Building Smart Contracts with Remix

NINA BREZNIK

@ninabreznik

YANN LEVREAU

@ninabreznik

IURI MATIAS

@iurimatias

ROB STUPAY

@ryestew

ALEX PRAETORIUS

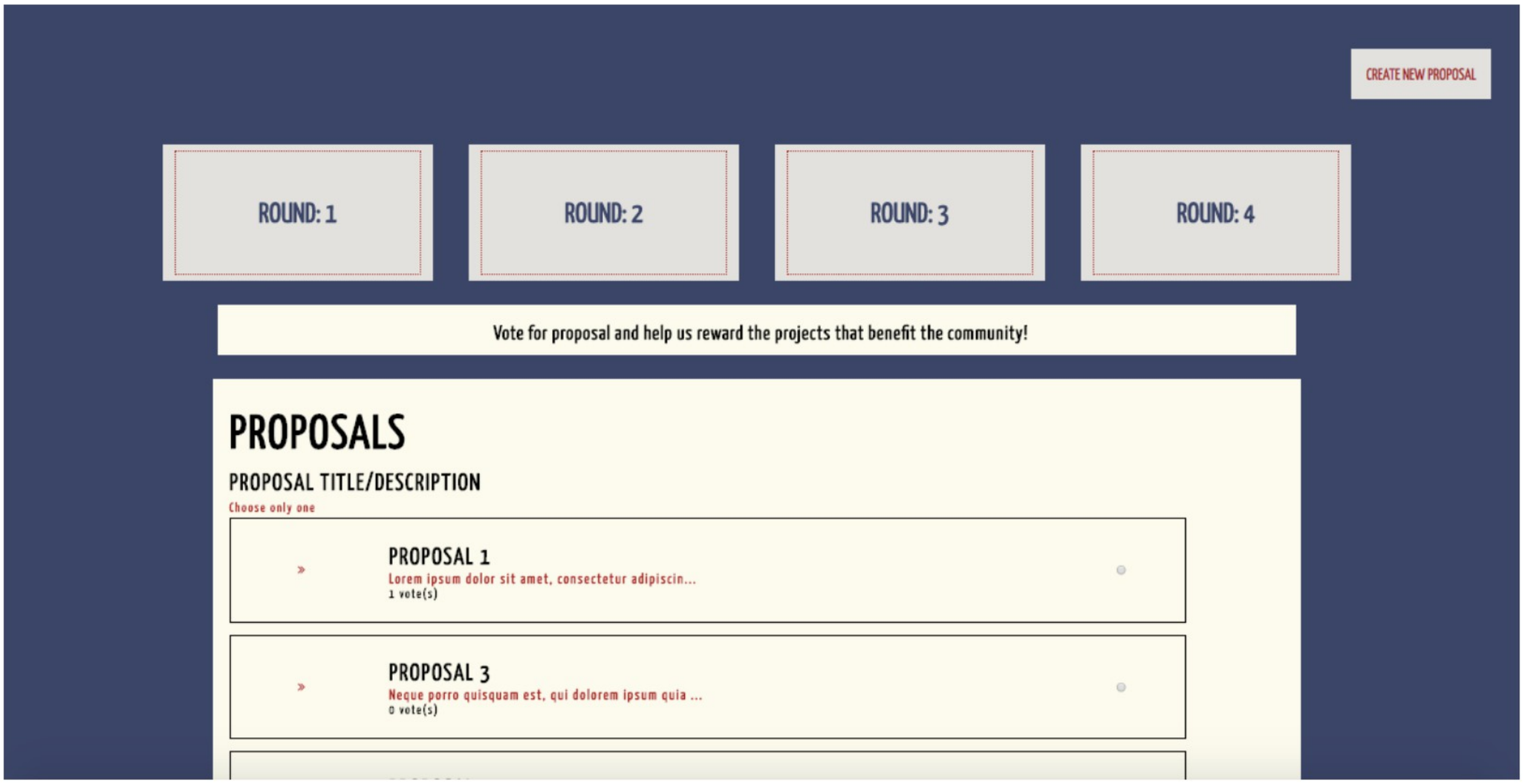
@SERAPATH

LIANA HUSIKYAN

@LianaHus

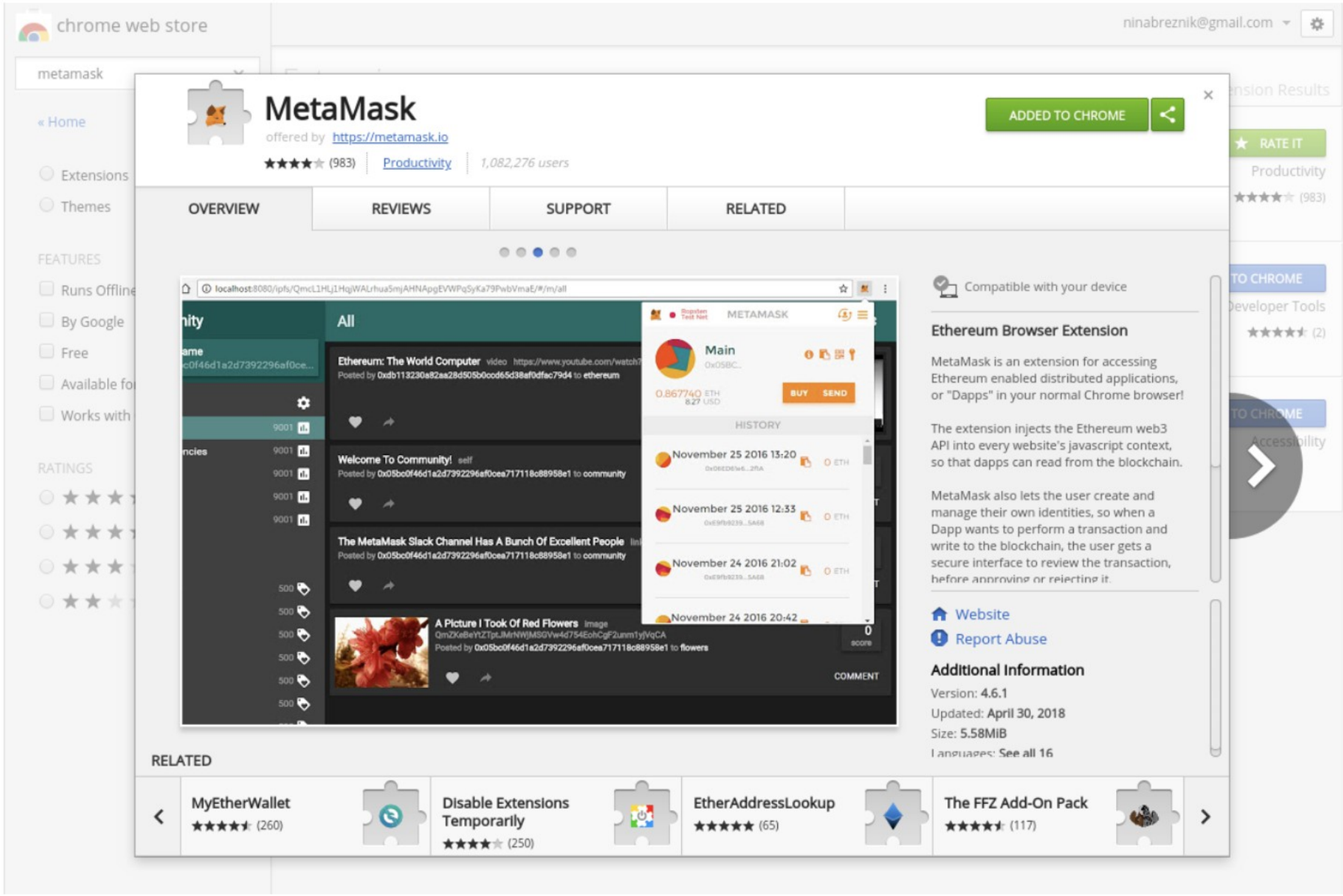
Ballot Dapp Workshop

bit.ly/remix-workshop-repository



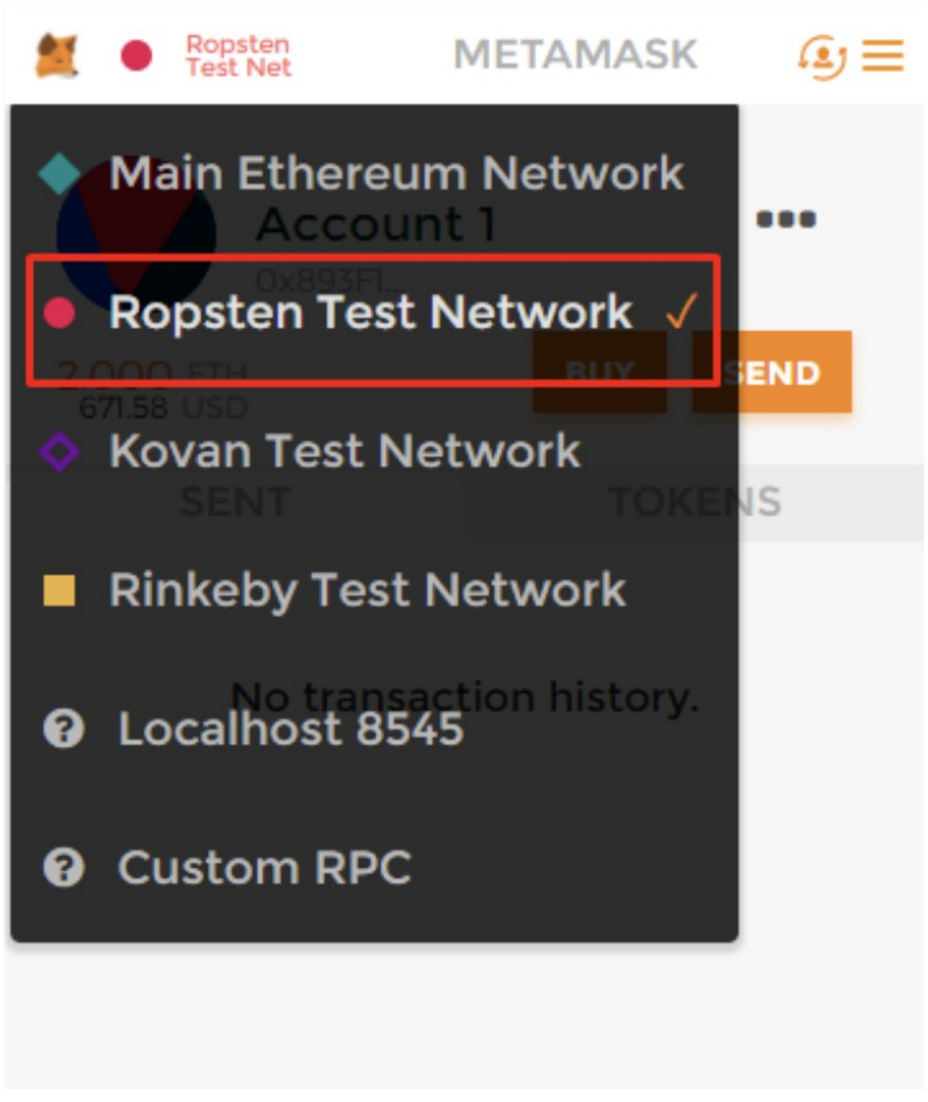
Install Metamask

chrome.google.com/webstore



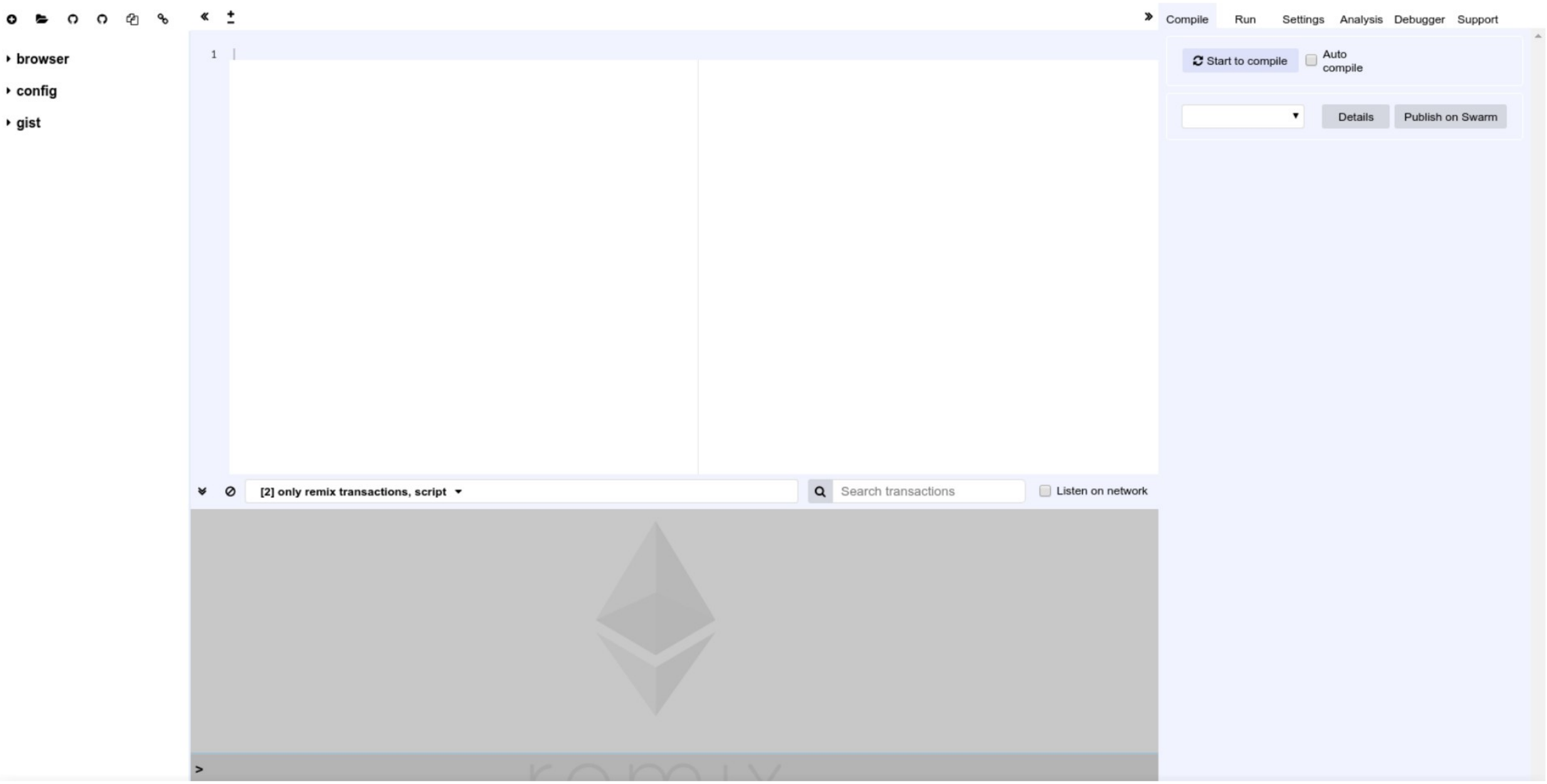
Login to Metamask

