETH Node

URL: http://127.0.0.1

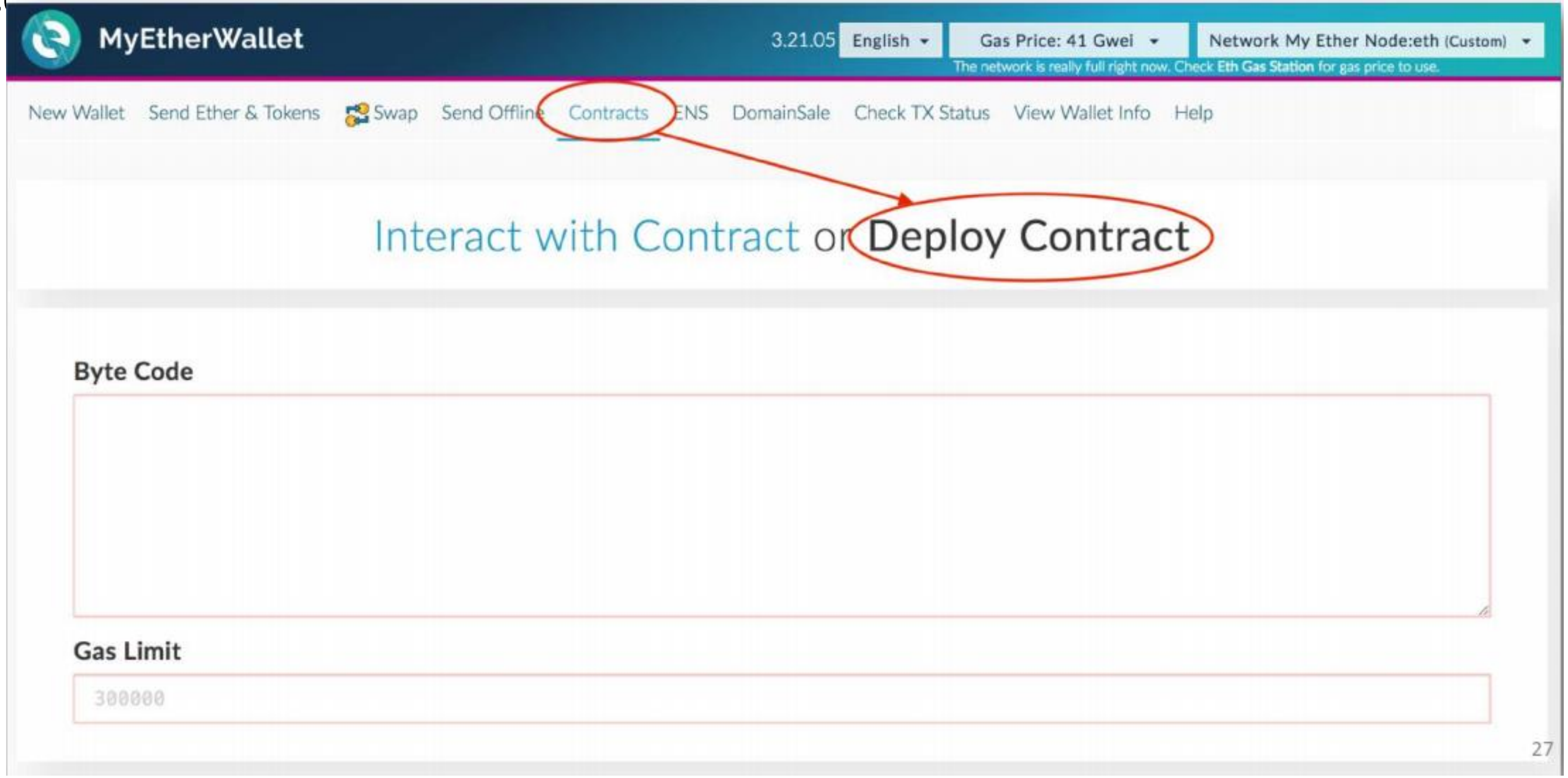
Port: 7545

☐ HTTP Basic access authentication

[✓ ETH] ETC Ropsten Kovan Rinkeby Custom ☐ Supports EIP-155

MyEtherWallet

- Contracts tab: Deploy Contract
-



MyEtherWallet 3.21.05 English Gas Price: 41 Gwei Network My Ether Node:eth (Custom)

The network is really full right now. Check Eth Gas Station for gas price to use.

New Wallet Send Ether & Tokens Swap Send Offline Contracts ENS DomainSale Check TX Status View Wallet Info Help

Interact with Contract or Deploy Contract

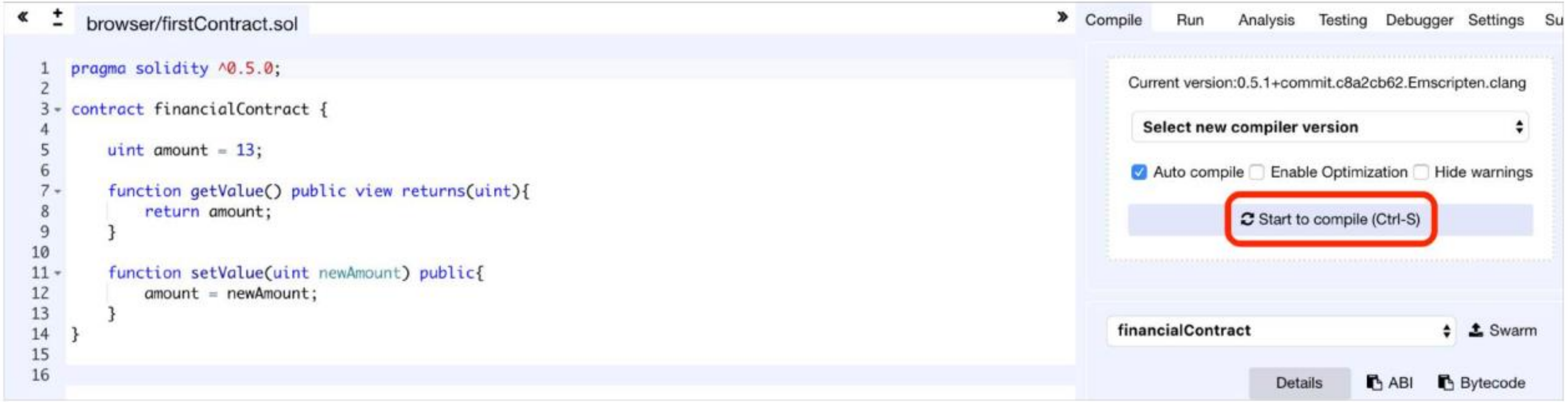
Byte Code

Gas Limit

300000

Remix

- Type your contract and compile it



browser/firstContract.sol

```
pragma solidity ^0.5.0;

contract financialContract {

    uint amount = 13;

    function getValue() public view returns(uint){
        return amount;
    }

    function setValue(uint newAmount) public{
        amount = newAmount;
    }
}
```

Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

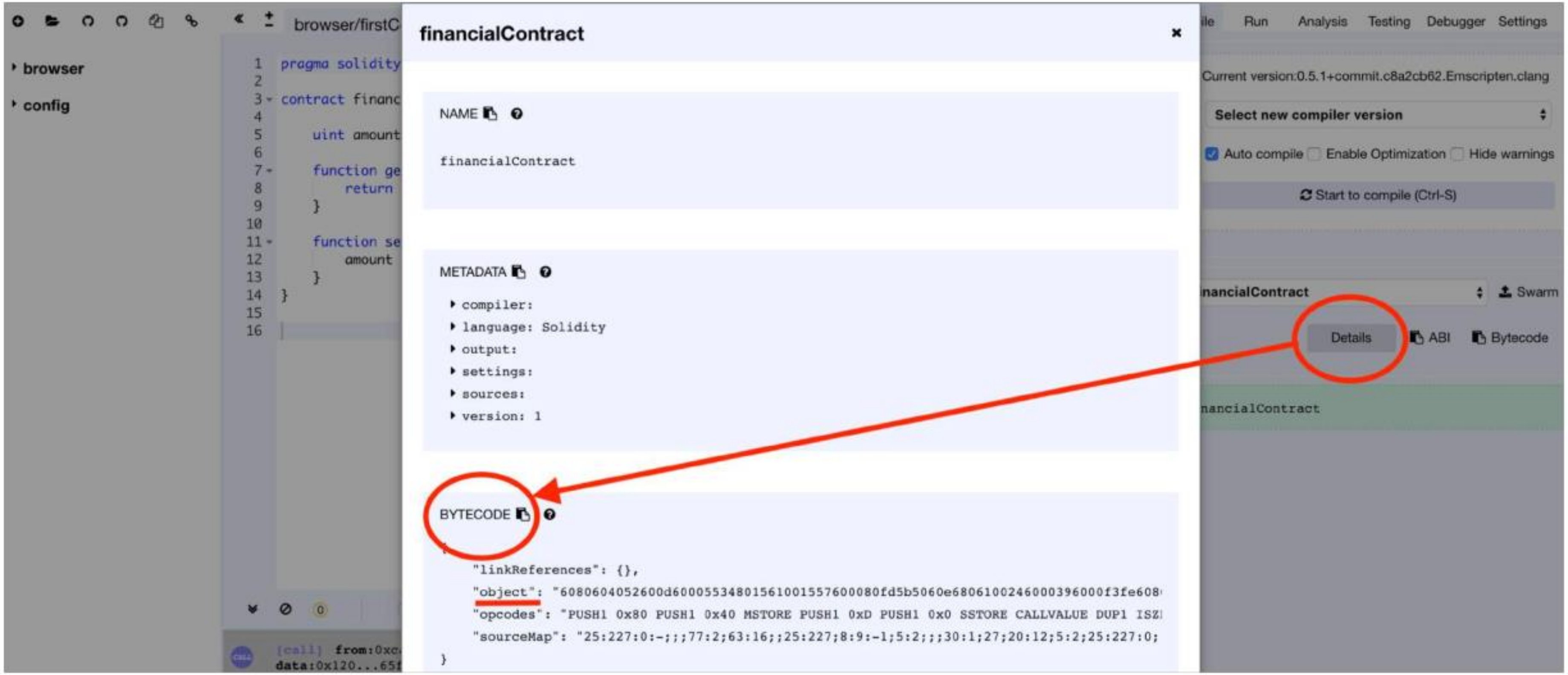
Select new compiler version

[✓ Auto compile] Enable Optimization Hide warnings

financialContract

Remix

- Click on Details Button: access ByteCode to import it to MyEtherWallet



```
pragma solidity
contract financ
    uint amount
    function ge
        return
    }
    function se
        amount
    }
}
```

Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

NAME: financialContract

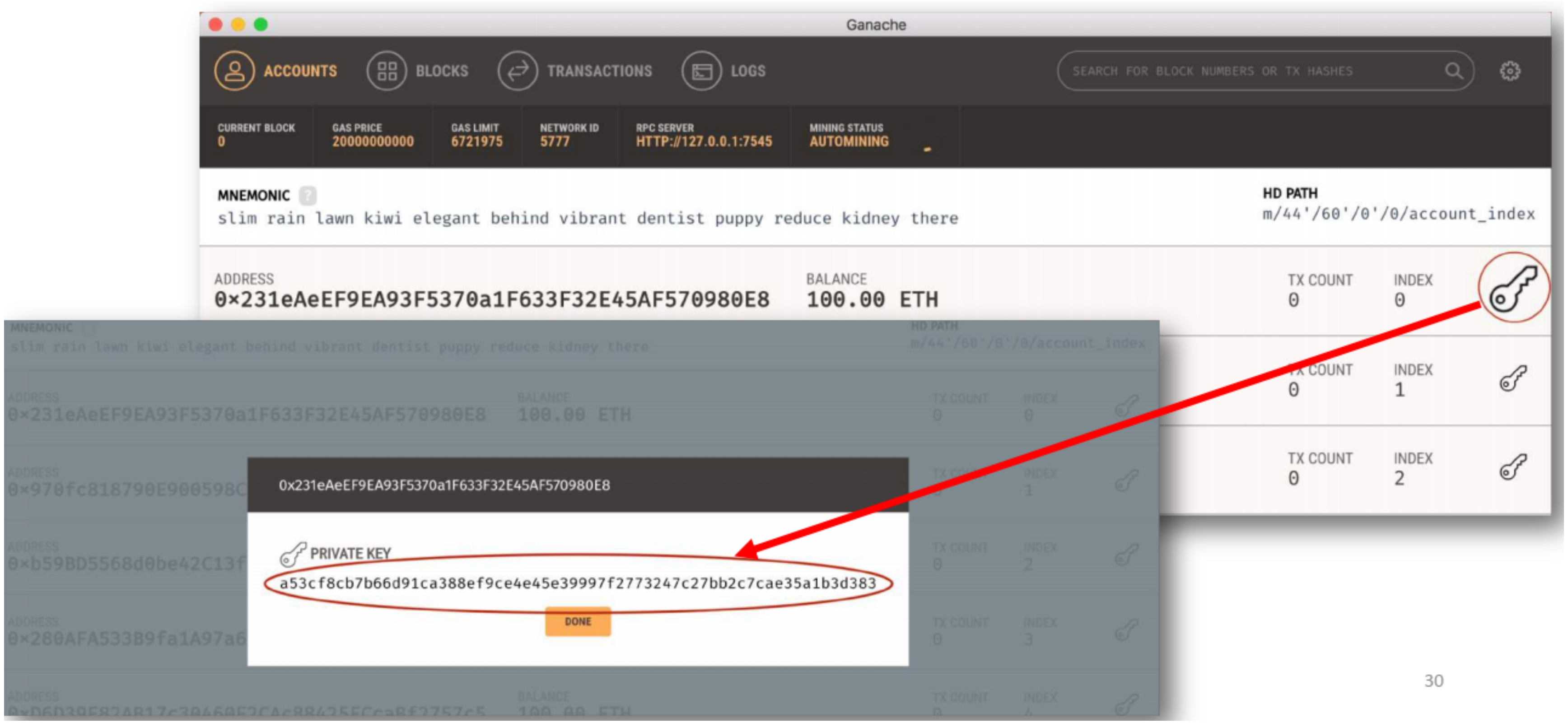
METADATA:
- compiler:
- language: Solidity
- output:
- settings:
- sources:
- version: 1

BYTECODE:

```
{
    "linkReferences": {},
    "object": "6080604052600d600055348015610015576000080fd5b5060e68061002460003960000f3fe608
    "opcodes": "PUSH1 0x80 PUSH1 0x40 MSTORE PUSH1 0xD PUSH1 0x0 SSTORE CALLVALUE DUP1 ISZ
    "sourceMap": "25:227:0:-;;;77:2;63:16;;25:227;8:9:-1;5:2;;;30:1;27;20:12;5:2;25:227:0;
}
```

Ganache

- Access your private key for signing your contract in MyEtherWallet.



Ganache

CURRENT BLOCK 0 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING

MNEMONIC
slim rain lawn kiwi elegant behind vibrant dentist puppy reduce kidney there

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS 0x231eAeEF9EA93F5370a1F633F32E45AF570980E8 | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 0

TX COUNT 0 | INDEX 1

TX COUNT 0 | INDEX 2

0x231eAeEF9EA93F5370a1F633F32E45AF570980E8

PRIVATE KEY
a53cf8cb7b66d91ca388ef9ce4e45e39997f2773247c27bb2c7cae35a1b3d383

MyEtherWallet

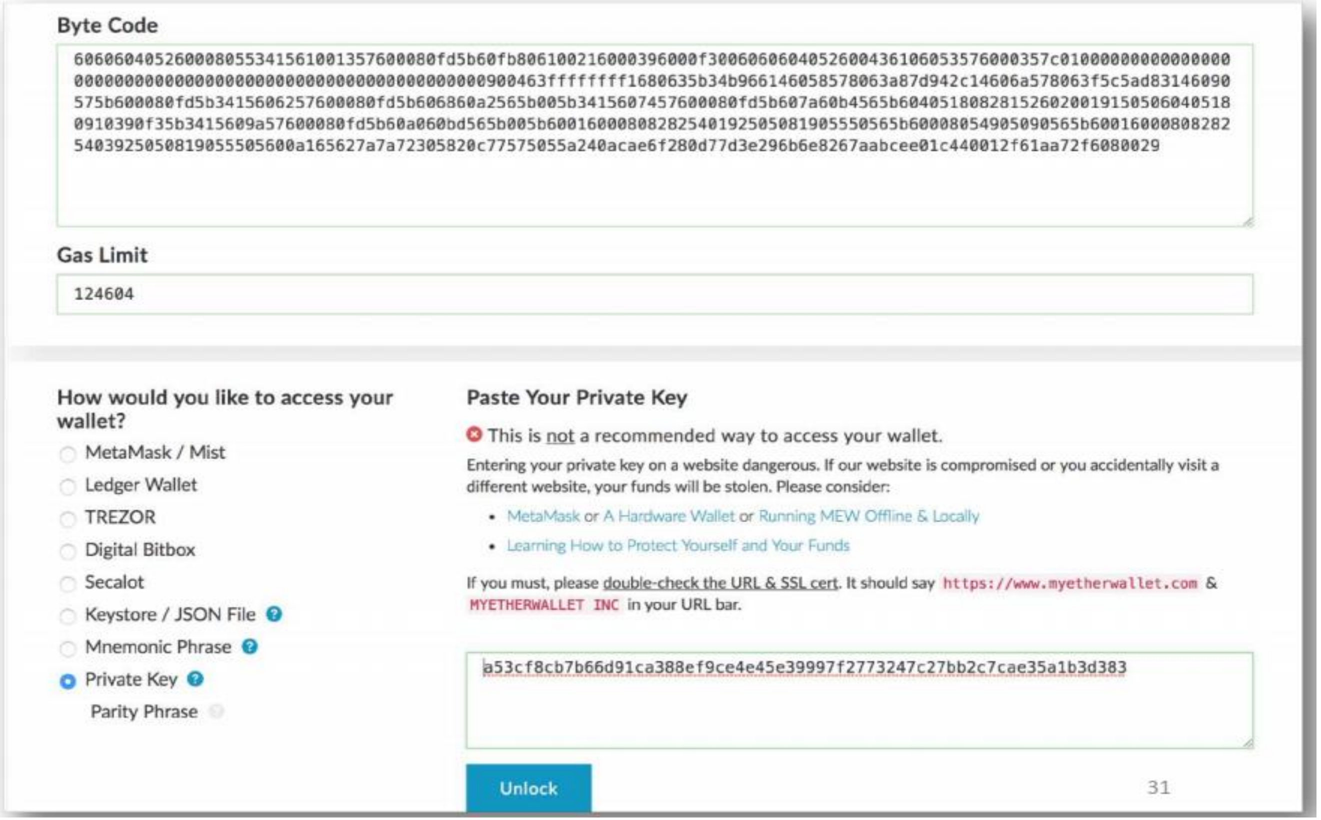
1. Paste the contract’s ByteCode from Remix

2. Gas Limit will automatically be calculated

3. Paste your private key from Ganache

4. Click Unlock

5. Now you have access to your wallet



Byte Code

```
6060604052600080553415610013576000080fd5b60fb806100216000396000f3006060604052600436106053576000357c0100000000000000
00000000000000000000000000000000000000000900463ffffffff1680635b34b966146058578063a87d942c14606a578063f5c5ad83146090
575b600080fd5b3415606257600080fd5b606860a2565b005b3415607457600080fd5b607a60b4565b6040518082815260200191505060405180
910390f35b3415609a57600080fd5b60a060bd565b005b600160008082825401925050819055505b565b6000805490509056 5b60016000808282
54039250508190555056 00a165627a7a72305820c77575055a240acae6f280d77d3e296b6e8267aabcee01c440012f61aa72f6080029
```

Gas Limit

```
124604
```

How would you like to access your wallet?

