

Ethereum Smart Contract Example

07-05-2022

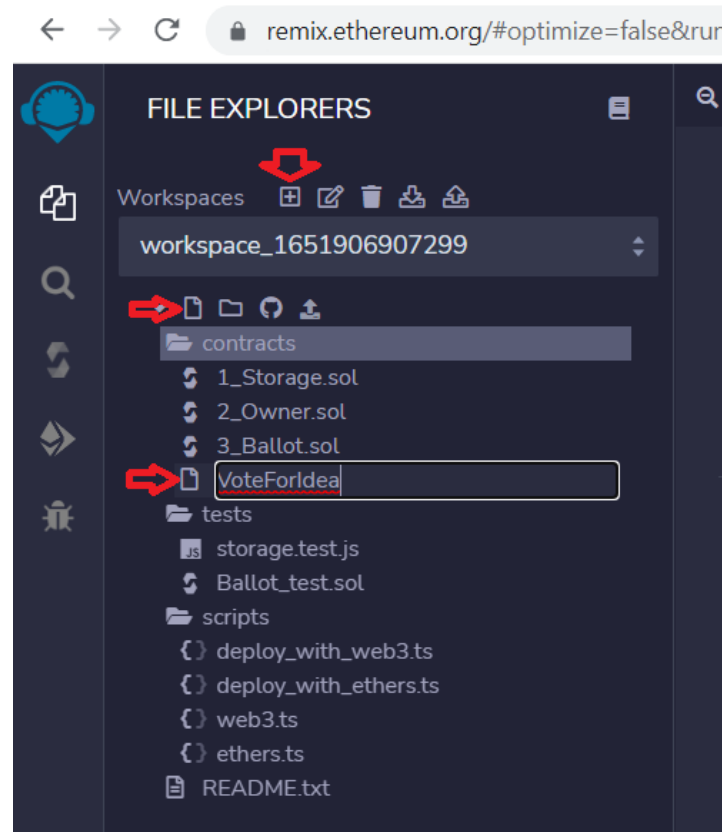
OBJECTIVE

The objective of this example is to get to know how to write a solidity programming language to create a simple voting smart contract in Ethereum blockchain testnet.

STEP 1: IDE

Go to <https://remix.ethereum.org> , REMIX is an online Ethereum IDE

STEP 2: Create a new workspace and add a new contract called "VoteForIdea"



STEP 3: Write the code "VoteForIdea.sol"

// Simple election Ethereum Smart contract

// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

// The objective of the contract is to ask for ideas from the public

// If an action should take place or not

```

contract VoteForIdea{
    //Vote contains the voter's address and their choice true or false (agree or disagree)
    struct vote{
        address voterAdr;
        bool choice;
    }

    //Voter contains the name of the voter and if they have voted or not yet
    struct voter{
        string voterName;
        bool voted;
    }

    uint private countResult =0;
    uint public finalResult =0;
    uint public totalVoter=0;
    uint public totalVote=0;

    //Official is the owner of the contract and starts the voting campaign and his proposal action
    address public officialAdr;
    string public officialName;
    string public proposal;

    mapping (uint => vote) private votes;
    mapping (address => voter) public voterRegistry;

    //The contract has three states
    enum State {Created, Voting, Ended}
    State public state;

    //Modifier is used for input validation
    modifier inState(State _state) {
        require(state == _state);
        _;
    }
    modifier onlyOwner {
        require(msg.sender == officialAdr);
        _;
    }

```

```
}
```

//When the contract is started or created, it has an official, proposal and the state of the contract is created

```
constructor(  
    string memory _officialName,  
    string memory _proposal  
)  
{  
    officialAdr = msg.sender;  
    officialName = _officialName;  
    proposal = _proposal;  
    state = State.Created;  
}
```

//Only the official can add voters to vote, and only when the contract is created

```
function addVoter(  
    address _voterAdr,  
    string memory _voterName  
)  
public  
inState(State.Created)  
onlyOwner  
{  
    voter memory v;  
    v.voterName = _voterName;  
    v.voted = false;  
    voterRegistry[_voterAdr] = v;  
    totalVoter++;  
}
```