Robsten Test Network



Let’s get started

https://bit.ly/remix-workshop-repository



Remix Tour

https://remix-alpha.ethereum.org

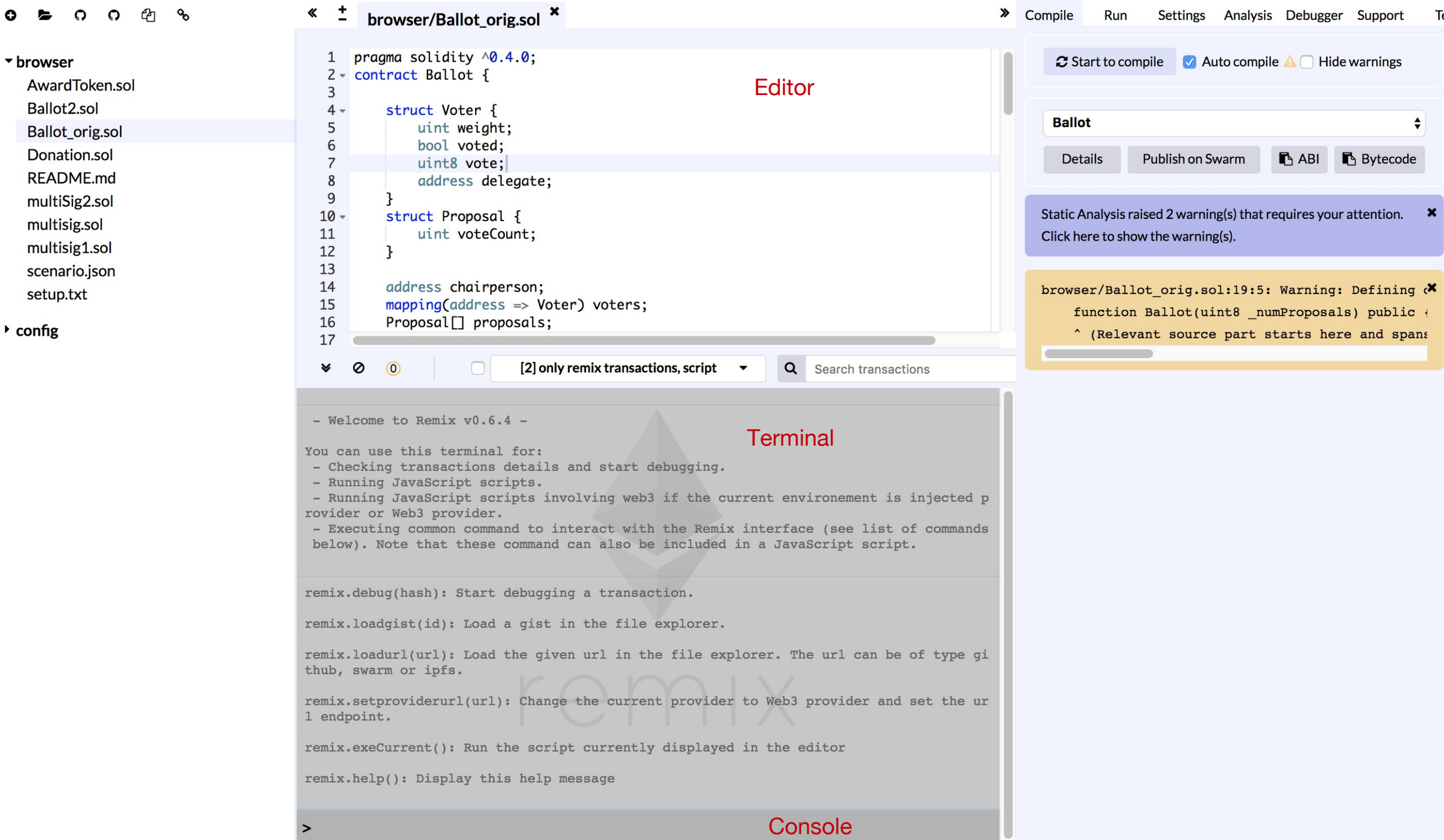
File Explorer

Compile Tab (active)

Editor

Terminal

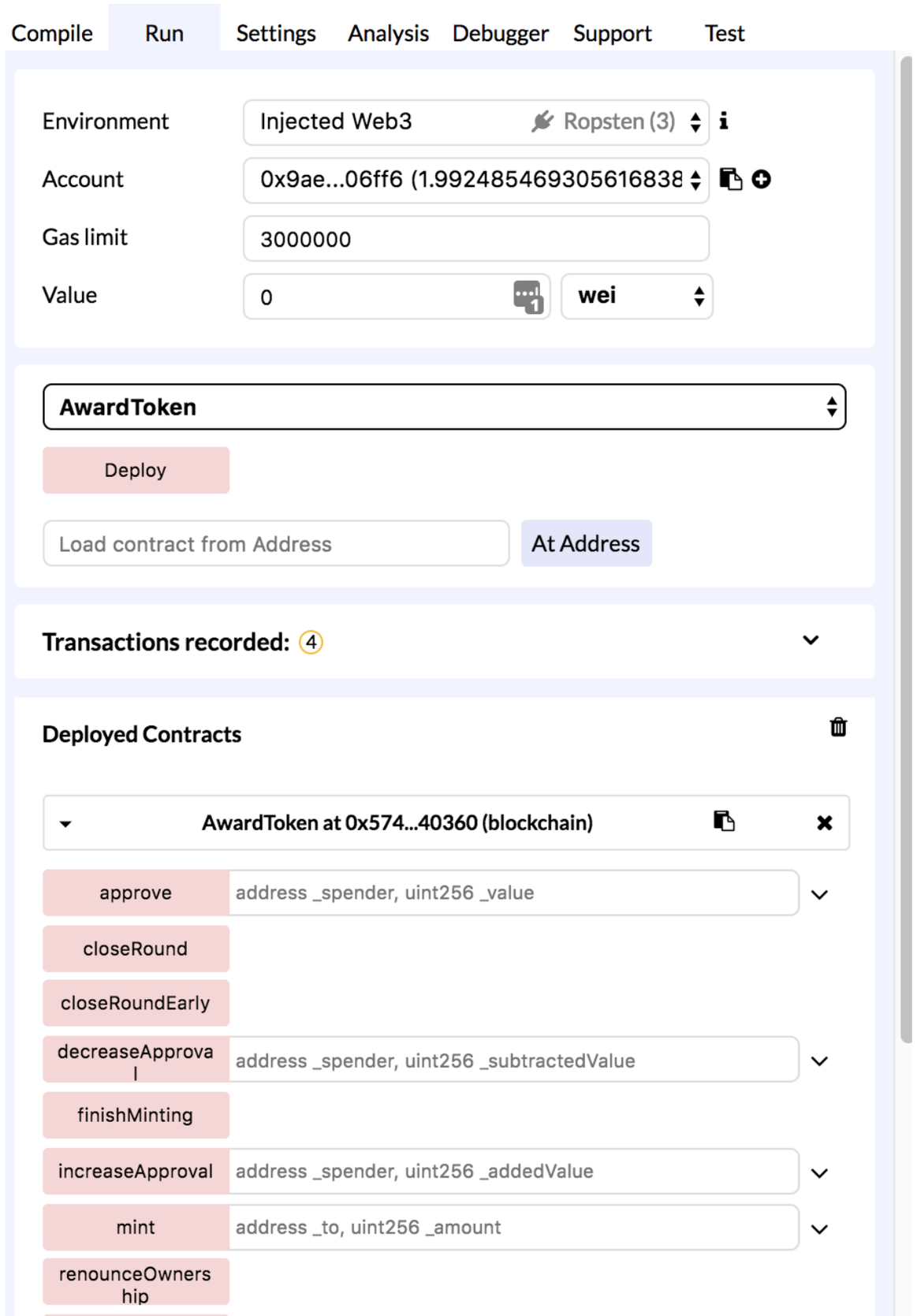
Console



Run Tab

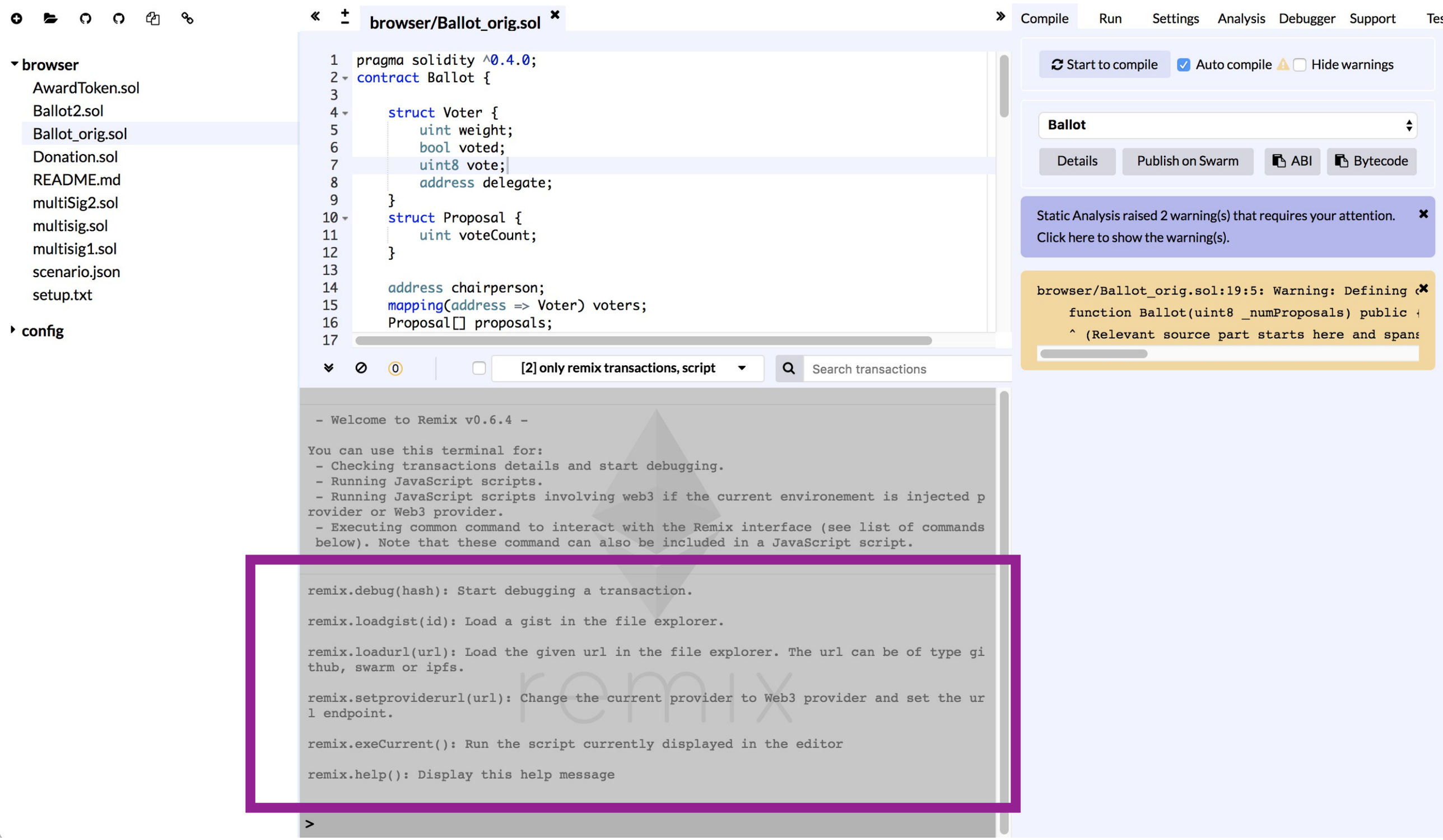
Universal DAPP

UI to the Contract



Remix Commands

https://remix-alpha.ethereum.org



Set environment

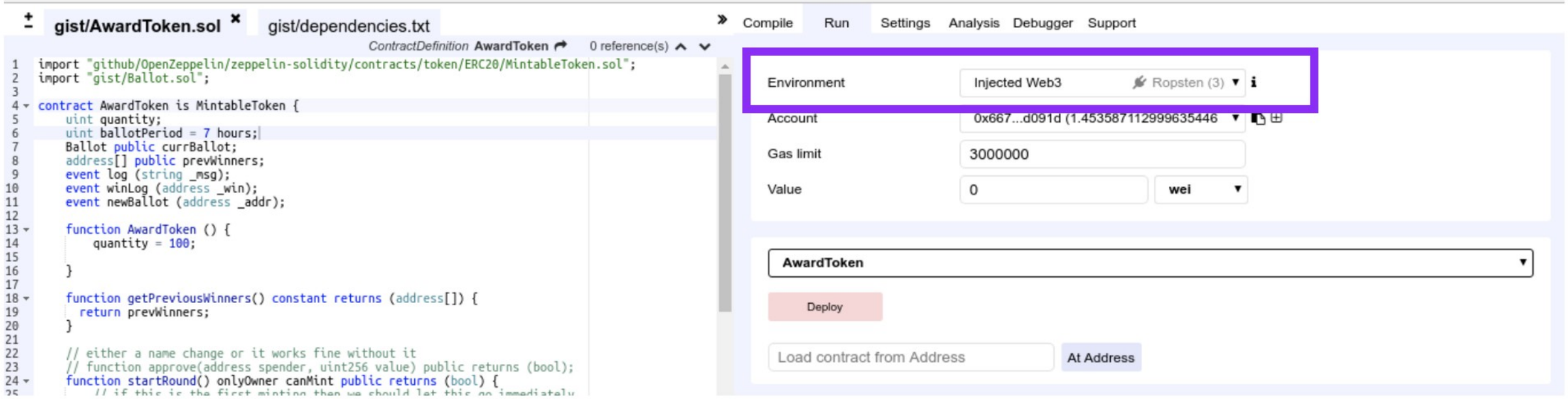
Run tab-> Environment:

Javascript VM - a simple and quick environment – only visible to your machine

Injected web3 (Ropsten) – or whatever testnet you have setup in metamask

Web3 Provider

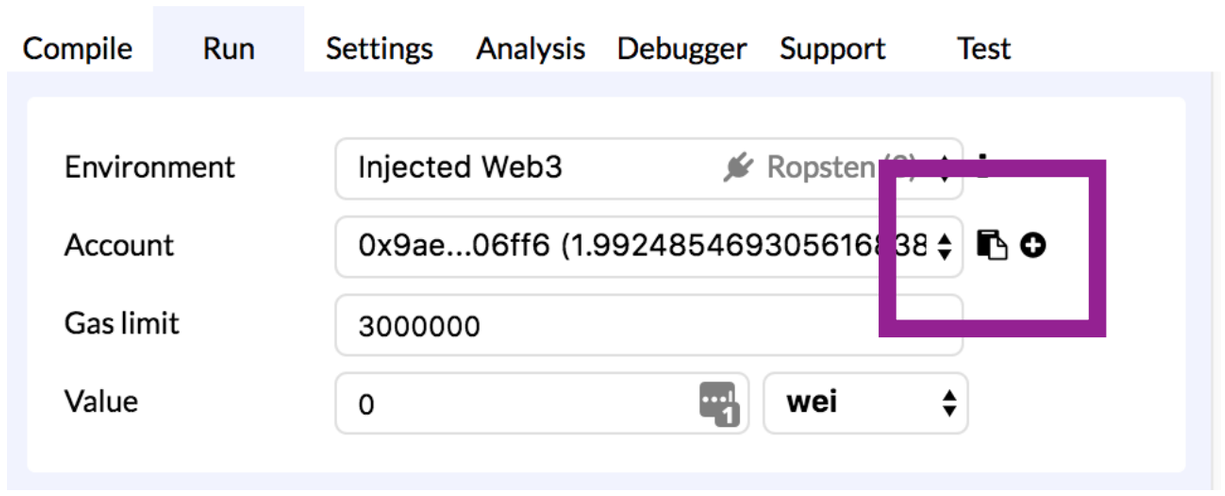
Choose: **Injected web3 (Ropsten)**



Get some TEST ether

http://faucet.ropsten.be:3001/

But FIRST:
Copy your address - here or in Metamask



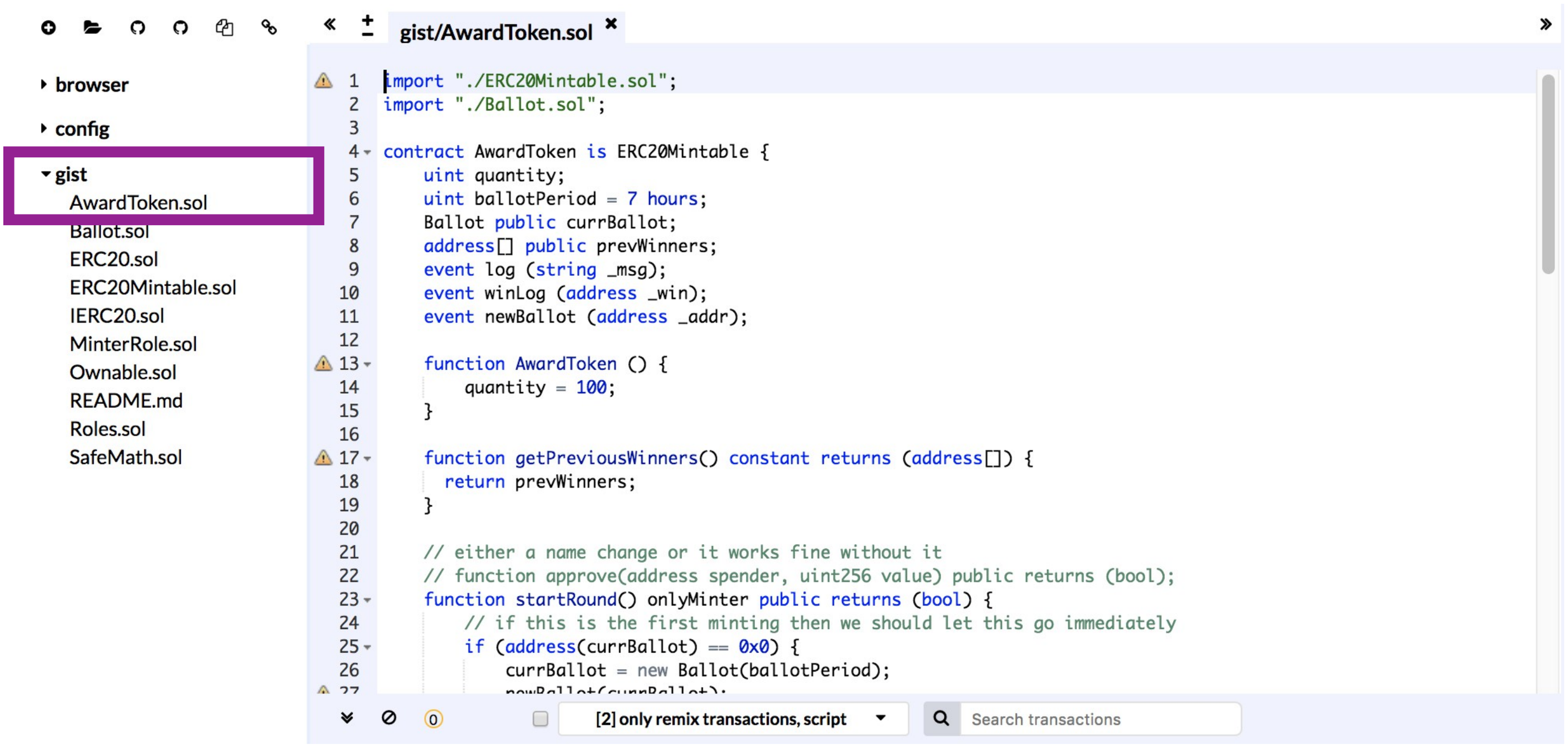
Load files to Remix

remix.loadgist('add9633c2b0101f6fda0aadcfe350f60')

here in the console

Open file

gist/AwardToken



Compile the contract

Compile tab: Start to compile button

(when dependencies.js is the active file)

gist/dependencies.js | gist/AwardToken.sol

```
1  import "github/OpenZeppelin/zeppelin-solidity/contracts/token/ERC20/MintableToken.sol";
2  import "gist/Ballot.sol";
3
4  contract AwardToken is MintableToken {
5      uint quantity;
6      uint ballotPeriod = 7 hours;
7      Ballot public currBallot;
8      address[] public prevWinners;
9      event log (string _msg);
10     event winLog (address _win);
11     event newBallot (address _addr);
12
13     function AwardToken () {
14         quantity = 100;
15     }
16
17     function getPreviousWinners() constant returns (address[]) {
18       return prevWinners;
19     }
20
21     // either a name change or it works fine without it
22     // function approve(address spender, uint256 value) public returns (bool);
23     function startRound() onlyOwner canMint public returns (bool) {
24         // if this is the first minting then we should let this go immediately
25         if (address(currBallot) == 0x0) {
```

[2] only remix transactions, script | Search transactions

Compile | Run | Settings | Analysis | Debugger | Support | Test

Start to compile | Auto compil | Hide warnings

AwardToken

Details | Publish on Swarm | ABI | Bytecode

Static Analysis raised 38 warning(s) that requires your attention. Click here to show the warning(s).

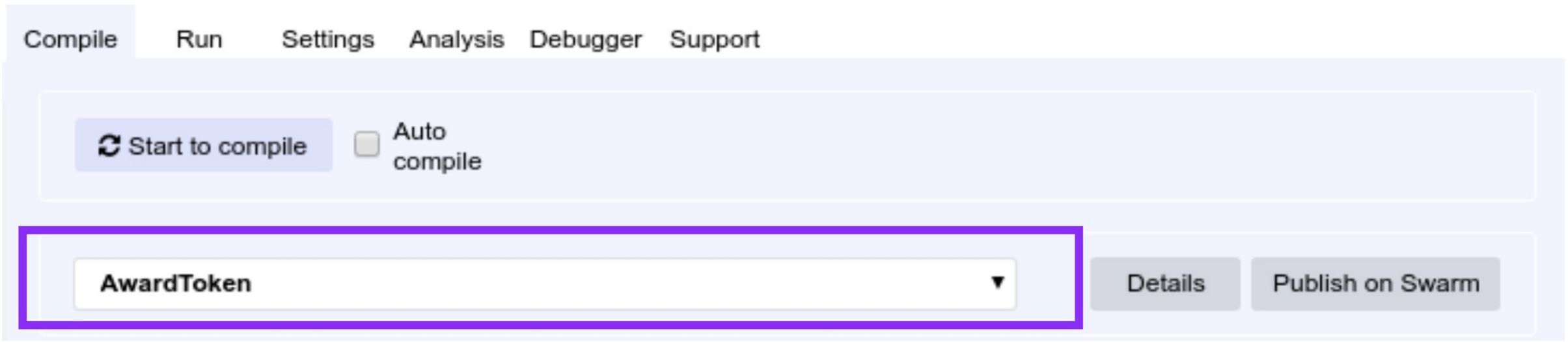
```
gist/Ballot.sol:26:5: Warning: Defining constructors as func
    function Ballot(uint duration) public {
    ^ (Relevant source part starts here and spans across mul
```

```
gist/AwardToken.sol:13:5: Warning: Defining constructors as
    function AwardToken () {
    ^ (Relevant source part starts here and spans across mul
```

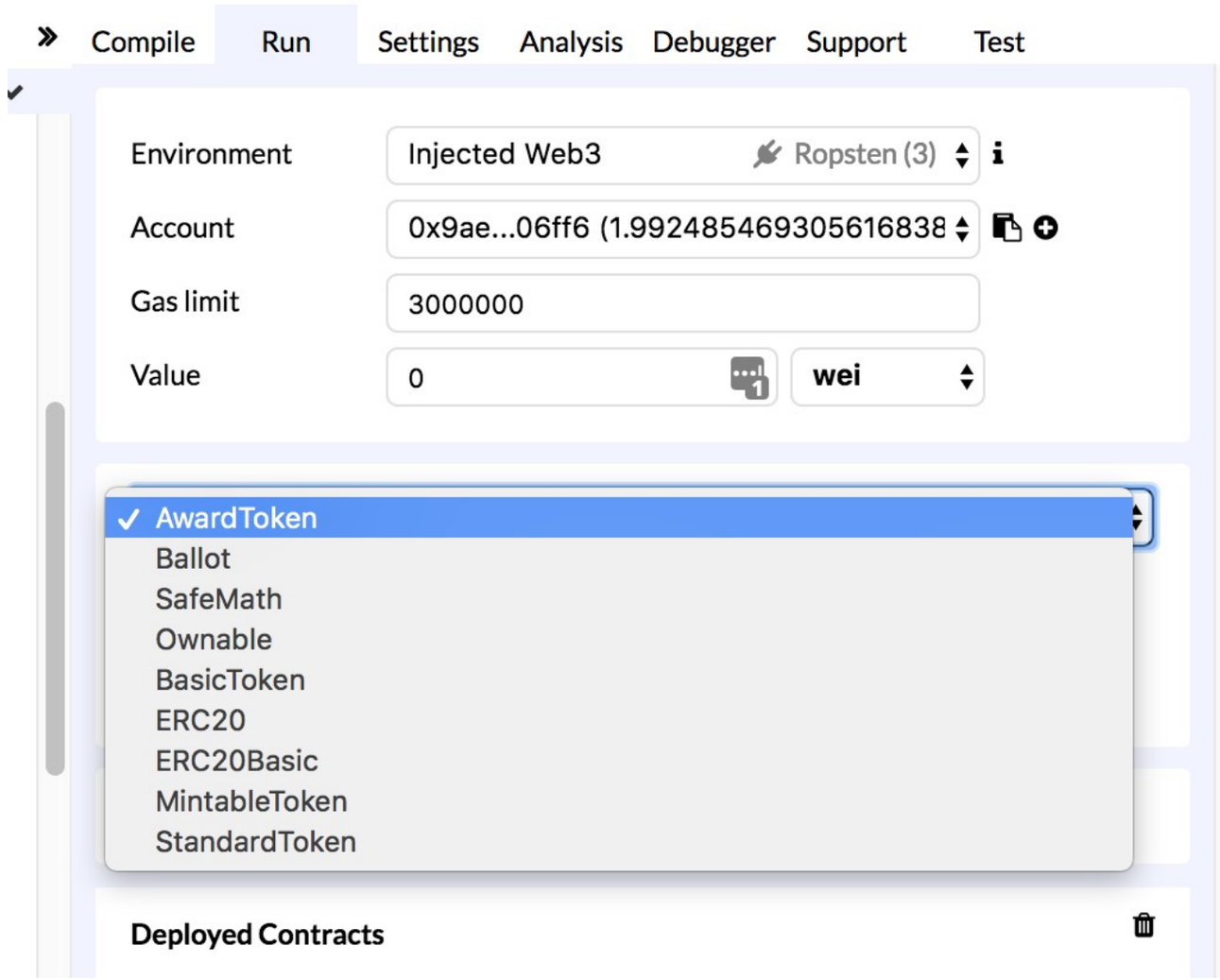
```
gist/AwardToken.sol:1:1: Warning: Source file does not speci
```