- MetaMask / Mist
- Ledger Wallet
- TREZOR
- Digital Bitbox
- Secalot
- Keystore / JSON File
- Mnemonic Phrase
- [✓ Private Key]
- Parity Phrase

Paste Your Private Key

This is not a recommended way to access your wallet.

Entering your private key on a website dangerous. If our website is compromised or you accidentally visit a different website, your funds will be stolen. Please consider:

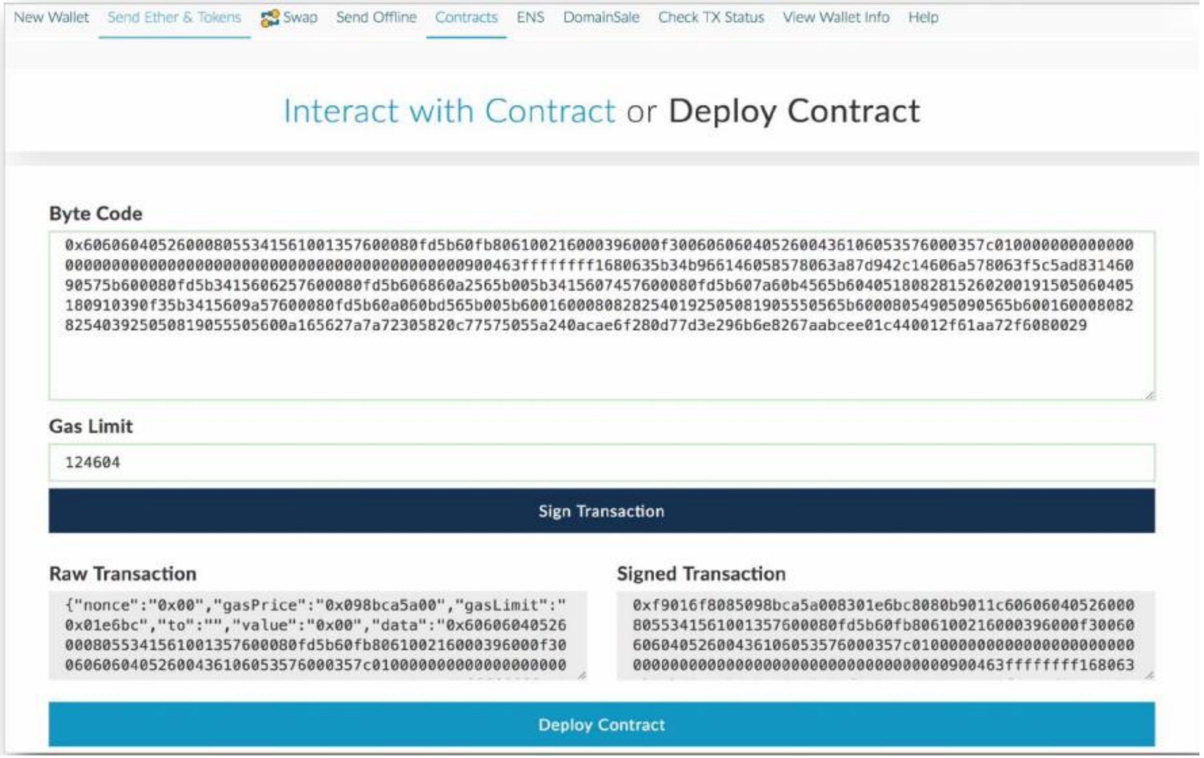
- MetaMask or A Hardware Wallet or Running MEW Offline & Locally
- Learning How to Protect Yourself and Your Funds

If you must, please double-check the URL & SSL cert. It should say `https://www.myetherwallet.com` & `MYETHERWALLET INC` in your URL bar.

```
a53cf8cb7b66d91ca388ef9ce4e45e39997f2773247c27bb2c7cae35a1b3d383
```

MyEtherWallet

- Click on *Sign Transaction* button to deploy your contract



Interact with Contract or Deploy Contract

Byte Code

```
0x6060604052600080553415610013576000080fd5b60fb80610021600039600 0f300606060405260043610605357600035 7c010000000000000000
```

Gas Limit

124604

Sign Transaction

Raw Transaction

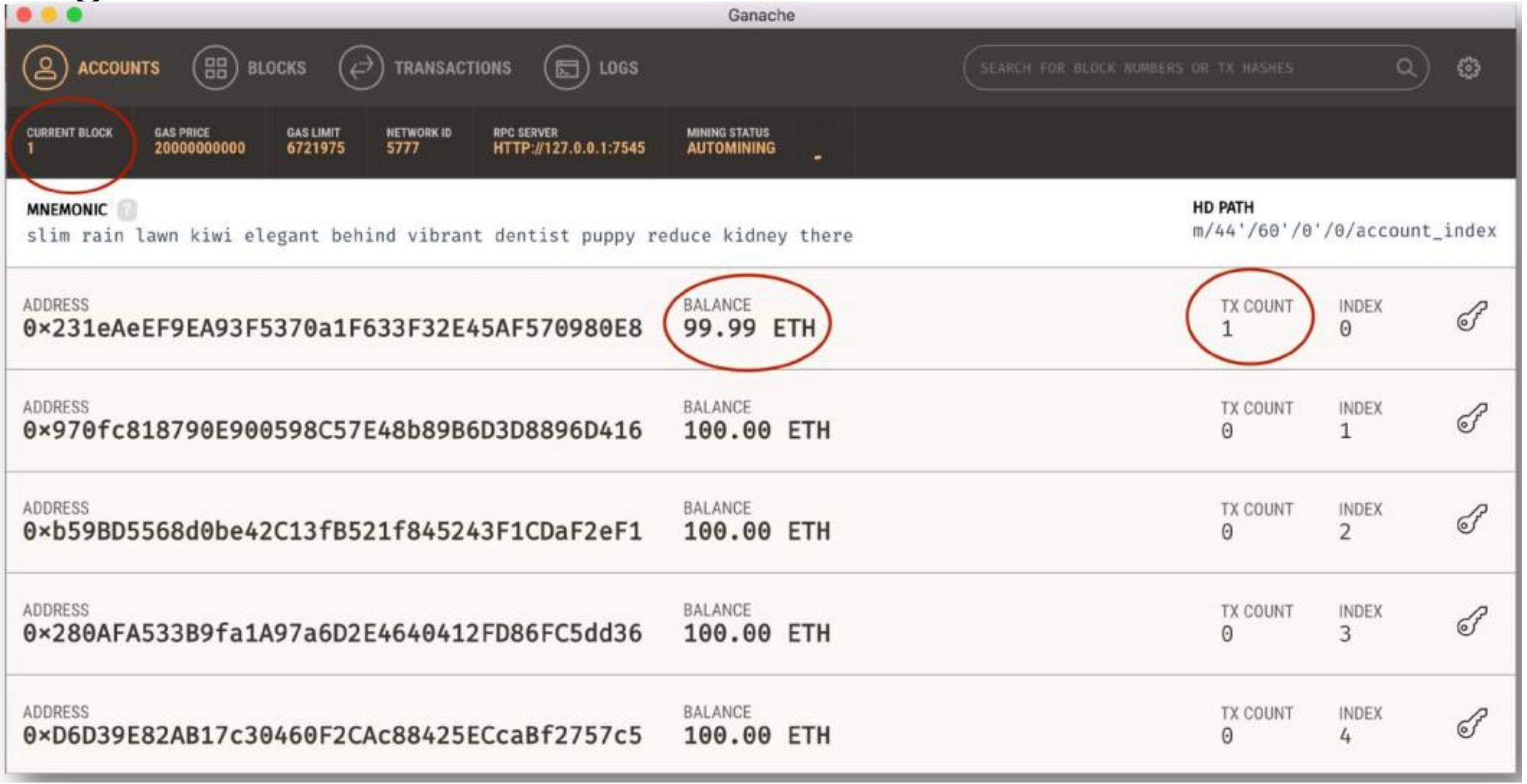
```
{"nonce":"0x00","gasPrice":"0x098bca5a00","gasLimit":"
0x01e6bc","to":"","value":"0x00","data":"0x60606040526
000805534156100135760 0080fd5b60fb806100216000396000f30
0606060405260043610605357600035 7c01000000000000000000
```

Signed Transaction

```
0xf9016f8085098bca5a008301e6bc8080b9011c60606040526000
80553415610013576000 80fd5b60fb806100216000396000f30060
60604052600436106053576000357c0100000000000000000000000
00000000000000000000000000000000000900463ffffffff168063
```

Deploy Contract

Ganache

- You can see now you have one transaction for your address and your balance has been changed because of the amount of gas you paid for creating the contract.
- 