//Official begins the voting campaign

```
function startVote()  
public  
inState(State.Created)  
onlyOwner  
{  
    state = State.Voting;  
}
```

//voter that has been added by an official can vote now

```

function doVote
(
    bool _choice
)
public
inState(State.Voting)
returns (bool voted)
{
    bool found = false;
    if (bytes(voterRegistry[msg.sender].voterName).length != 0
    && !voterRegistry[msg.sender].voted
    ){
        voterRegistry[msg.sender].voted = true;
        vote memory v;
        v.voterAdr = msg.sender;
        v.choice = _choice;
        if(_choice){
            countResult++;
        }
        votes[totalVote] = v;
        totalVote++;
        found = true;
    }
    return found;
}

```

//Official ended the voting campaign

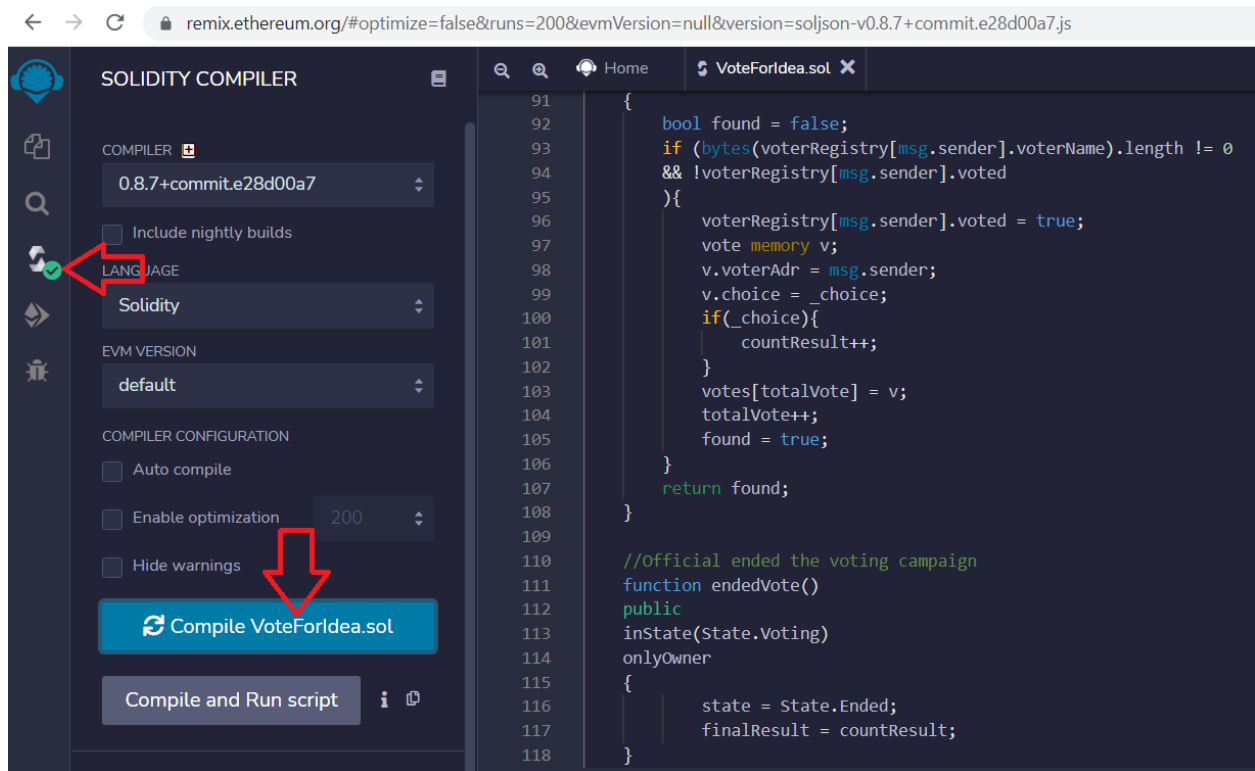
```

function endedVote()
public
inState(State.Voting)
onlyOwner
{
    state = State.Ended;
    finalResult = countResult;
}
}