See compiled contracts

AwardToken + all dependencies

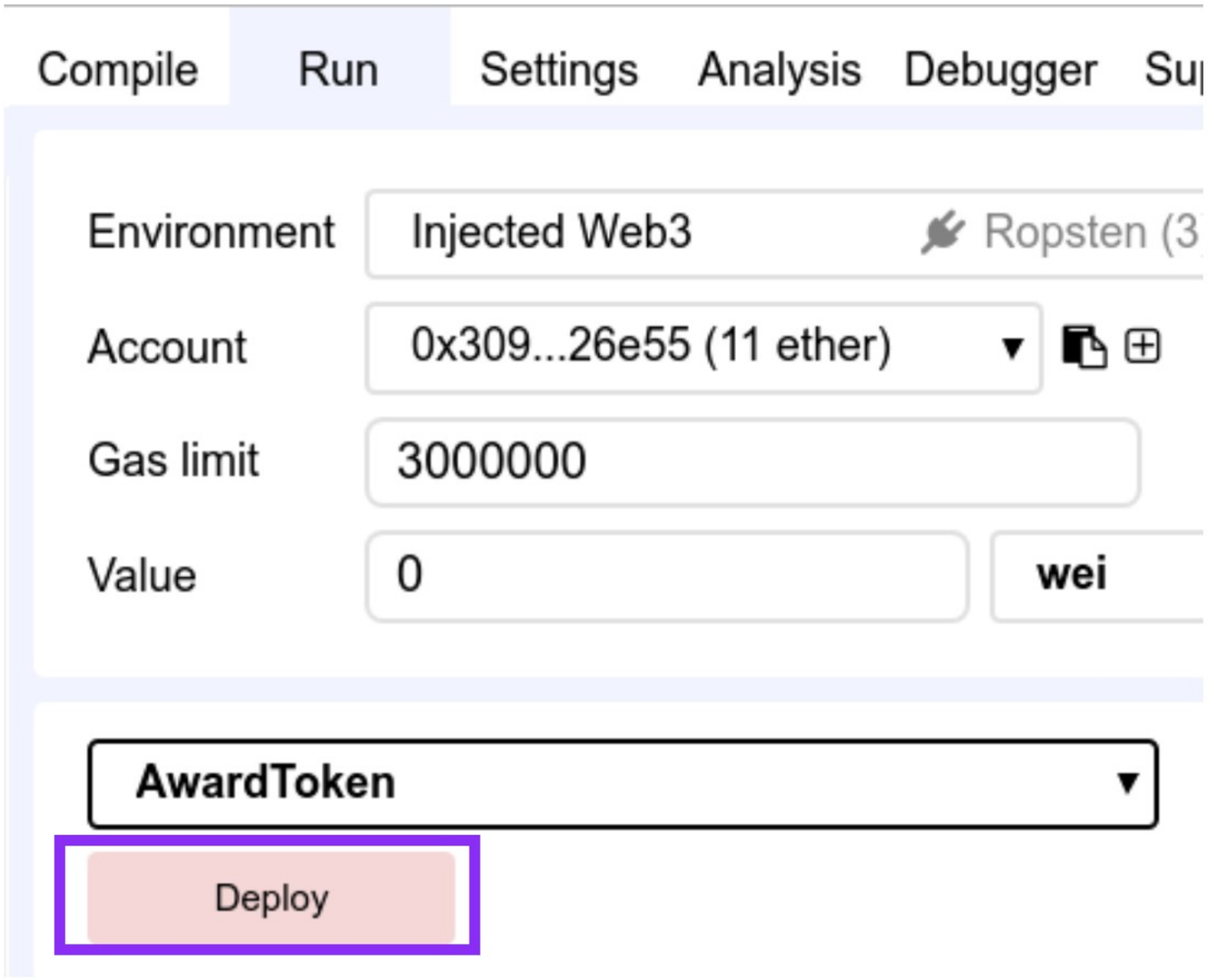


Imported Contracts



Deploy the contract

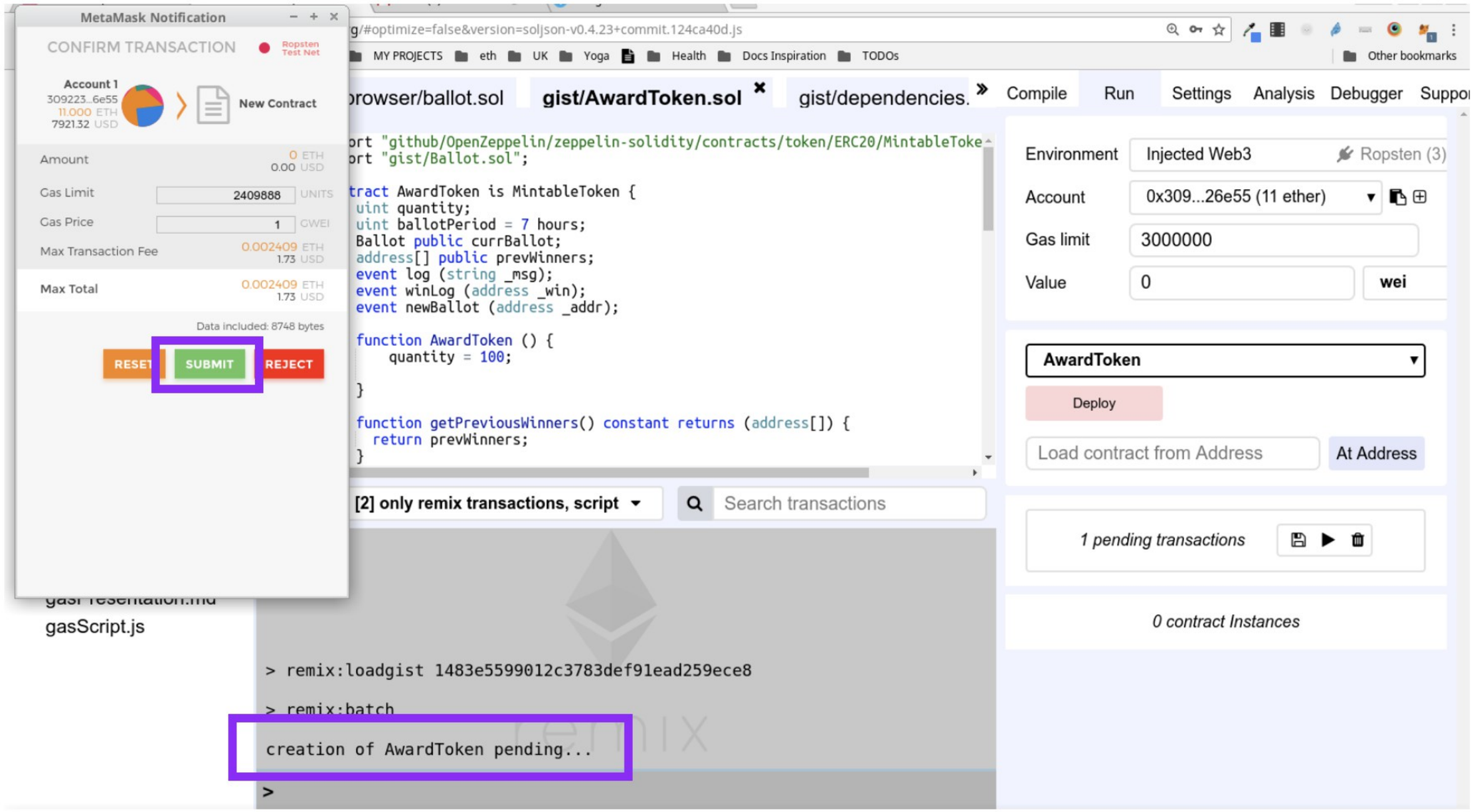
Run tab: Deploy button



Confirm the transaction

Submit button

But make sure you put in a gas price!



Check if tx is mined

Terminal logs in Remix

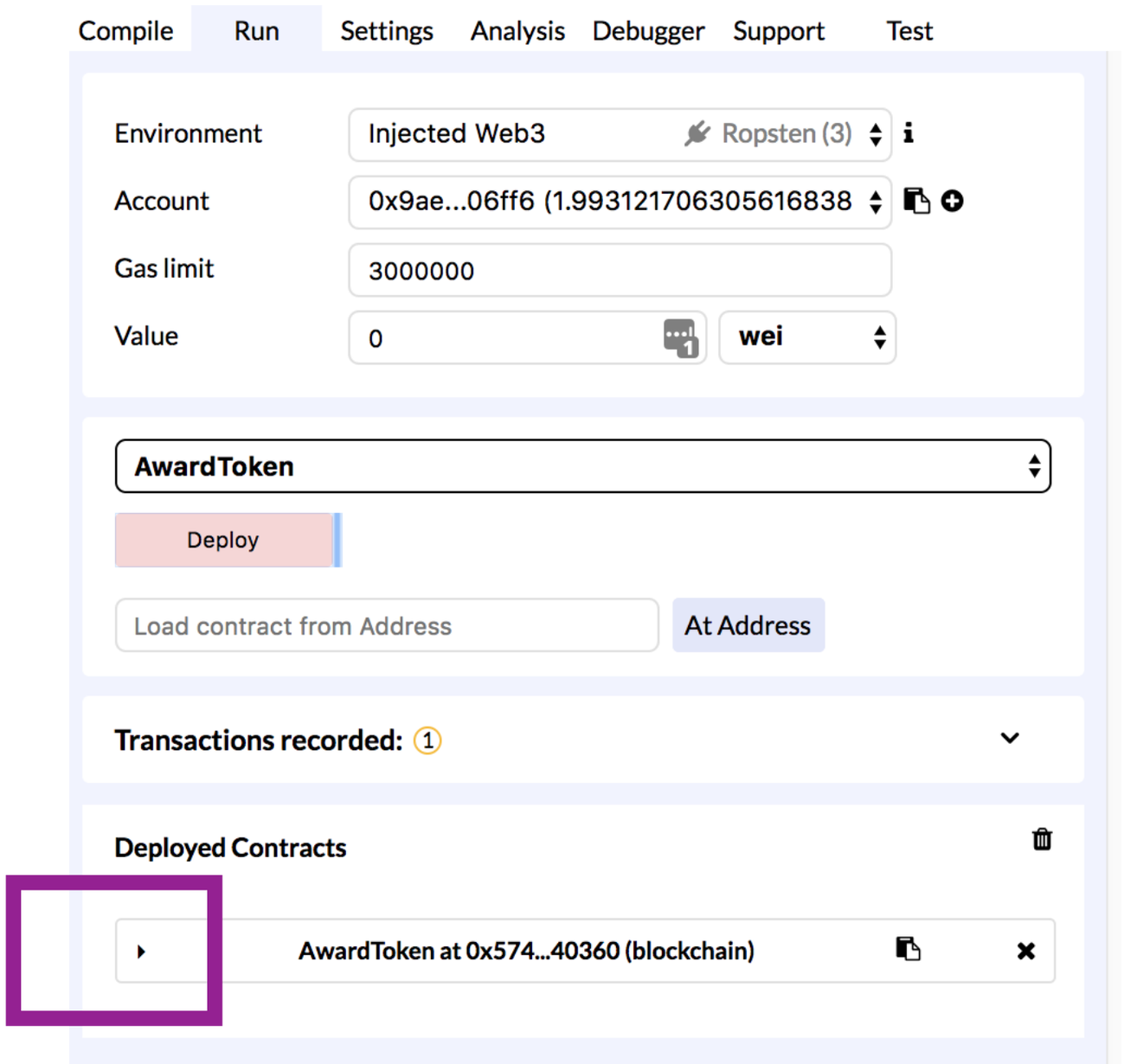
```
creation of AwardToken pending...

https://ropsten.etherscan.io/tx/0x404a4445ebb3a969b15257a586a61582afa07
dcf02b1b2617f77519b30378be8
```

```
▶ [block:3159099 txIndex:2]  from:0x309...26e55
to:AwardToken.(constructor) value:0 wei data:0x608...70029
logs:0 hash:0x404...78be8
```

Debug

Click to see the contract's UI

On the deployed contract



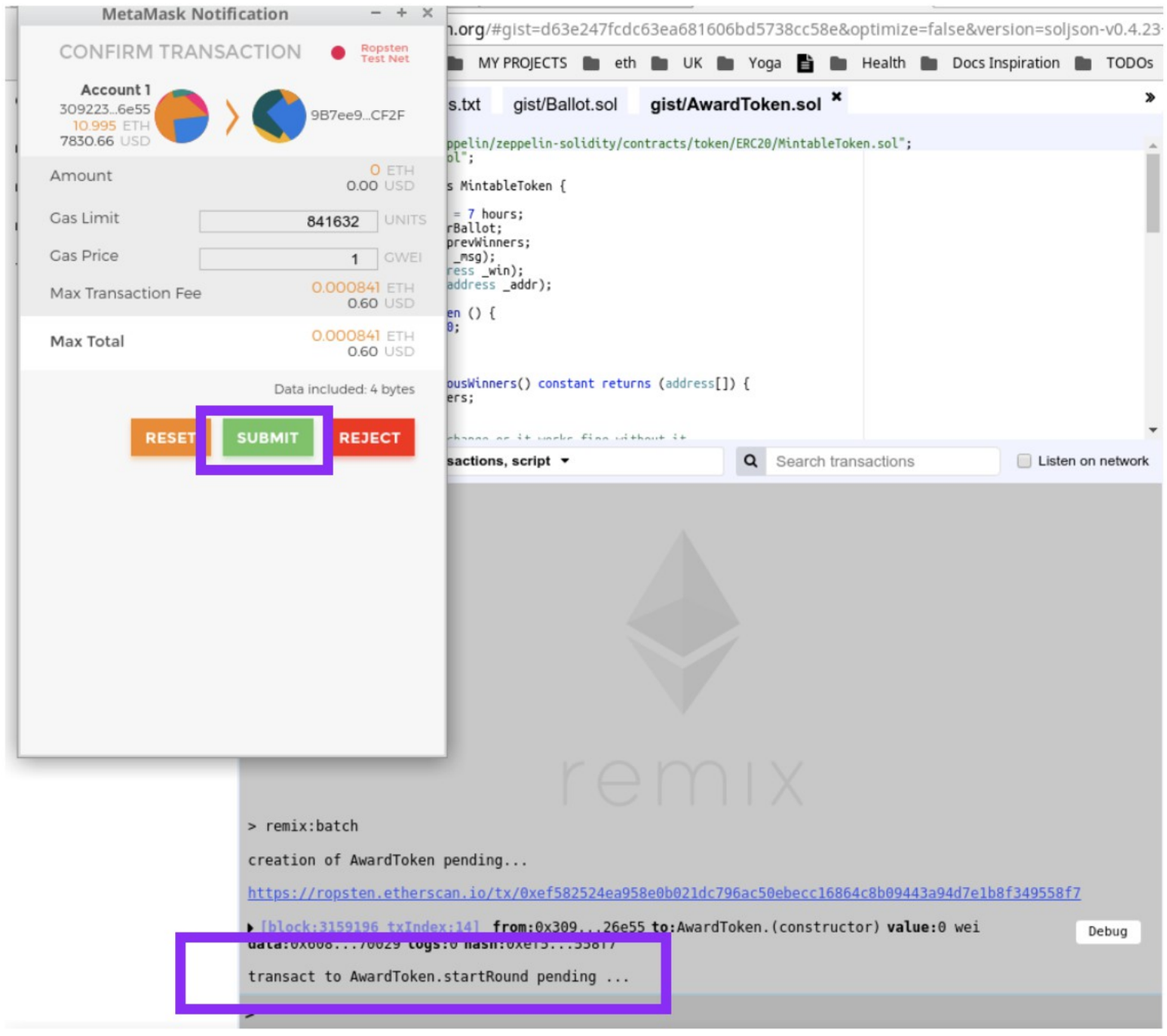
Behold!