Ganache

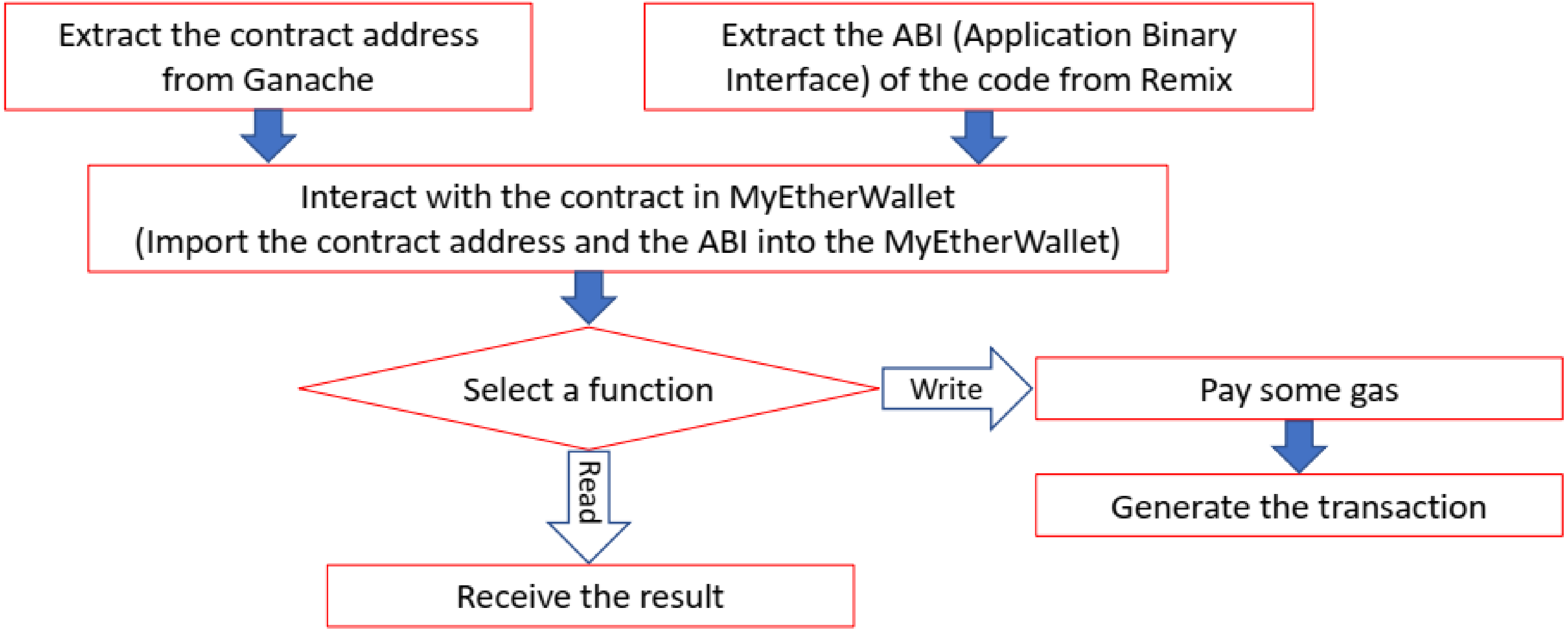
CURRENT BLOCK 1 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING

MNEMONIC
slim rain lawn kiwi elegant behind vibrant dentist puppy reduce kidney there

HD PATH
m/44'/60'/0'/0/account_index

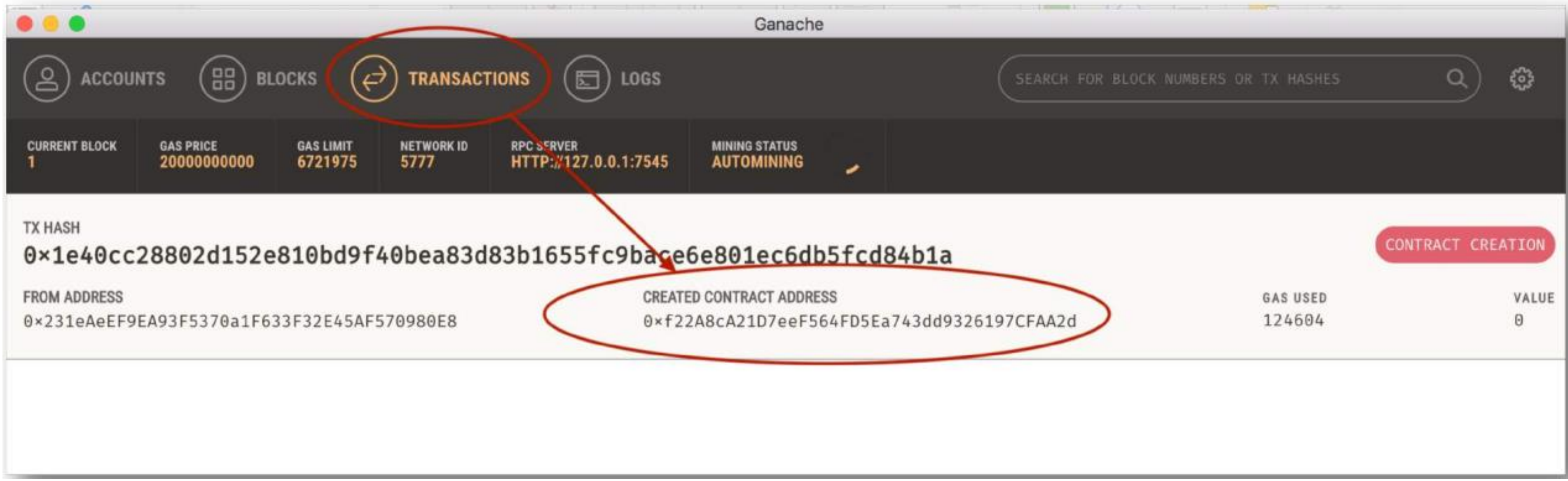
ADDRESS	BALANCE	TX COUNT	INDEX
0×231eAeEF9EA93F5370a1F633F32E45AF570980E8	99.99 ETH	1	0
0×970fc818790E900598C57E48b89B6D3D8896D416	100.00 ETH	0	1
0×b59BD5568d0be42C13fB521f845243F1CDaF2eF1	100.00 ETH	0	2
0×280AFA533B9fa1A97a6D2E4640412FD86FC5dd36	100.00 ETH	0	3
0×D6D39E82AB17c30460F2CAc88425ECcaBf2757c5	100.00 ETH	0	4

Interacting with the smart contract


Extract the contract address from Ganache
Extract the ABI (Application Binary Interface) of the code from Remix
Interact with the contract in MyEtherWallet (Import the contract address and the ABI into the MyEtherWallet)
Select a function
Write
Pay some gas
Generate the transaction
Read
Receive the result

Ganache

- Transactions tab: Copy the created contract address



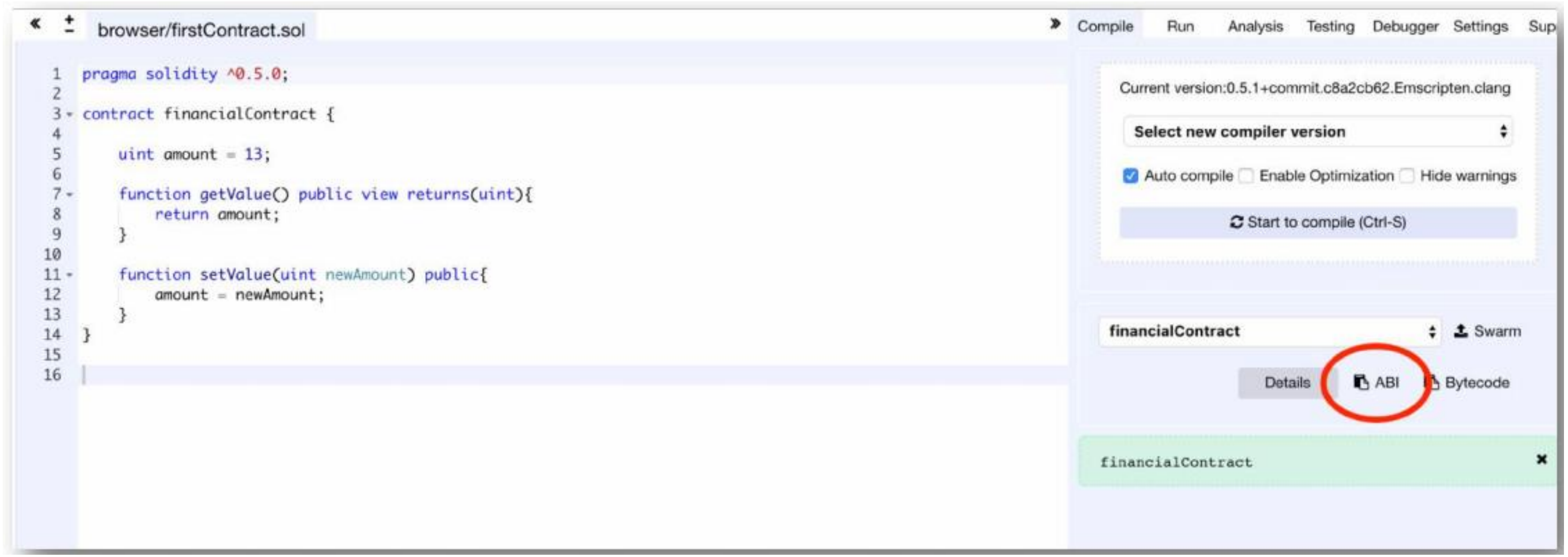
CURRENT BLOCK 1
GAS PRICE 20000000000
GAS LIMIT 6721975
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING

TX HASH
0×1e40cc28802d152e810bd9f40bea83d83b1655fc9bace6e801ec6db5fcd84b1a

FROM ADDRESS 0×231eAeEF9EA93F5370a1F633F32E45AF570980E8
CREATED CONTRACT ADDRESS 0×f22A8cA21D7eeF564FD5Ea743dd9326197CFAA2d
GAS USED 124604
VALUE 0

Remix

- Copy the ABI (ABI is the interface that tells MyEtherWallet how to interact with the contract)


browser/firstContract.sol
Compile
Run
Analysis
Testing
Debugger
Settings
Sup
Select new compiler version
Start to compile (Ctrl-S)
Swarm
Details
ABI
Bytecode

```
pragma solidity ^0.5.0;

contract financialContract {

    uint amount = 13;

    function getValue() public view returns(uint){
        return amount;
    }

    function setValue(uint newAmount) public{
        amount = newAmount;
    }
}

```

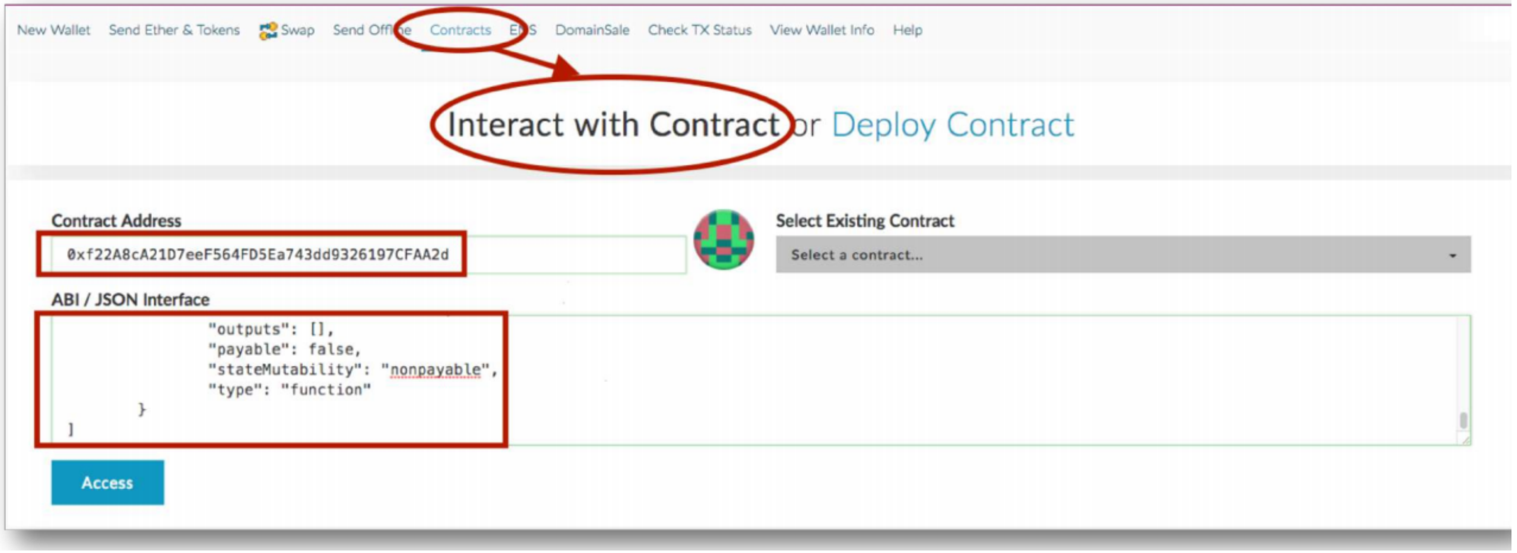
Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

[✓] Auto compile [ ] Enable Optimization [ ] Hide warnings

financialContract

financialContract

MyEtherWallet

- Contracts tab: Interact with Contract = Paste the contract address from Ganache and the ABI from Remix
- 

Contract Address

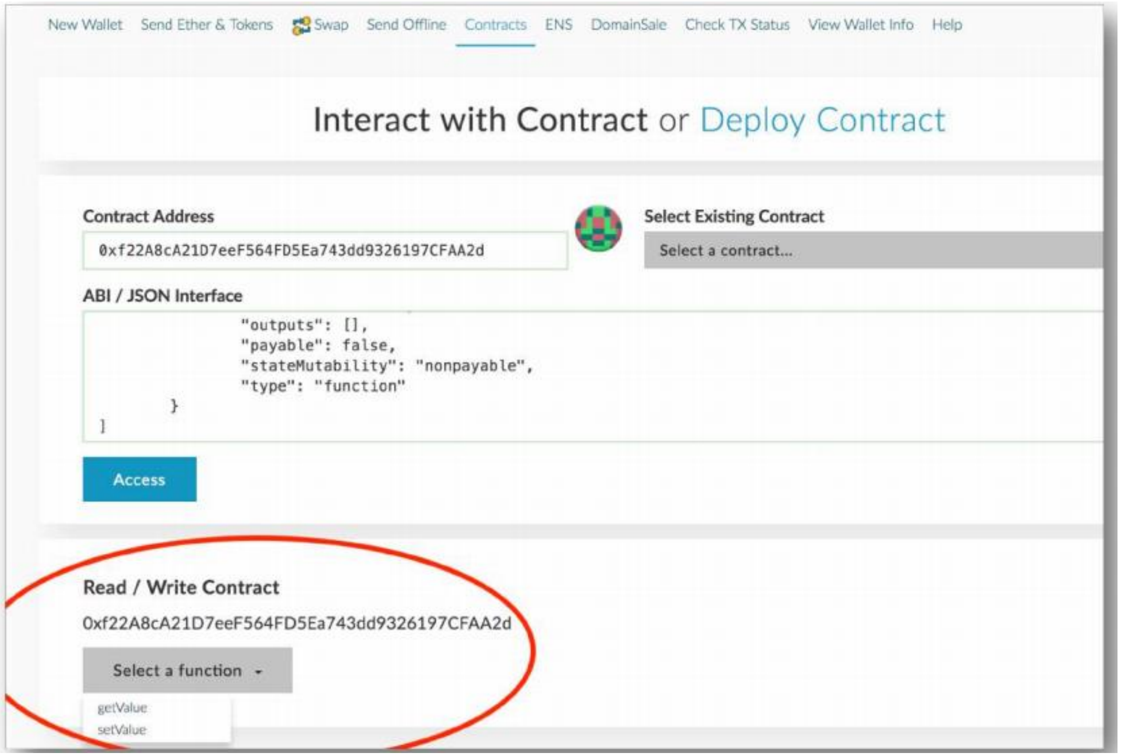
```
0xf22A8cA21D7eeF564FD5Ea743dd9326197CFAA2d
```

ABI / JSON Interface

```
                        "outputs": [],
                        "payable": false,
                        "stateMutability": "nonpayable",
                        "type": "function"
                }
]
```

MyEtherWallet

- You now can interact with the contract by selecting a function and invoking it



Interact with Contract or Deploy Contract

Contract Address

0xf22A8cA21D7eeF564FD5Ea743dd9326197CFAA2d

Select Existing Contract

Select a contract...

ABI / JSON Interface

```
                "outputs": [],
                "payable": false,
                "stateMutability": "nonpayable",
                "type": "function"
        }
]
```

Access

Read / Write Contract

0xf22A8cA21D7eeF564FD5Ea743dd9326197CFAA2d

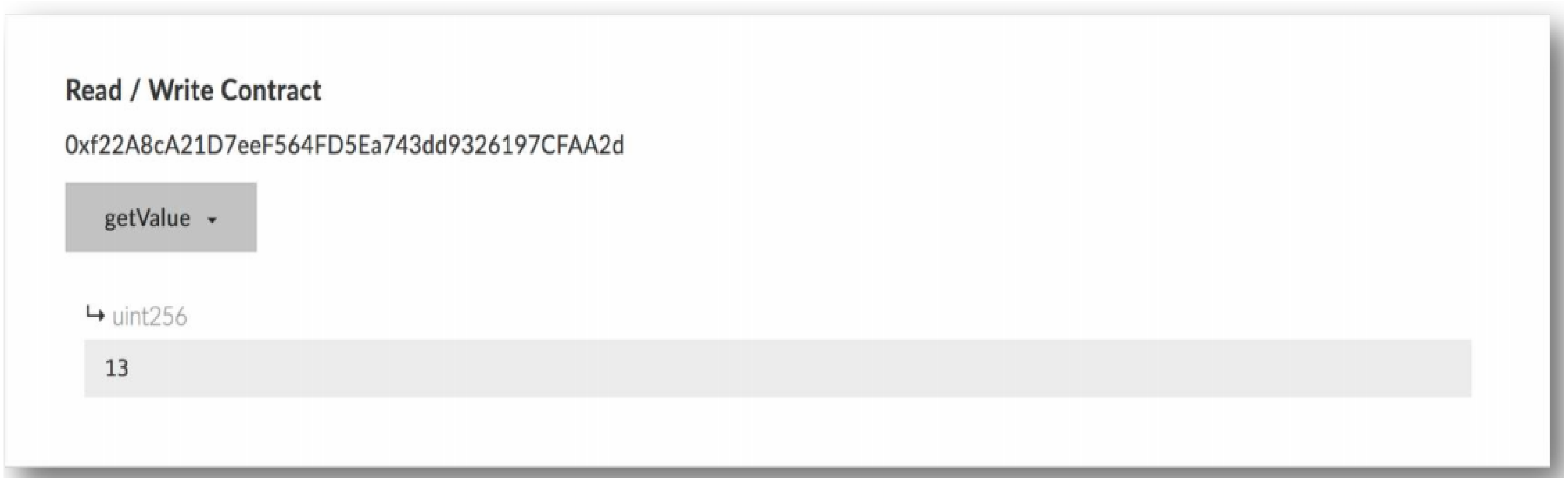
Select a function

getValue

setValue

MyEtherWallet

- If you select the getValue function you will receive the value without paying any gas (There is no operation cost for getting information)