```

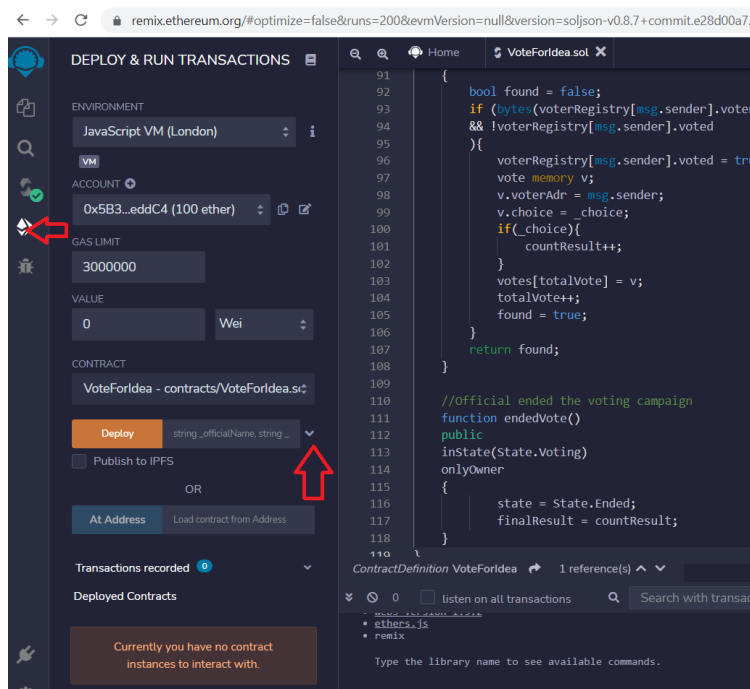
STEP 4: Compile the code

Click on the Menu compile on the left side and click on the button compile

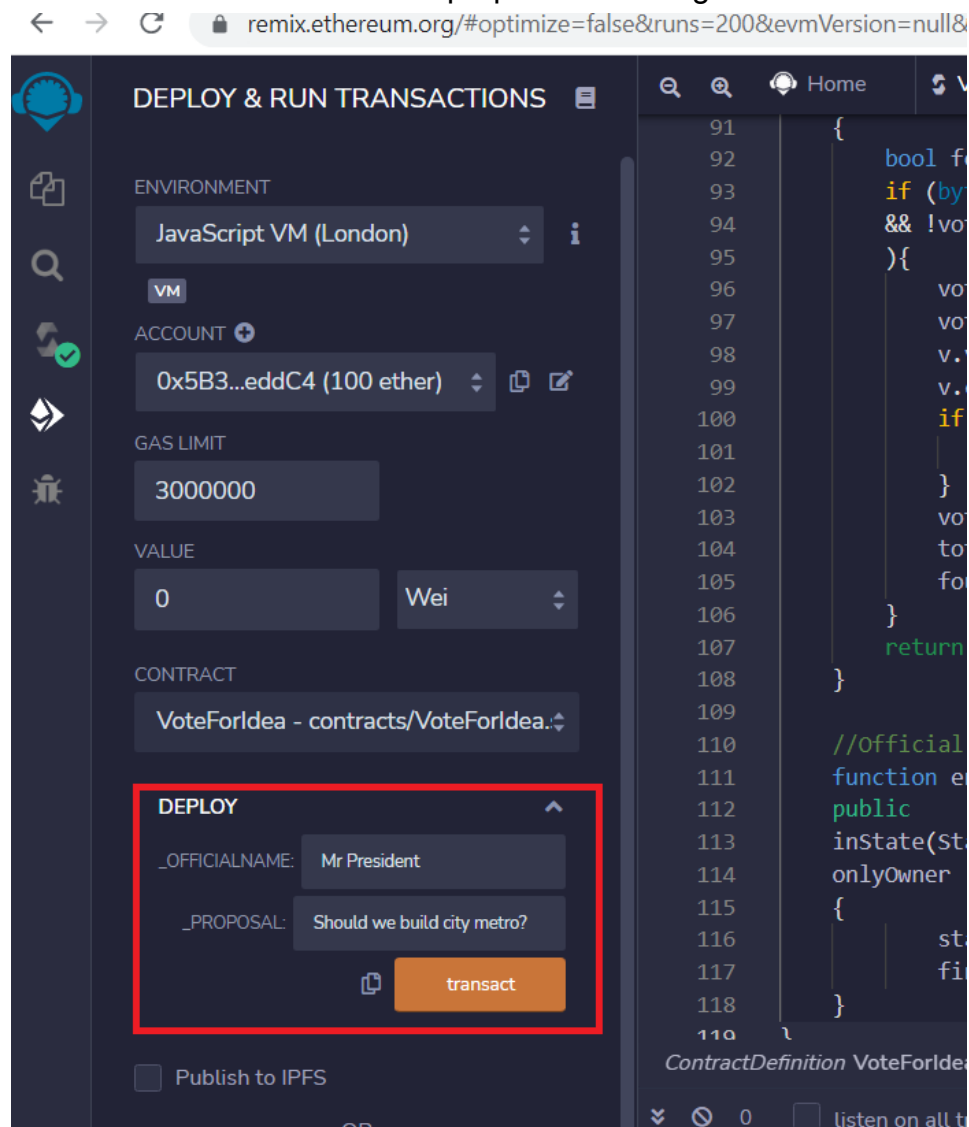


STEP 5: Deploy the contract

Click on the menu deploy on the left side, and Click on the arrow dropdown to prepare for deploy



Fill in the official name and the proposal for voting and click on transact button.



The contract has been deployed successfully, and now you can click on the arrow of the created contract to see and test those codes that you have written

remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljs

DEPLOY & RUN TRANSACTIONS

ACCOUNT: 0x5B3...eddC4 (99.999999%)

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: VoteForIdea - contracts/VoteForIdea

DEPLOY

_OFFICIALNAME: Mr President

_PROPOSAL: Should we build city metro?

☐ Publish to IPFS

OR

Transactions recorded 1

Deployed Contracts

> VOTEFORIDEA AT 0XD91...39138 (ME)

VoteForIdea.sol

```
91 {
92   bool found = false;
93   if (bytes(voterReg
94     && !voterRegistry[
95     ){
96     voterRegistry[
97     vote memory v;
98     v.voterAdr = m
99     v.choice = _ch
100    if(_choice){
101      countResult