The Interactive UI for AwardToken.sol contract

Execute startRound

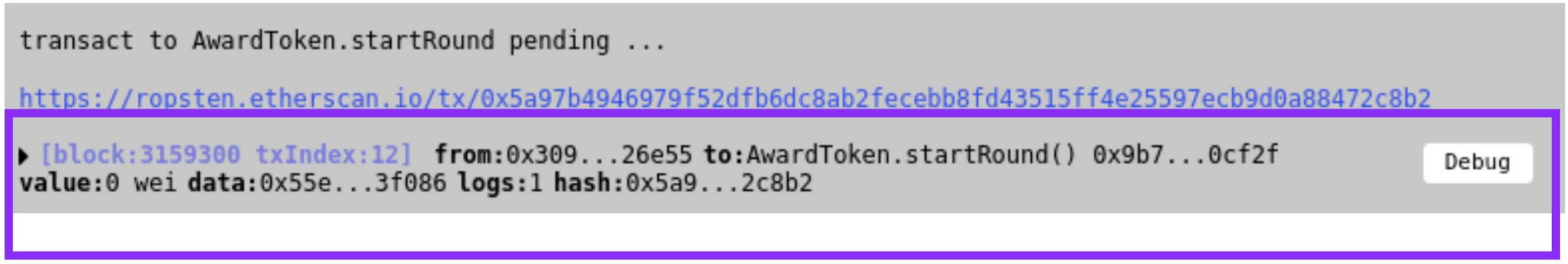
Its a payable function
(as opposed to a call function - which is free)

Confirm the transaction



Check if tx is mined

In the terminal logs in Remix



Expand tx log

to see the logs

[block:3665523 txIndex:4] **from:**0x9ae...06ff6 **to:**AwardToken.startRound() 0x574...40360 **value:**0 wei **data:**0x55e...3f086 **logs:**1 **hash:**0x16c...0a81c

Debug

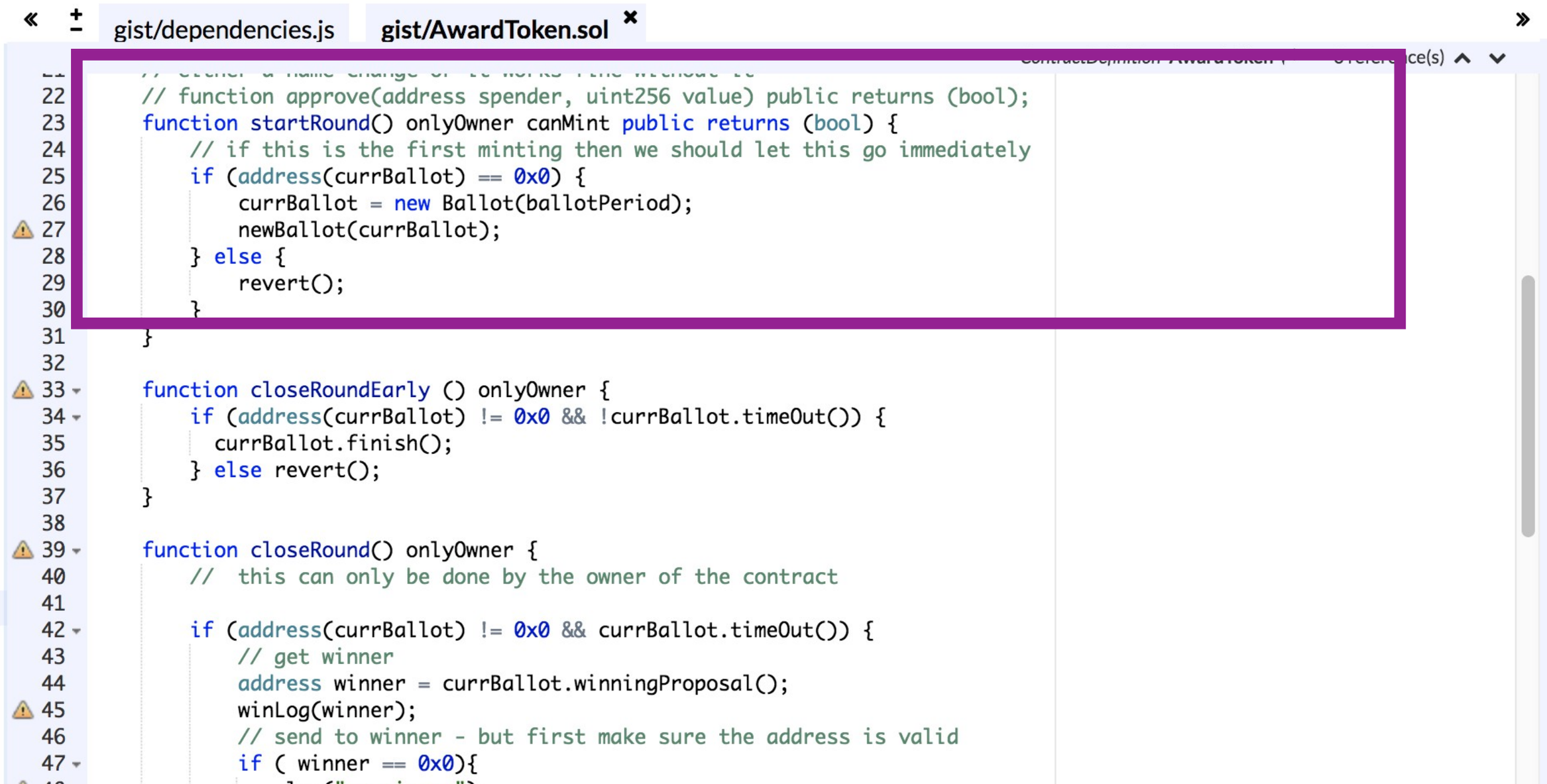
status	0x1 Transaction mined and execution succeed
transaction hash	0x16c8af5a3fd0e5bcacd8858ab42d4f8eff39fc33bb98290740c03eeb4880a81c
from	0x9ae59af2e33480caa48f2dc6f6cede7ffab06ff6
to	AwardToken.startRound() 0x574d270dc04e89c5d65e24e19f1deb9e17240360
gas	613643 gas
transaction cost	613643 gas
hash	0x16c8af5a3fd0e5bcacd8858ab42d4f8eff39fc33bb98290740c03eeb4880a81c
input	0x55e...3f086
decoded input	{}
decoded output	-

logs	[
	{
	"from": "0x574d270dc04e89c5d65e24e19f1deb9e17240360",
	"topic": "0x65f35fb257c91daed794331bfd2ad0f4439d49319d52a5b3bfb04c84969fdbeb",
	"event": "newBallot",
	"args": {
	"0": "0xD6052C85A3D26eE9EeC8262d462bfDC672B80D93",
	"_addr": "0xD6052C85A3D26eE9EeC8262d462bfDC672B80D93",
	"length": 1
	}
	}
	]

value	0 wei

Take a look at the startRound function in the editor

– just to see what it is doing...