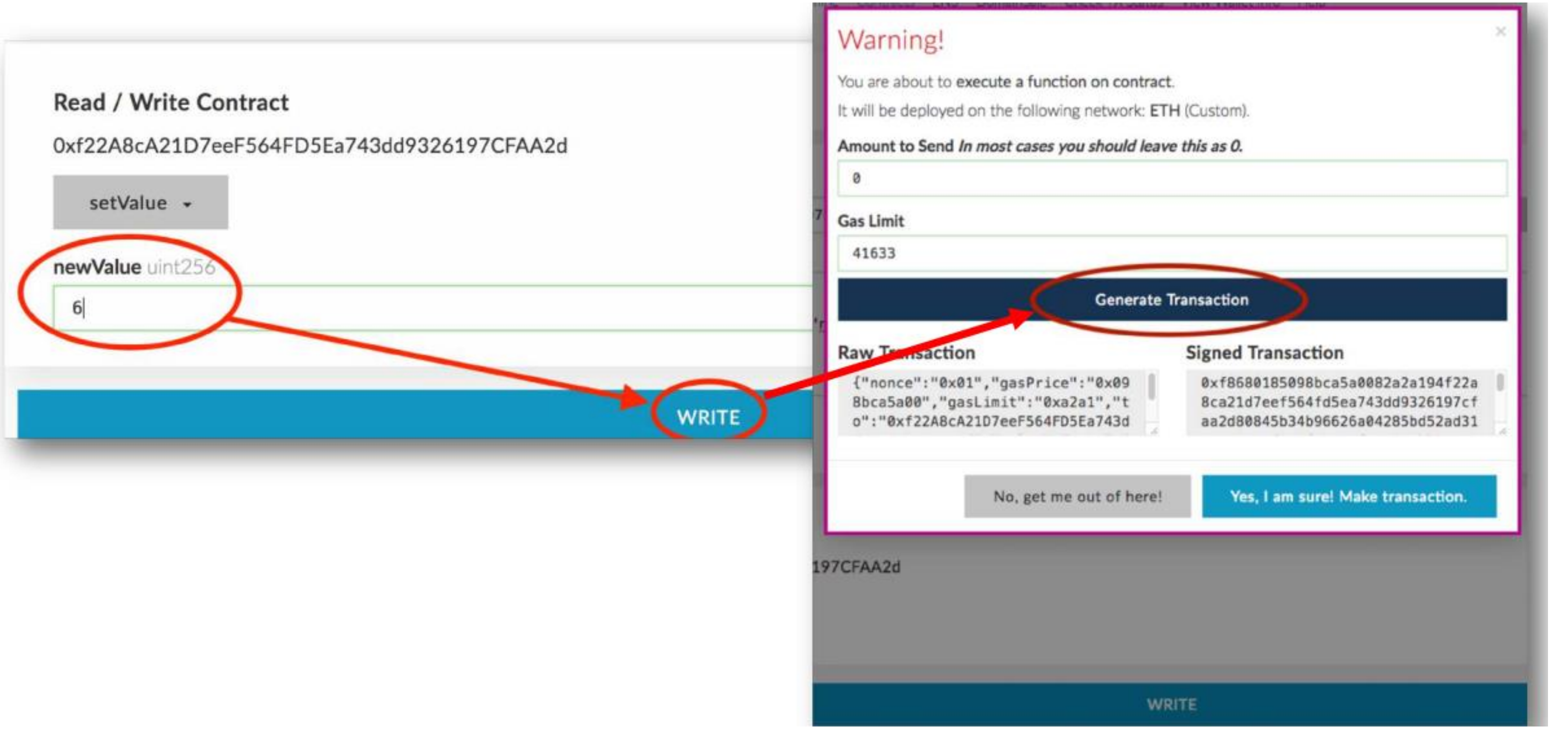
Read / Write Contract

0xf22A8cA21D7eeF564FD5Ea743dd9326197CFAA2d

↳ uint256

13

MyEtherWallet

- If you choose a function that updates the state of the contract, you will need to pay gas for it in a transaction
- 

Read / Write Contract

0xf22A8cA21D7eeF564FD5Ea743dd9326197CFAA2d

setValue

newValue uint256

6

WRITE

Warning!

You are about to execute a function on contract.

It will be deployed on the following network: ETH (Custom).

Amount to Send *In most cases you should leave this as 0.*

0

Gas Limit

41633

Generate Transaction

Raw Transaction

```
{"nonce":"0x01","gasPrice":"0x09
8bca5a00","gasLimit":"0xa2a1","t
o":"0xf22A8cA21D7eeF564FD5Ea743d
```

Signed Transaction

```
0xf8680185098bca5a0082a2a194f22a
8ca21d7eef564fd5ea743dd9326197cf
aa2d80845b34b96626a04285bd52ad31
```

No, get me out of here!

Yes, I am sure! Make transaction.

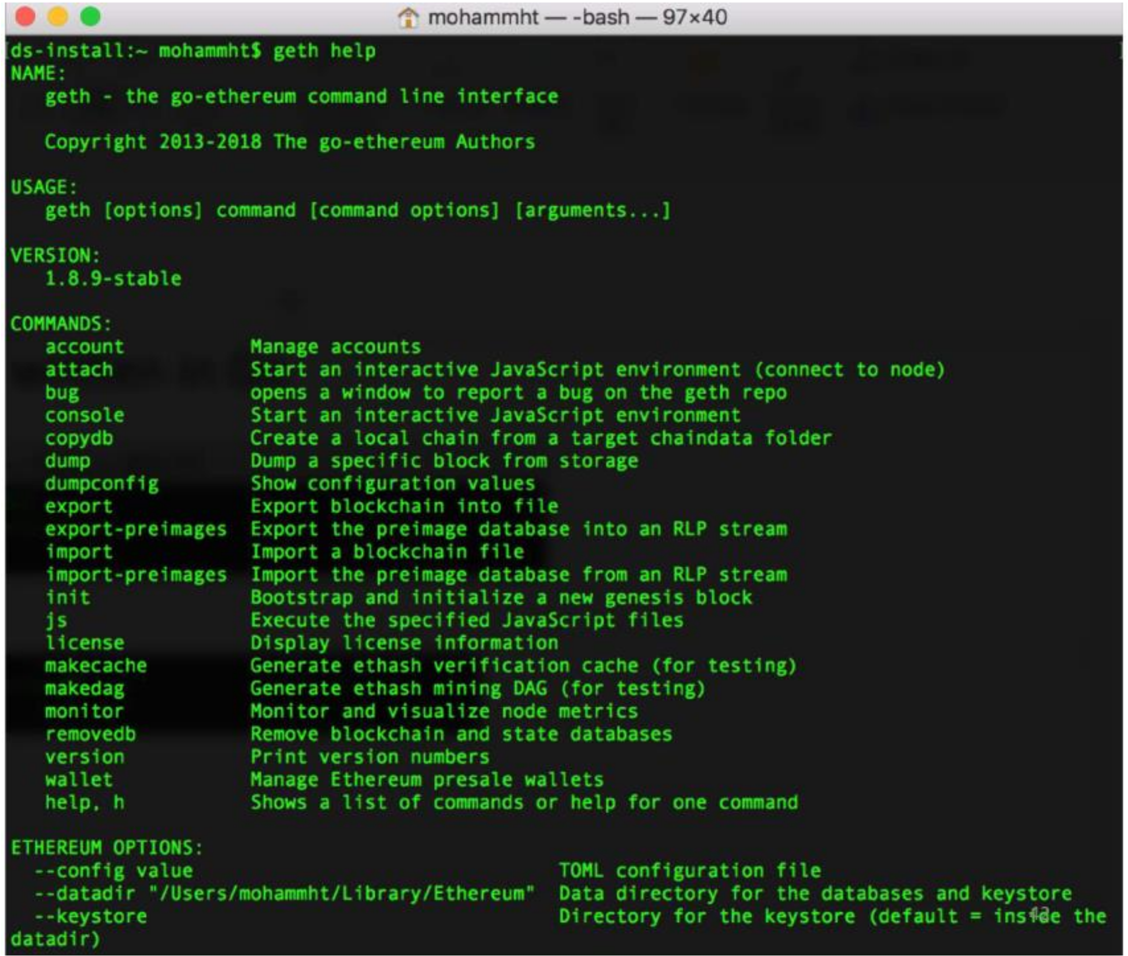
197CFAA2d

WRITE

Create Custom Ethereum Blockchain

- Instead of using Ganache with its default properties for private blockchain you can run your own blockchain
- **Install Geth:** One of the implementations of Ethereum written in Go
- Create the genesis block
- Create storage of the blockchain
- Deploy blockchain nodes
- Connect MyEtherWallet to your blockchain to interact with it