102    }
103    votes[totalVot
104    totalVote++;
105    found = true;
106  }
107  return found;
108 }
109
110 //Official ended the v
111 function endedVote()
112 public
113 inState(State.Voting)
114 onlyOwner
115 {
116   state = State.
117   finalResult =
118 }
119
120 ContractDefinition VoteForIdea 1 reference
```

☒ [vm] from: 0x5B3...eddC4 to: Vo

STEP 6: Testing all the functionalities

remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljs-v0

DEPLOY & RUN TRANSACTIONS

Transactions recorded 1

Deployed Contracts

> VOTEFORIDEA AT 0XD91...39138 (ME)

address_voterAdr, string _

bool _choice

address

VoteForIdea.sol

```
91 {
92   bool found = false;
93   if (bytes(voterRegis
94     && !voterRegistry[
95     ){
96     voterRegistry[
97     vote memory v;
98     v.voterAdr = msg.s
99     v.choice = _choic
100    if(_choice){
101      countResult++;
102    }
103    votes[totalVote]
104    totalVote++;
105    found = true;
106  }
107  return found;
108 }
109
110 //Official ended the vot
111 function endedVote()
112 public
113 inState(State.Voting)
114 onlyOwner
115 {
116   state = State.End
117   finalResult = cou
118 }
119
120 ContractDefinition VoteForIdea 1 reference
```