Get ballot's address

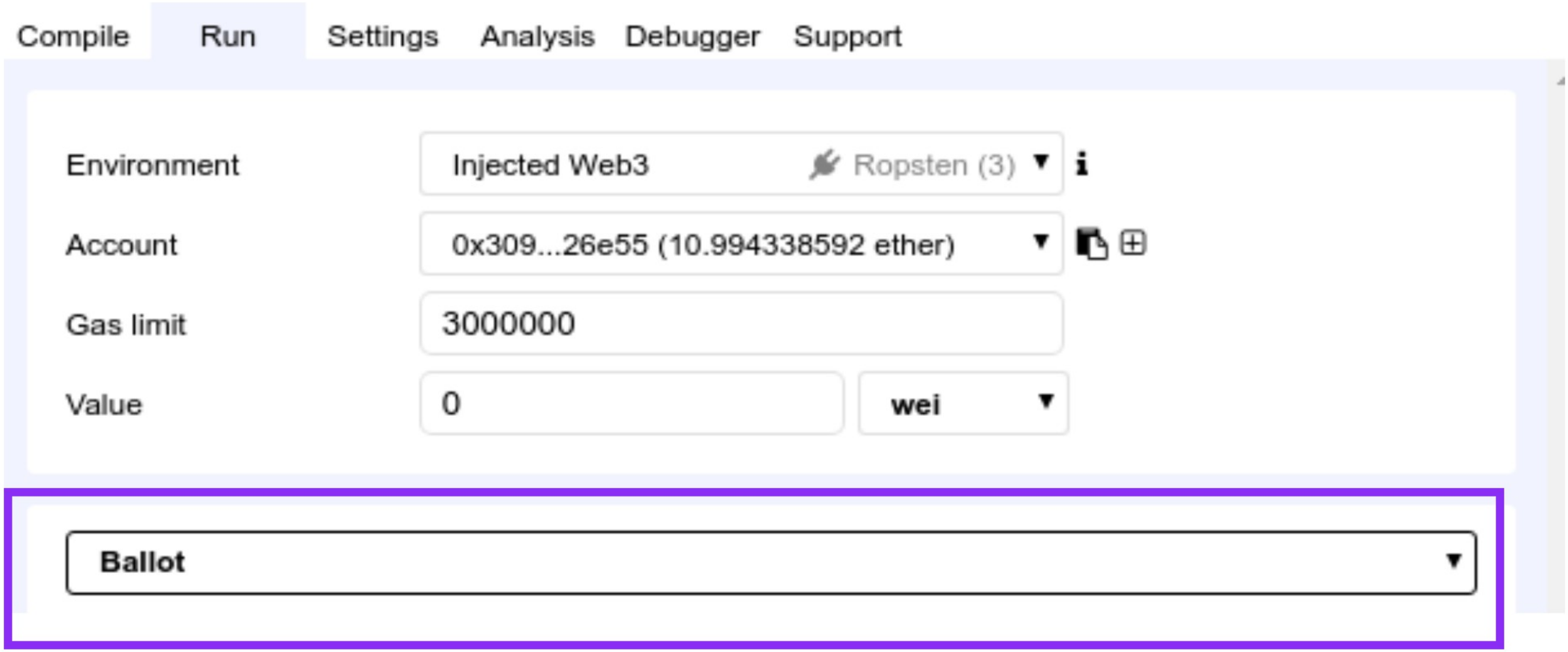
Execute currBallot call

Copy ballot's address

currBallot output

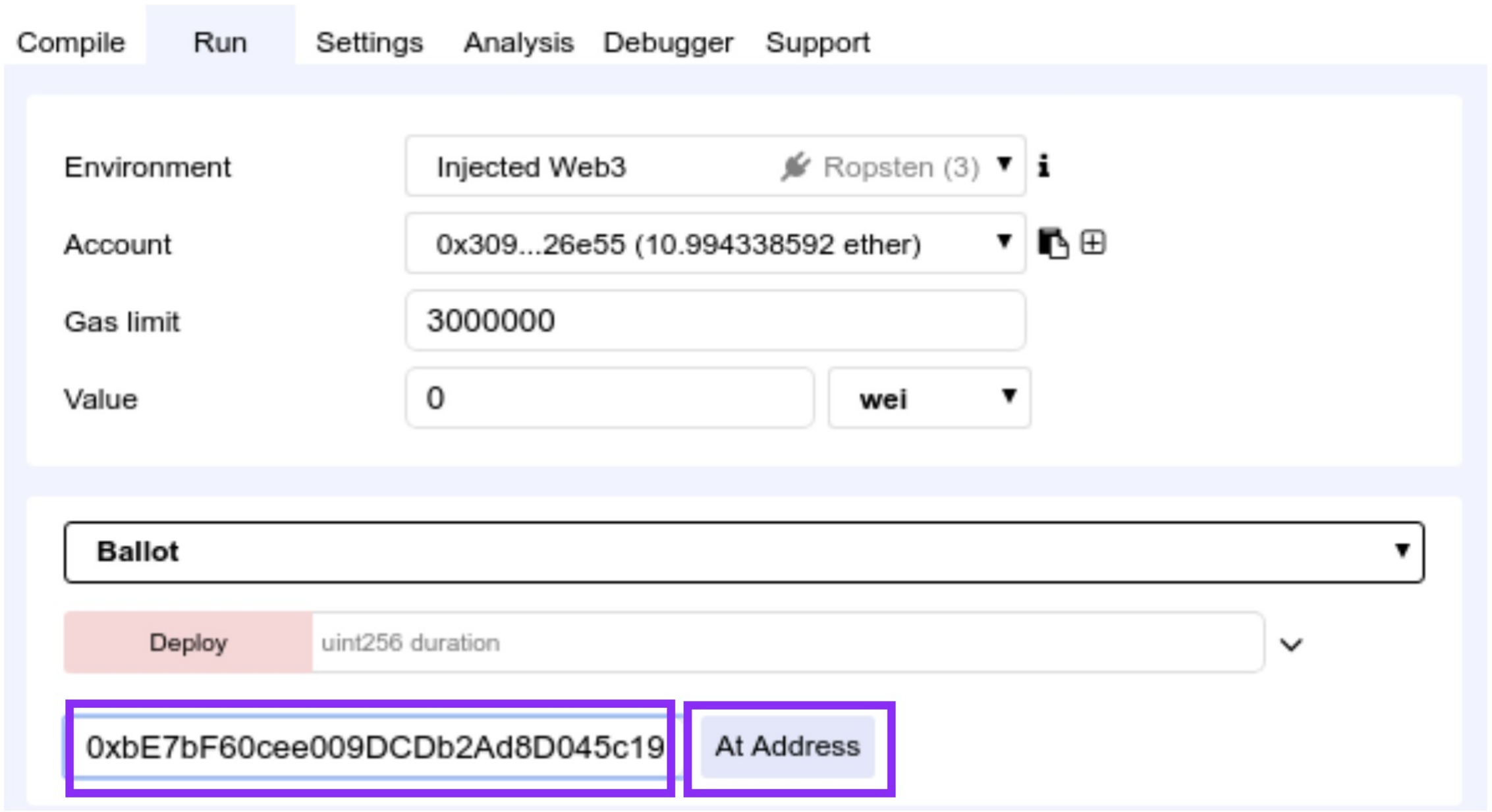
Switch to Ballot

Run tab: dropdown



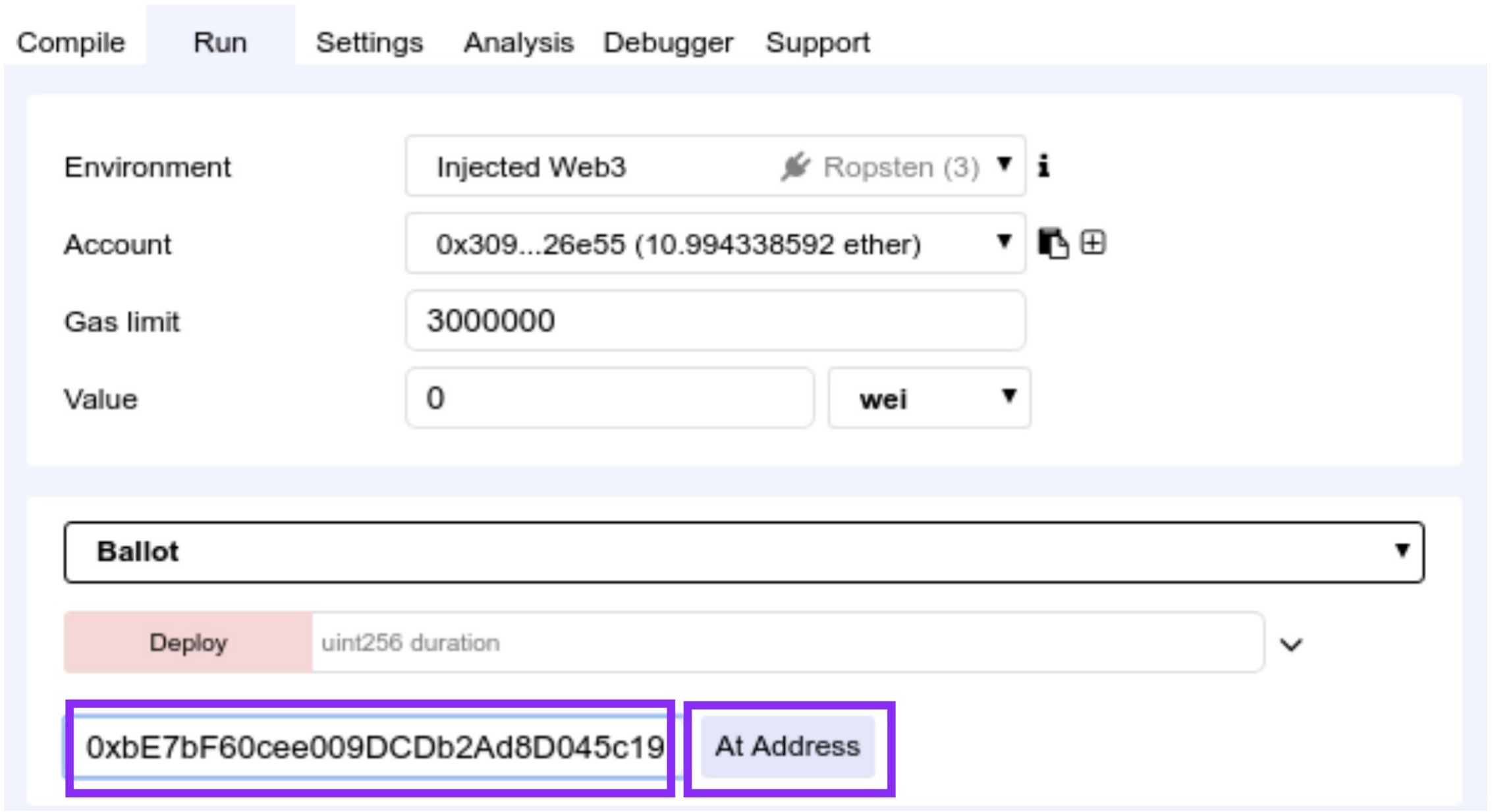
Access Ballot contract

Paste address + click At Address



Access Ballot contract

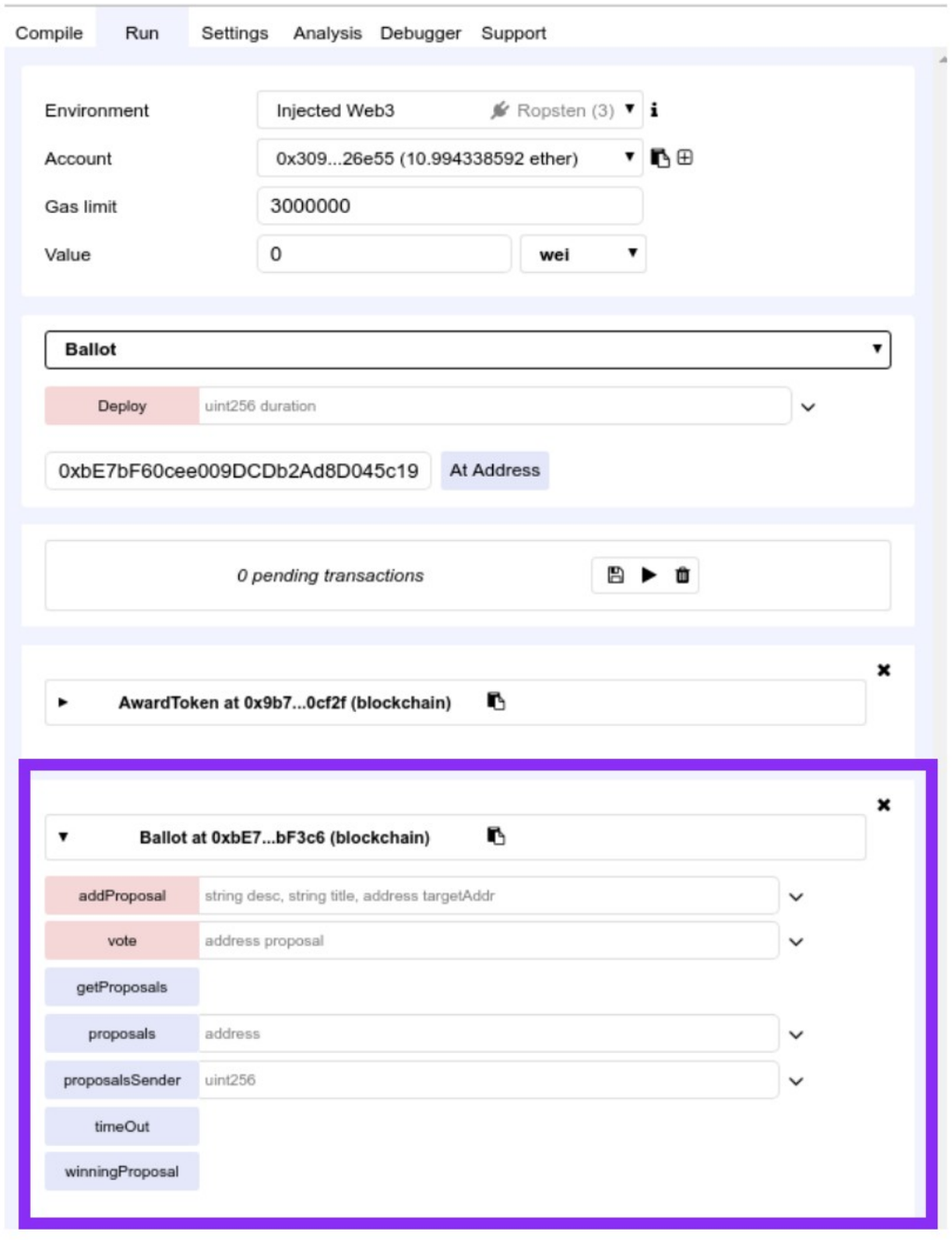
Paste address + click At Address



See autogenerated UI

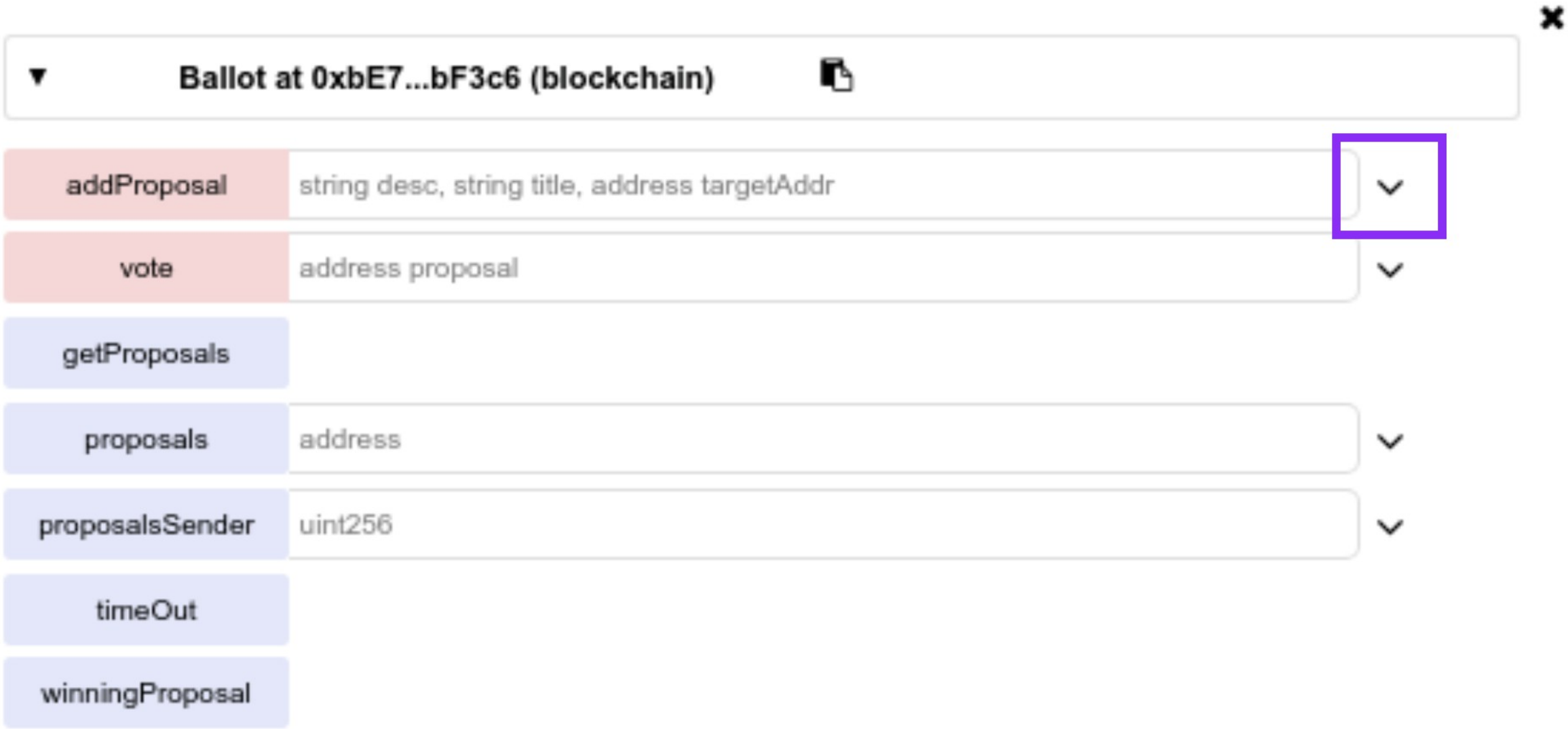
(you might need to scroll down)

- Interact with all the functions in the contract
- And all the functions in the inherited contracts