Geth help

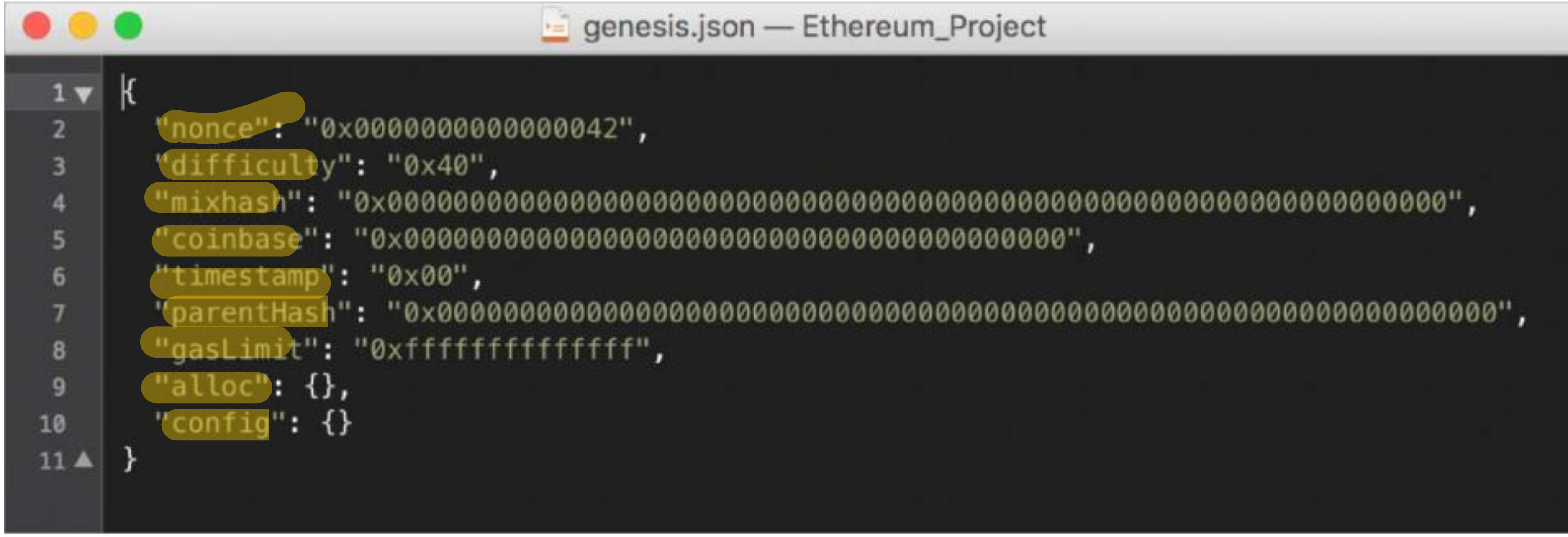


mohammht — -bash — 97×40

```
ds-install:~ mohammht$ geth help
NAME:
   geth - the go-ethereum command line interface

   Copyright 2013-2018 The go-ethereum Authors

USAGE:
   geth [options] command [command options] [arguments...]

VERSION:
   1.8.9-stable

COMMANDS:
   account           Manage accounts
   attach            Start an interactive JavaScript environment (connect to node)
   bug               opens a window to report a bug on the geth repo
   console           Start an interactive JavaScript environment
   copydb            Create a local chain from a target chaindata folder
   dump              Dump a specific block from storage
   dumpconfig        Show configuration values
   export            Export blockchain into file
   export-preimages  Export the preimage database into an RLP stream
   import            Import a blockchain file
   import-preimages  Import the preimage database from an RLP stream
   init              Bootstrap and initialize a new genesis block
   js                Execute the specified JavaScript files
   license           Display license information
   makecache         Generate ethash verification cache (for testing)
   makedag           Generate ethash mining DAG (for testing)
   monitor           Monitor and visualize node metrics
   removedb          Remove blockchain and state databases
   version           Print version numbers
   wallet            Manage Ethereum presale wallets
   help, h           Shows a list of commands or help for one command

ETHEREUM OPTIONS:
  --config value                           TOML configuration file
  --datadir "/Users/mohammht/Library/Ethereum"  Data directory for the databases and keystore
  --keystore                               Directory for the keystore (default = inside the
datadir)
```

Genesis block

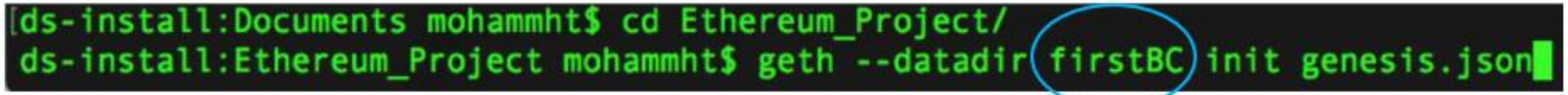
- The first block in the chain and a json file that stores the configuration of the chain
- Create and store the file as genesis.json



```
{
  "nonce": "0x0000000000000042",
  "difficulty": "0x40",
  "mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000000000000000000",
  "timestamp": "0x00",
  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "gasLimit": "0xffffffffffff",
  "alloc": {},
  "config": {}
}
```

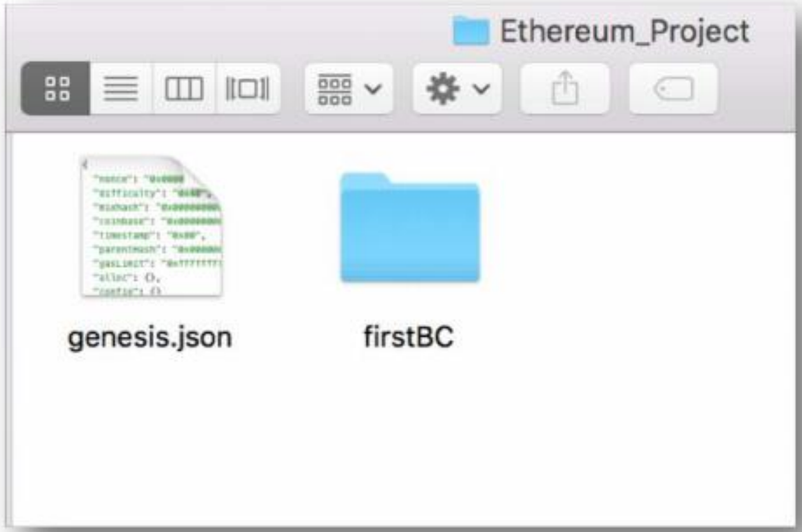
Create the storage of the blockchain

- Go to the directory of the genesis.json file
- Specify directory of your blockchain
- Create the storage from the genesis block



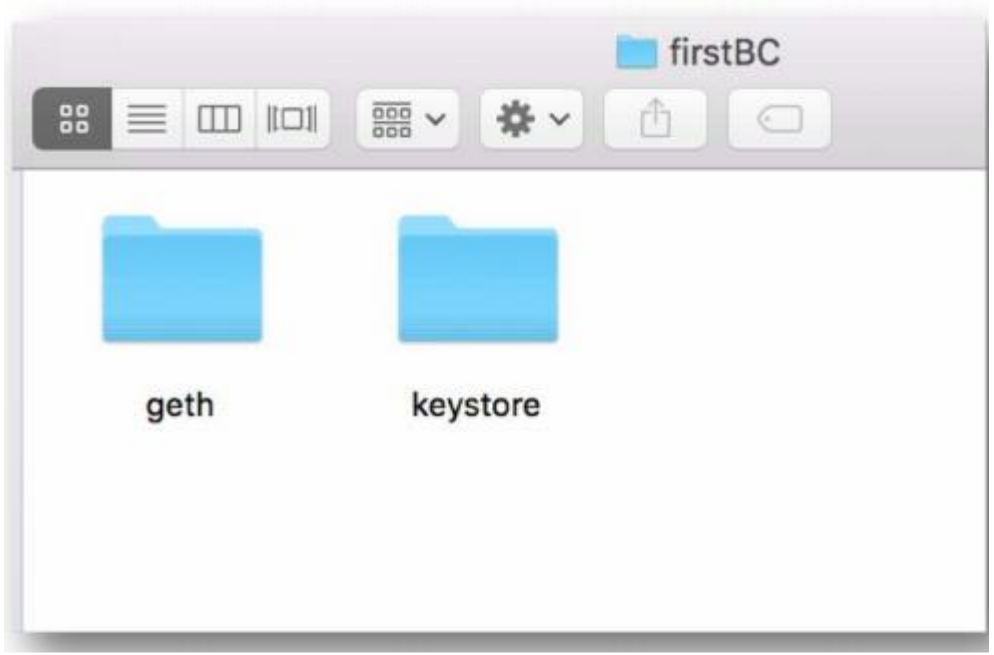
```
[ds-install:Documents mohammht$ cd Ethereum_Project/
ds-install:Ethereum_Project mohammht$ geth --datadir firstBC init genesis.json
```