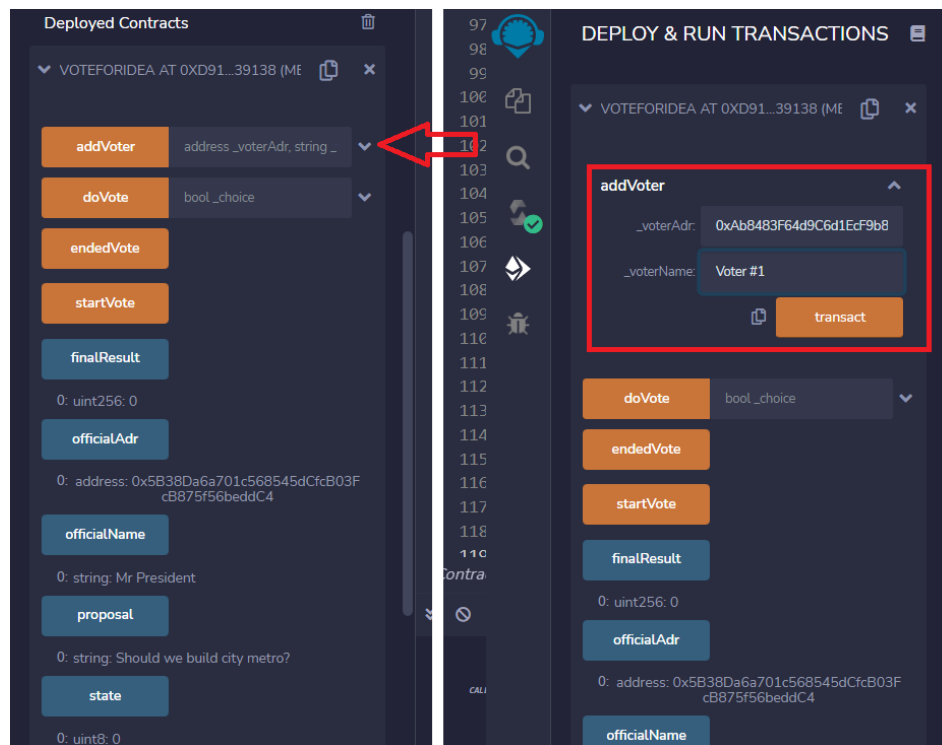
☒ [vm] from: 0x5B3...eddC4 to: VoteForIdea

You can click on the blue buttons to see the initial data in your contract such as Official name, proposal, state, total voter, and final result.....

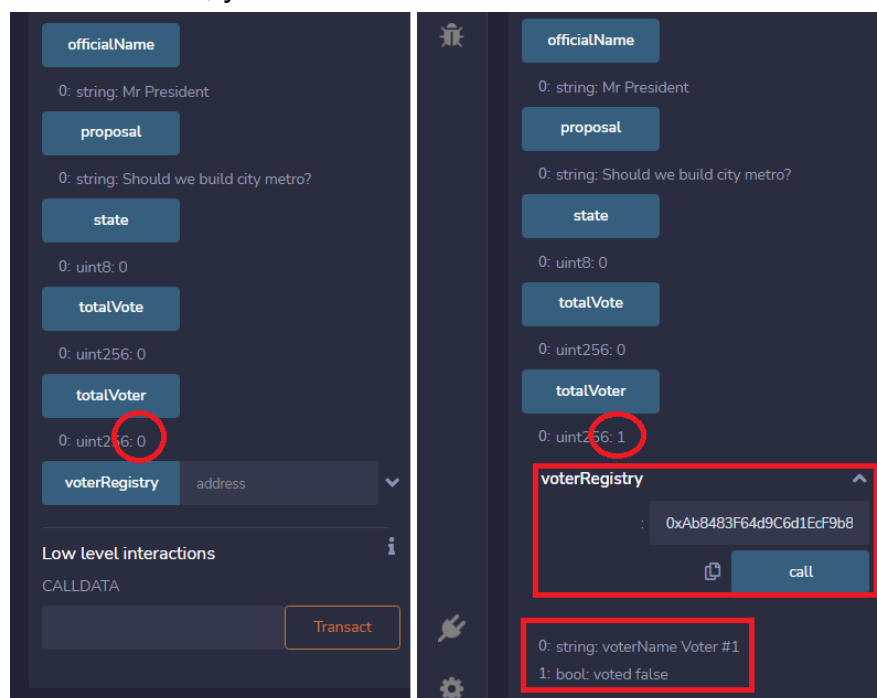
The screenshot displays the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel lists various contract functions and their current states. The functions listed are: **doVote** (with a dropdown menu set to 'bool_choice'), **endedVote**, **startVote**, **finalResult** (0: uint256: 0), **officialAdr** (0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4), **officialName** (0: string: Mr President), **proposal** (0: string: Should we build city metro?), **state** (0: uint8: 0), **totalVote** (0: uint256: 0), **totalVoter** (0: uint256: 0), and **voterRegistry** (with a dropdown menu set to 'address'). Red arrows point to each of these function buttons. On the right, the Solidity code editor shows the source code for 'VoteForIdea.sol'. The code includes a function 'doVote' (lines 91-106) and a function 'endedVote' (lines 111-119). The 'endedVote' function is marked as 'public' and 'onlyOwner'. The bottom of the interface shows a transaction log with a single entry: '[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4'.