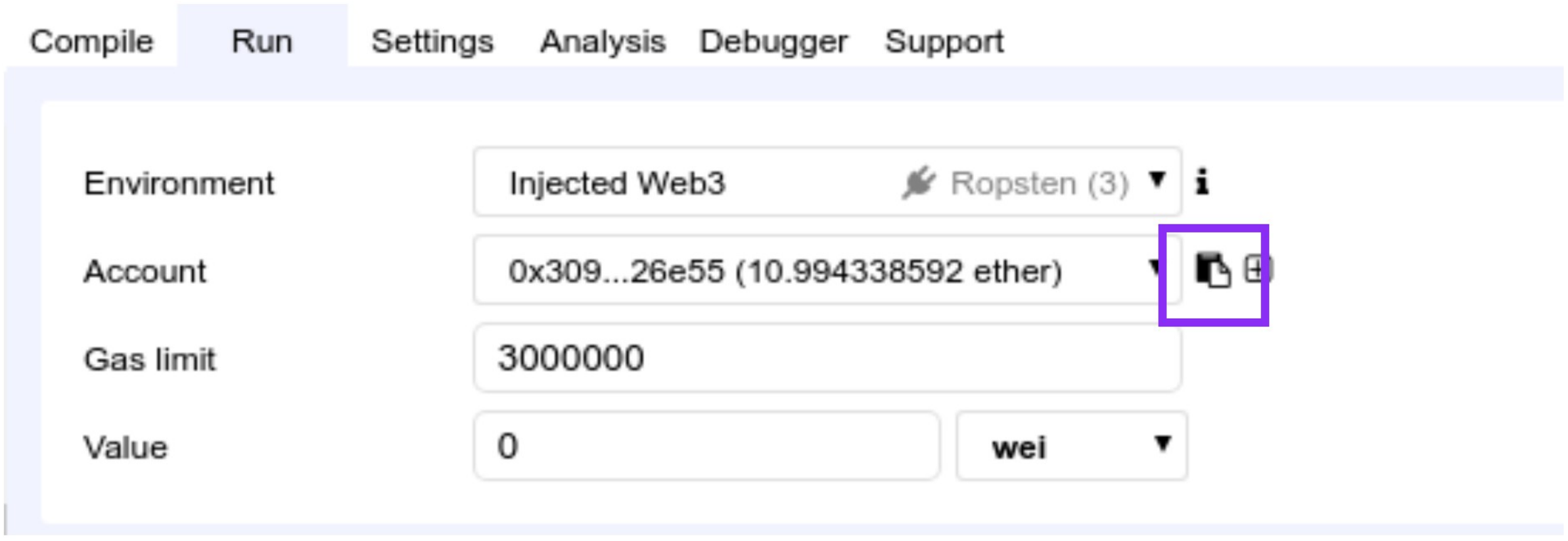
Add a new proposal

Expand addProposal function



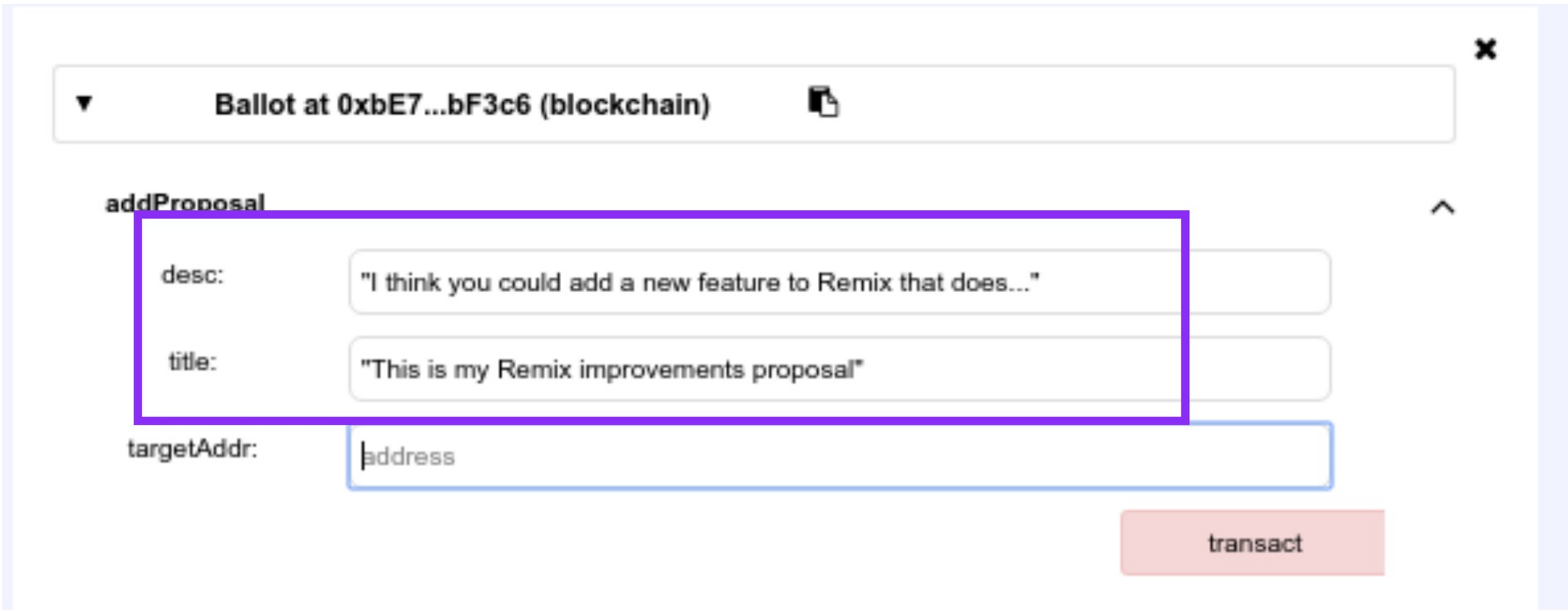
Copy your address

Run tab: Account



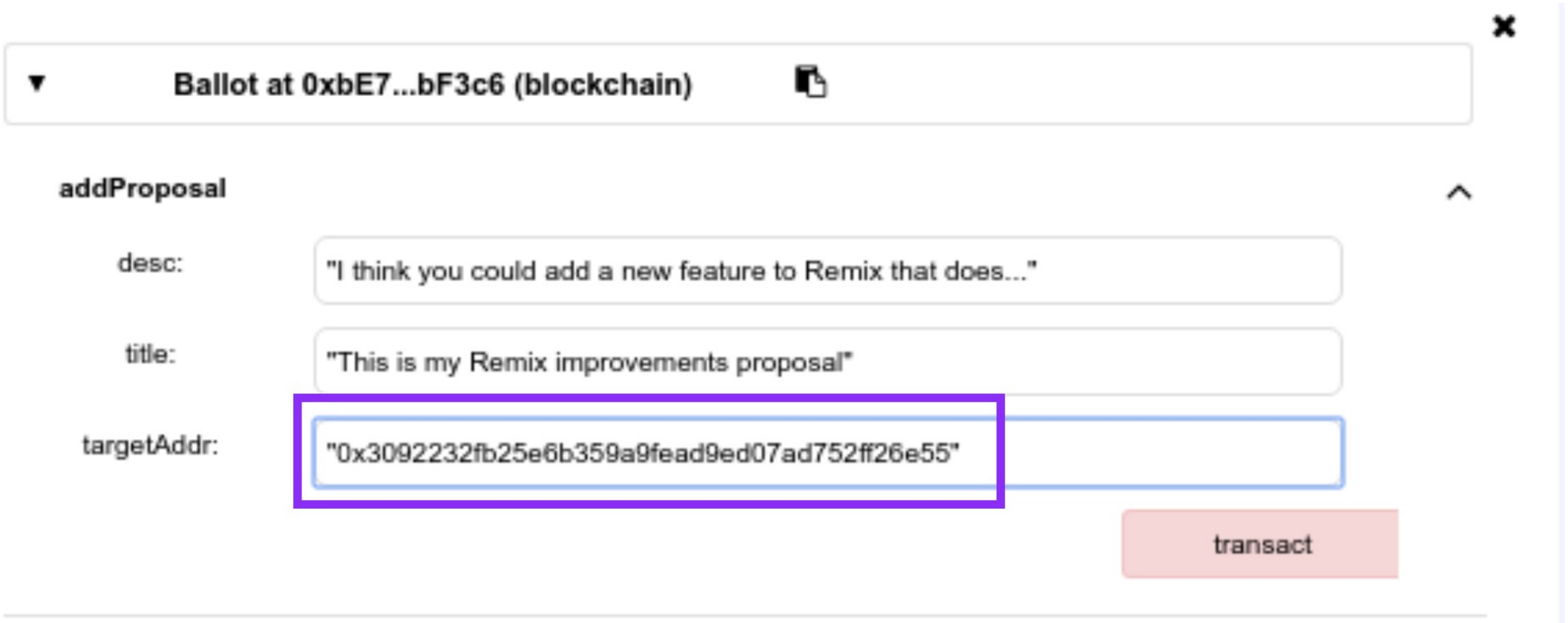
Type a proposal

Run tab: Account



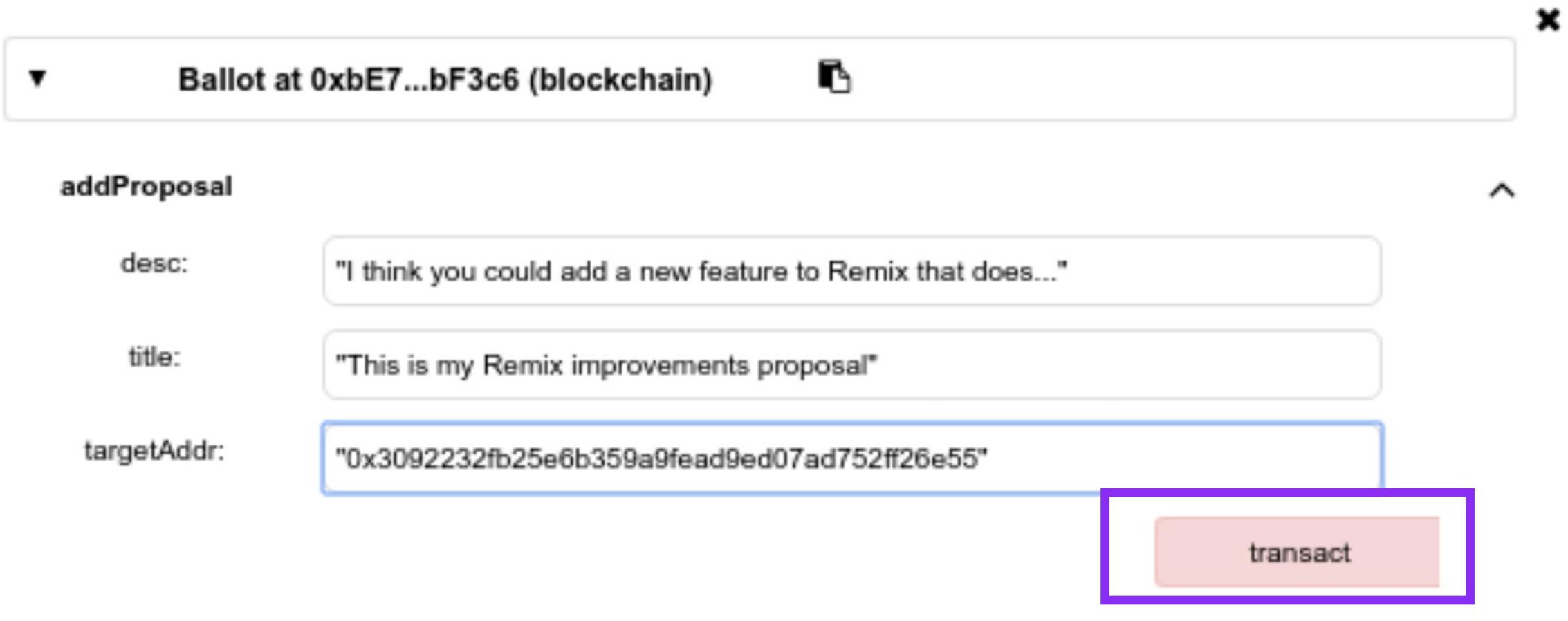
Add your address

Paste the address



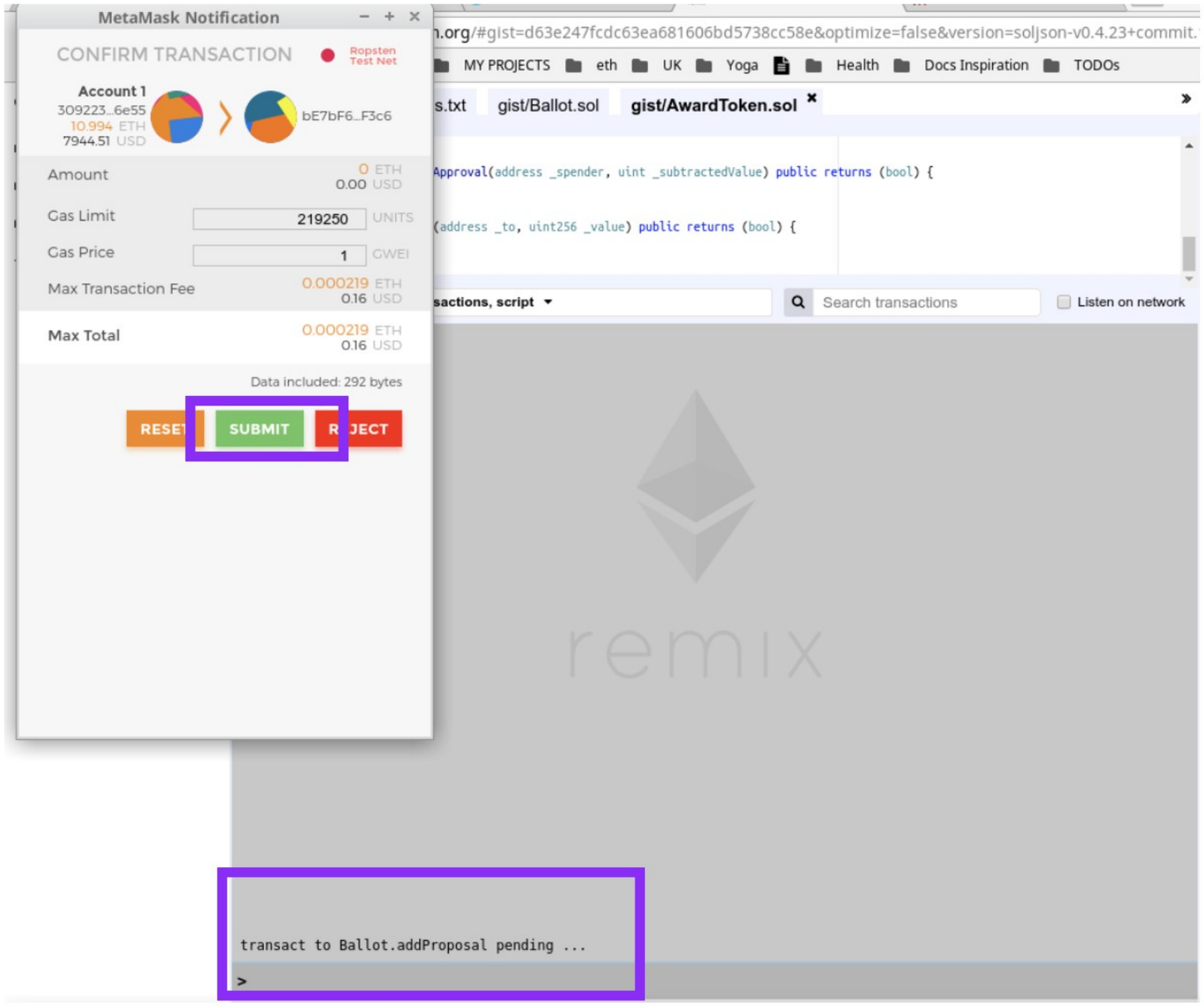
Execute addProposal

Hit the transact button



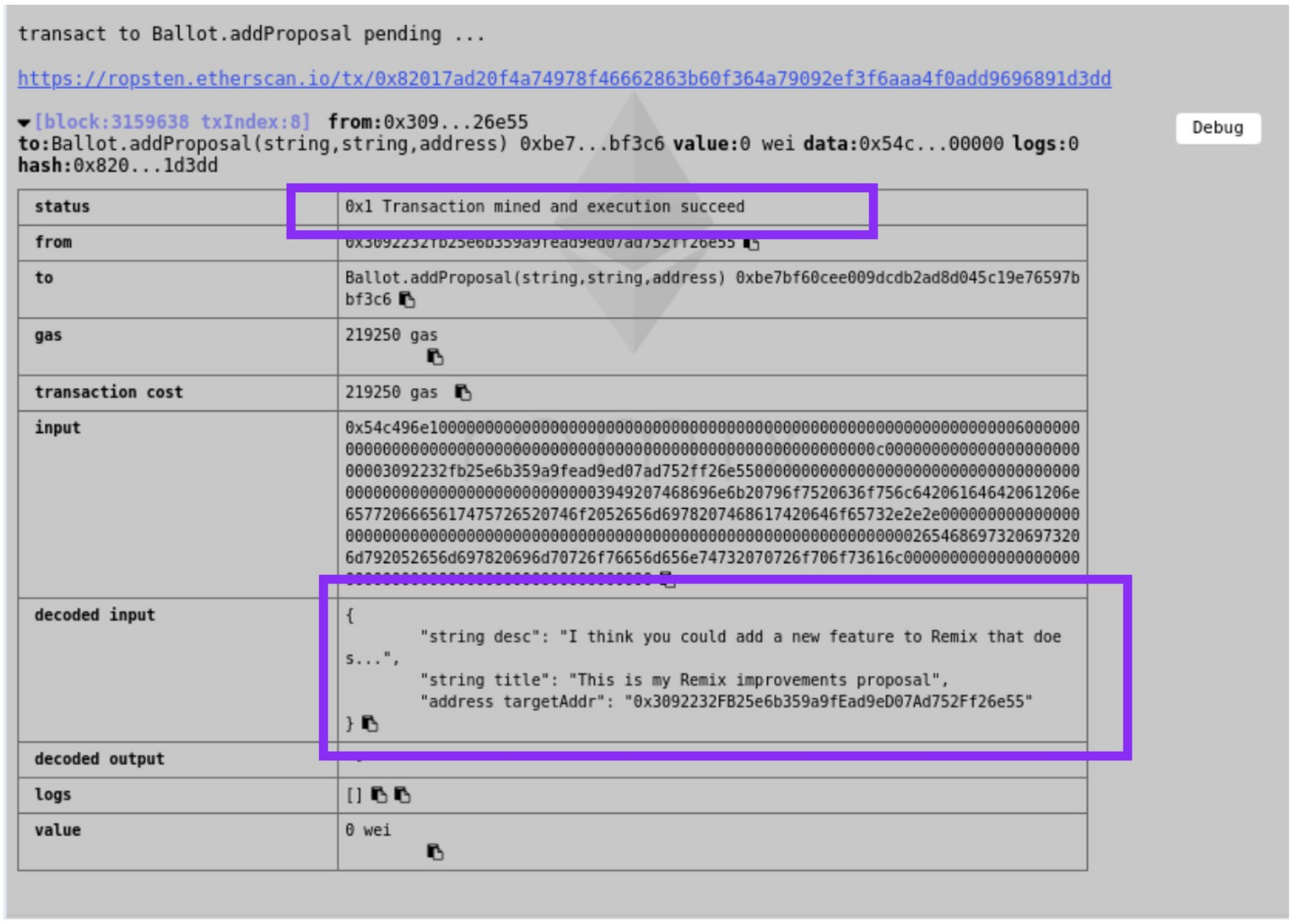
Confirm the transaction

Submit button



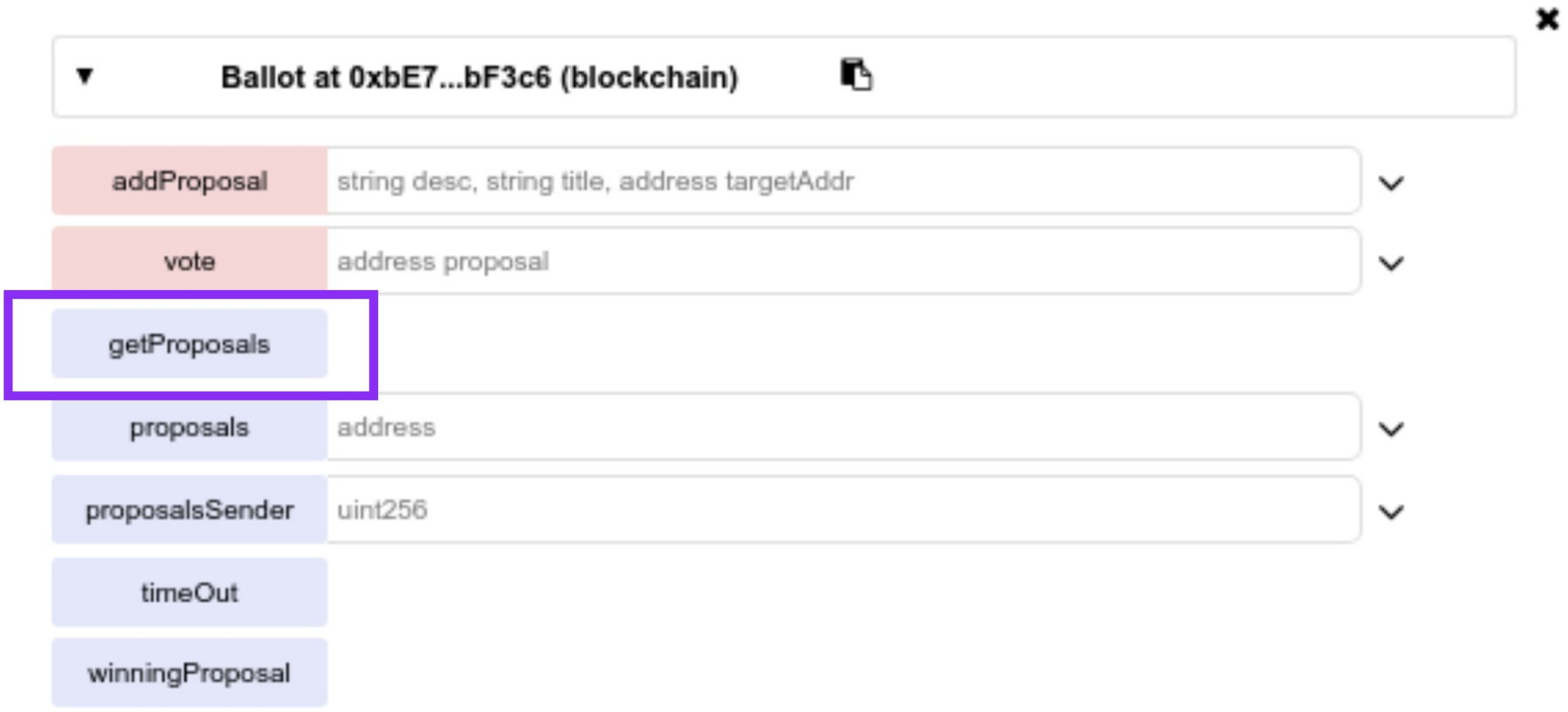
Check if tx succeeded

Terminal logs in Remix



Execute getProposals

getProposals call



See Proposals Addresses