Folder name of your blockchain



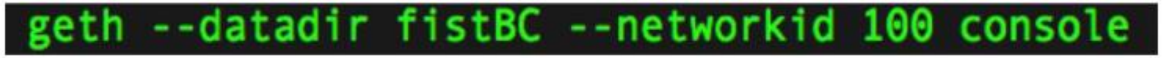
Inside the Blockchain Folder

- geth folder: Store your database
- keystore: Store your Ethereum accounts



Start the Ethereum peer node

- Start the blockchain

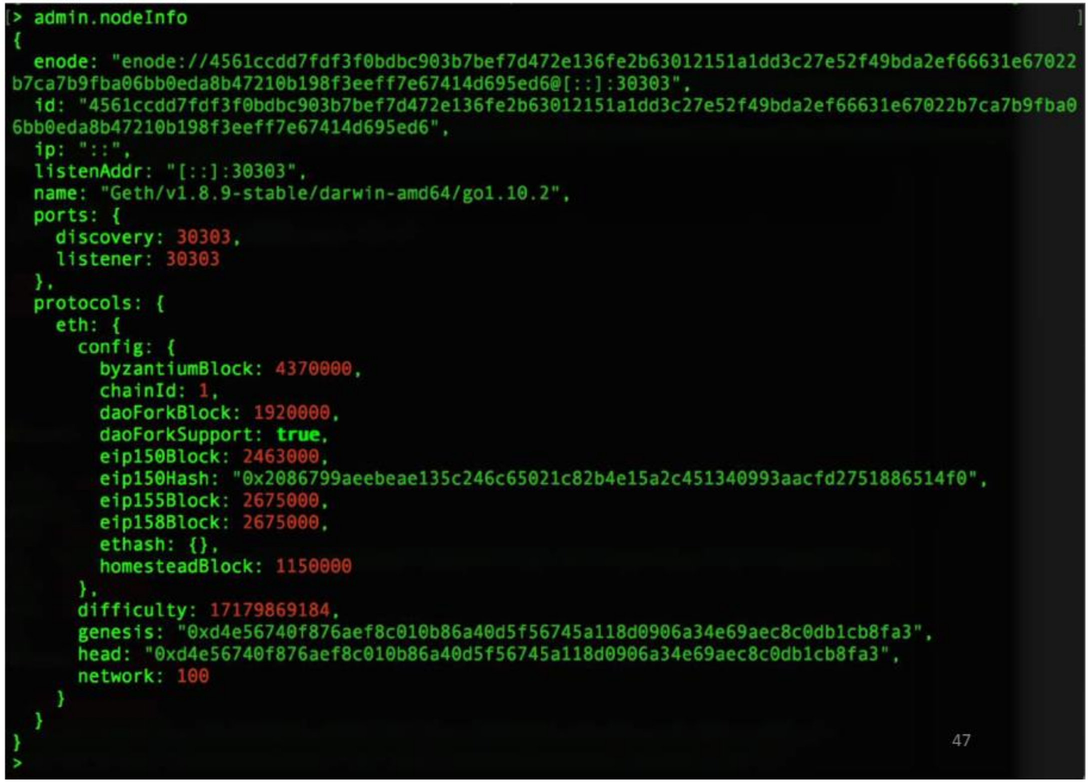


```
geth --datadir fistBC --networkid 100 console
```

- Networkid provides privacy for your network.
- Other peers joining your network must use the same networkid.

Blockchain started

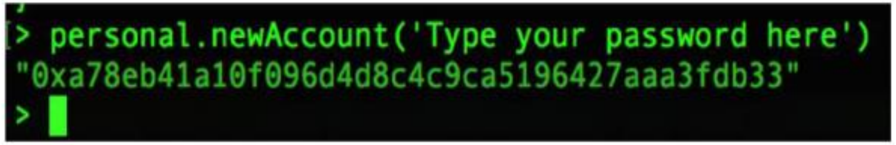
- Type *admin.nodeInfo* to get the information about your current node



```
> admin.nodeInfo
{
  enode: "enode://4561ccdd7fdf3f0bdbc903b7bef7d472e136fe2b63012151a1dd3c27e52f49bda2ef66631e67022
b7ca7b9fba06bb0eda8b47210b198f3eeff7e67414d695ed6@[::]:30303",
  id: "4561ccdd7fdf3f0bdbc903b7bef7d472e136fe2b63012151a1dd3c27e52f49bda2ef66631e67022b7ca7b9fba0
6bb0eda8b47210b198f3eeff7e67414d695ed6",
  ip: "::",
  listenAddr: "[::]:30303",
  name: "Geth/v1.8.9-stable/darwin-amd64/go1.10.2",
  ports: {
    discovery: 30303,
    listener: 30303
  },
  protocols: {
    eth: {
      config: {
        byzantiumBlock: 4370000,
        chainId: 1,
        daoForkBlock: 1920000,
        daoForkSupport: true,
        eip150Block: 2463000,
        eip150Hash: "0x2086799aeebeae135c246c65021c82b4e15a2c451340993aacfd2751886514f0",
        eip155Block: 2675000,
        eip158Block: 2675000,
        ethash: {},
        homesteadBlock: 1150000
      },
      difficulty: 17179869184,
      genesis: "0xd4e56740f876aef8c010b86a40d5f56745a118d0906a34e69aec8c0db1cb8fa3",
      head: "0xd4e56740f876aef8c010b86a40d5f56745a118d0906a34e69aec8c0db1cb8fa3",
      network: 100
    }
  }
}
>
```

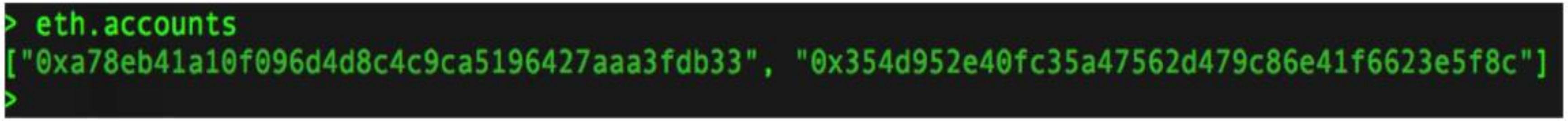
Create an account

- Type *personal.newAccount* to create as many accounts as you need



```
> personal.newAccount('Type your password here')
"0xa78eb41a10f096d4d8c4c9ca5196427aaa3fdb33"
> 
```

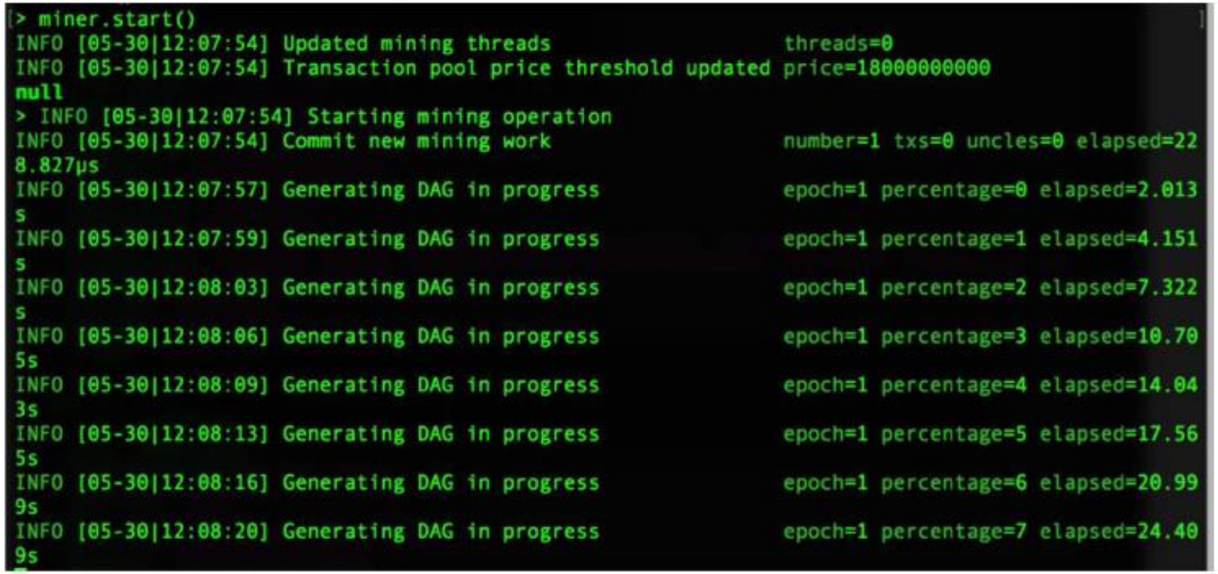
- See the created account(s)



```
> eth.accounts
["0xa78eb41a10f096d4d8c4c9ca5196427aaa3fdb33", "0x354d952e40fc35a47562d479c86e41f6623e5f8c"]
>
```

Mining

- Type *miner.start()* to start mining



```
> miner.start()
INFO [05-30|12:07:54] Updated mining threads                   threads=0
INFO [05-30|12:07:54] Transaction pool price threshold updated price=18000000000
null
> INFO [05-30|12:07:54] Starting mining operation
INFO [05-30|12:07:54] Commit new mining work                   number=1 txs=0 uncles=0 elapsed=22
8.827µs
INFO [05-30|12:07:57] Generating DAG in progress               epoch=1 percentage=0 elapsed=2.013
s
INFO [05-30|12:07:59] Generating DAG in progress               epoch=1 percentage=1 elapsed=4.151
s
INFO [05-30|12:08:03] Generating DAG in progress               epoch=1 percentage=2 elapsed=7.322
s
INFO [05-30|12:08:06] Generating DAG in progress               epoch=1 percentage=3 elapsed=10.70
5s
INFO [05-30|12:08:09] Generating DAG in progress               epoch=1 percentage=4 elapsed=14.04
3s
INFO [05-30|12:08:13] Generating DAG in progress               epoch=1 percentage=5 elapsed=17.56
5s
INFO [05-30|12:08:16] Generating DAG in progress               epoch=1 percentage=6 elapsed=20.99
9s
INFO [05-30|12:08:20] Generating DAG in progress               epoch=1 percentage=7 elapsed=24.40
9s
```

- Type *miner.stop()* to stop mining

THANK YOU