If you scroll to the top you can see that, in order to deploy the Ethereum Smart contract, you need to pay a fee in the form of cryptocurrency. So the money is taken from the first test address account. Initially, the account had 100 Ether, now it has only 99.9999 Ether. This first address is used for the official address, and others are used for voters. So only the official address account and add voter to the contract, and others addresses that have been added could be used for voting.

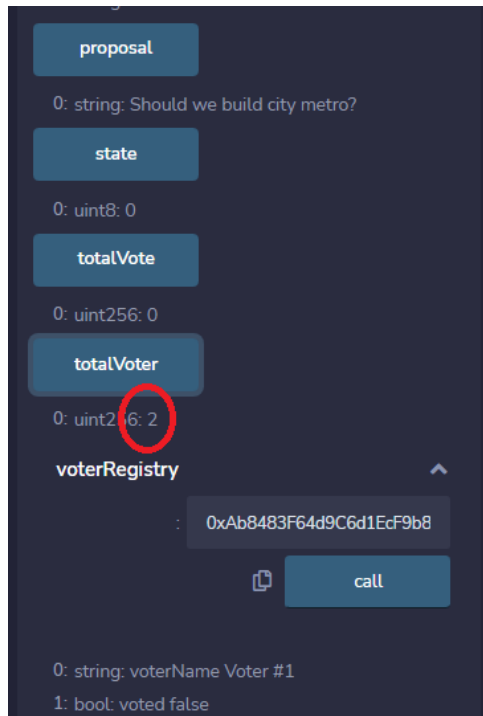
Scroll down and click on the dropdown arrow of addvoter, paste the 2nd address that we have just copied and add voter name “Voter #1”, and then click on transact.



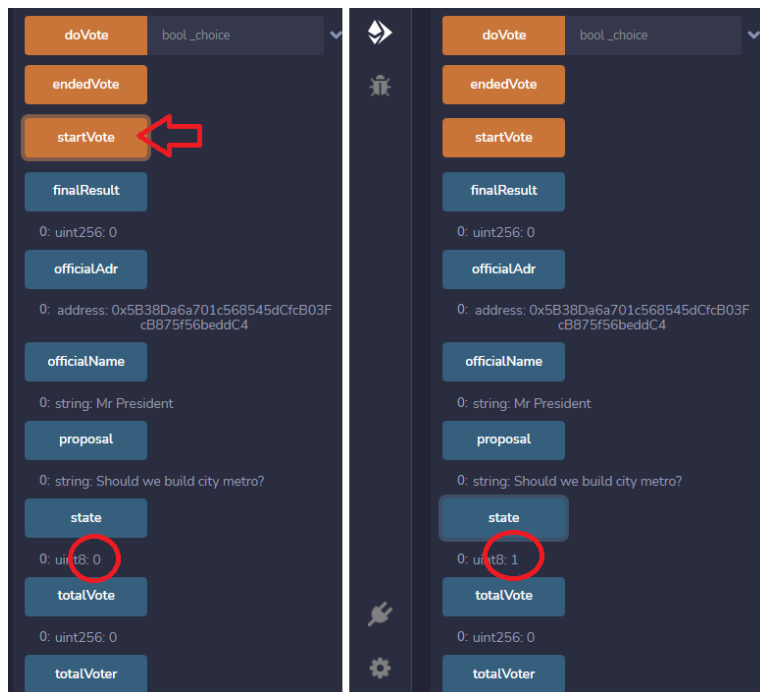
So before the totalVoter is 0, and now when you click on the button again it will give 1. You can also copy and paste the address of the voter #1 into voterRegister and click on the call button, you will see the voter's name and he/she has not voted yet “Vote = false”



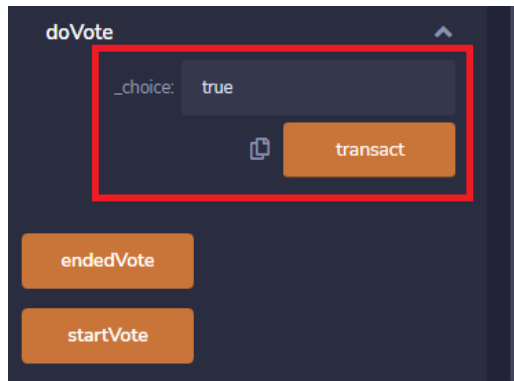
You can add as many voters as you want.



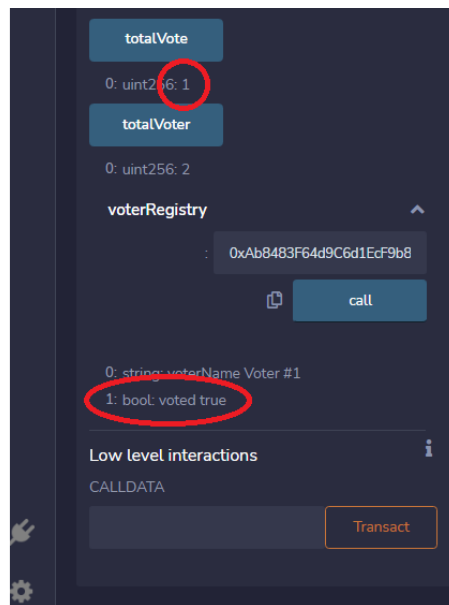
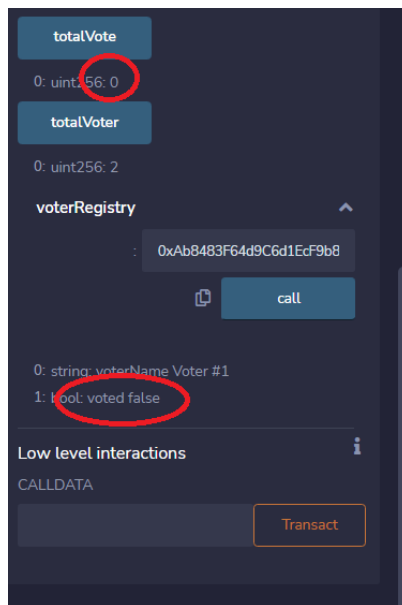
Now, using the official address to start the voting process by clicking on the button startvote and you can see that the state is changed from 0 to 1, which means from created state to voting state.



Use one of the voter addresses to vote, by typing “true” in the doVote function and clicking on the button transact.



Now the totalVote becomes 1 and Voter #1 has already voted



So using the official address to stop the voting, by clicking on endedVote, and see the finalresult. The contract state changes to 2 which is ended state.