well in so far there will only be 1 address

call to Ballot.getProposals

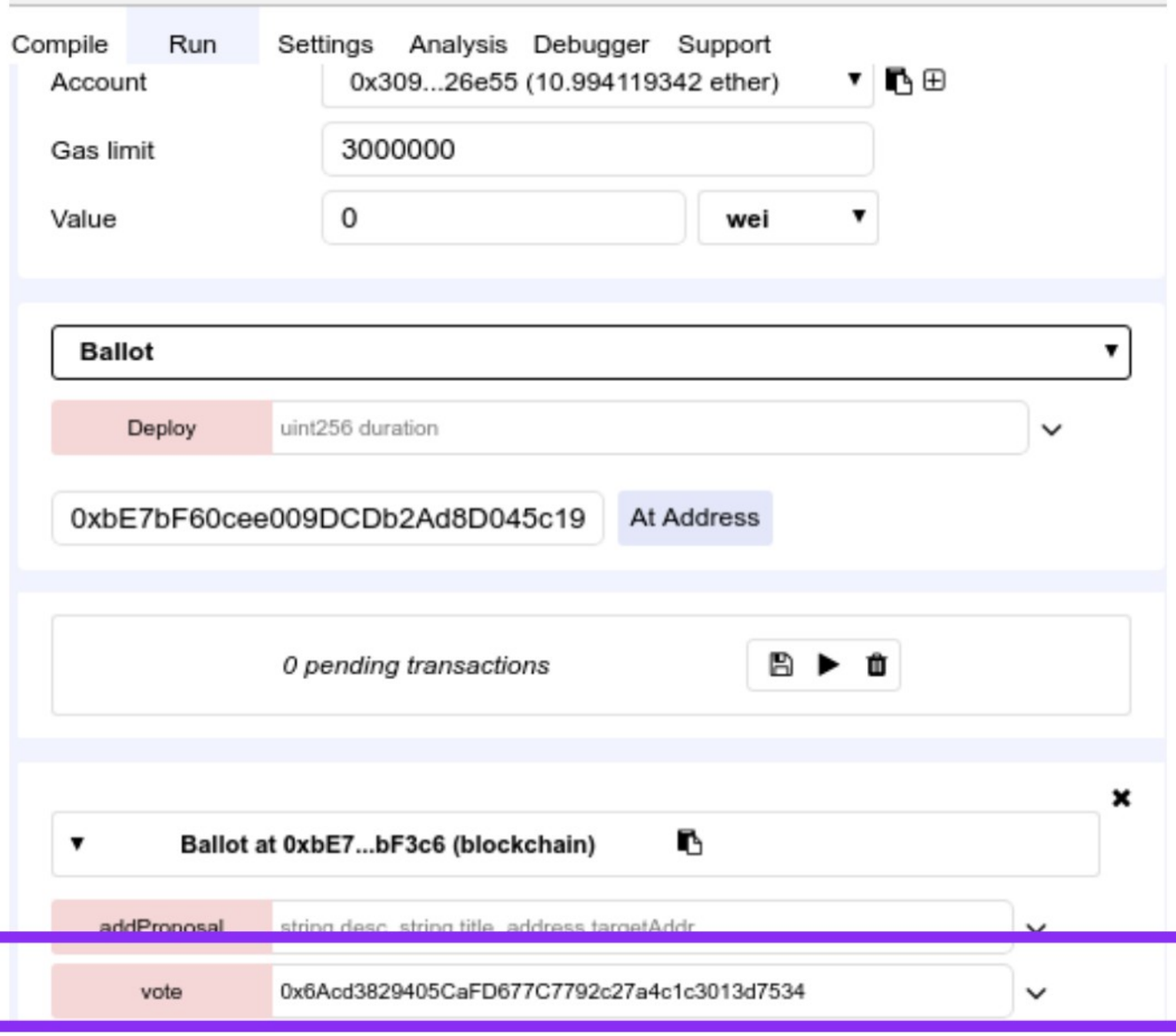
▼[call] **from:**0x3092232fb25e6b359a9fead9ed07ad752ff26e55 **to:**Ballot.getProposals()
data:0x625...64c48

Debug

from	0x3092232fb25e6b359a9fead9ed07ad752ff26e55
to	Ballot.getProposals() 0xbE7bF60cee009DCDb2Ad8D045c19e76597bbF3c6
input	0x62564c48
decoded input	{}
decoded output	{ "0": "address[]: 0x3092232FB25e6b359a9fEad9eD07Ad752Ff26e55,0xFd0f51afb685Cd8735AfE7685D21355589602b8c,0x6Acd3829405CaFD677C7792c27a4c1c3013d7534" }
logs	[]

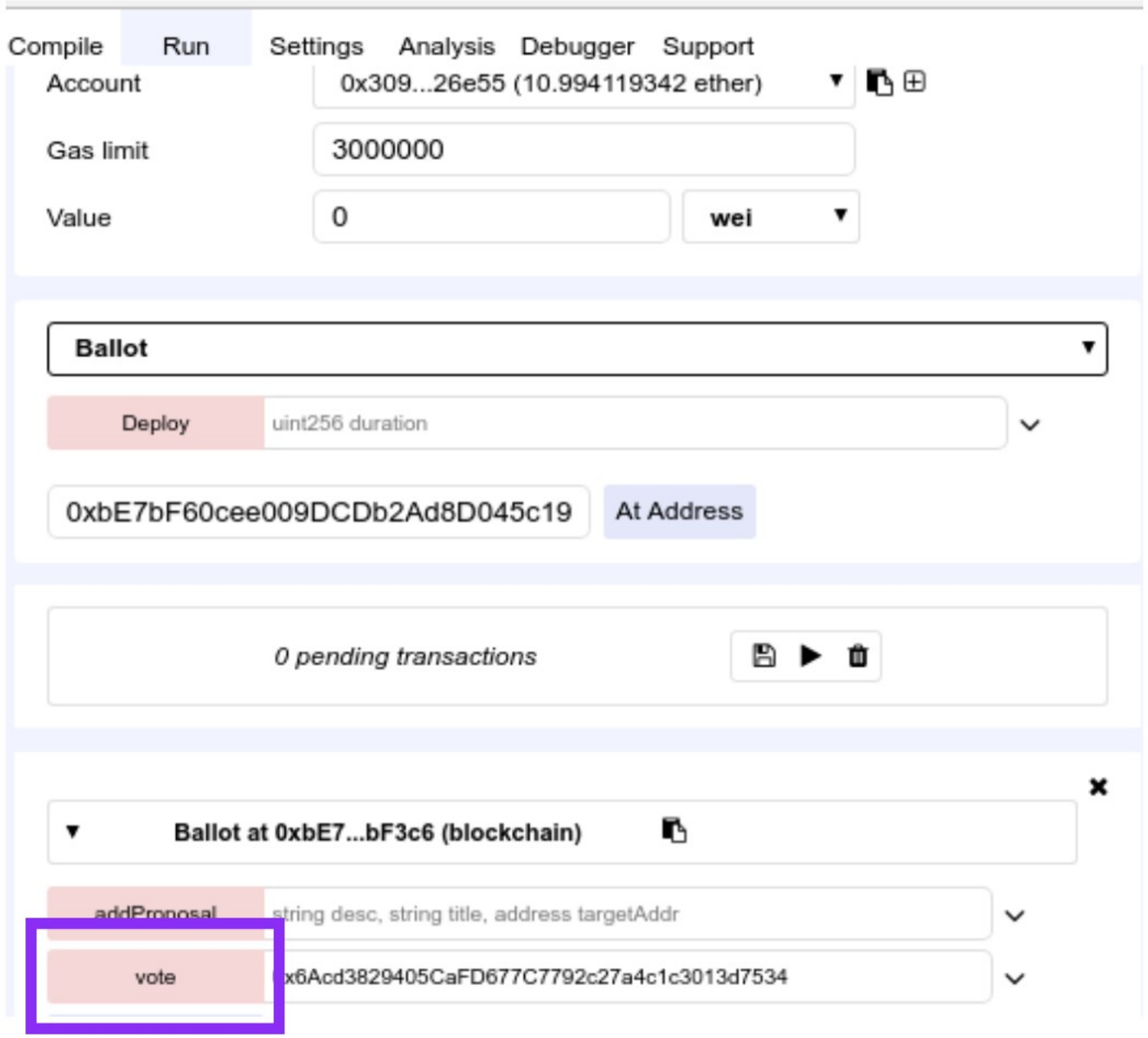
Vote for one Proposal

Paste Proposal Address you want to vote for



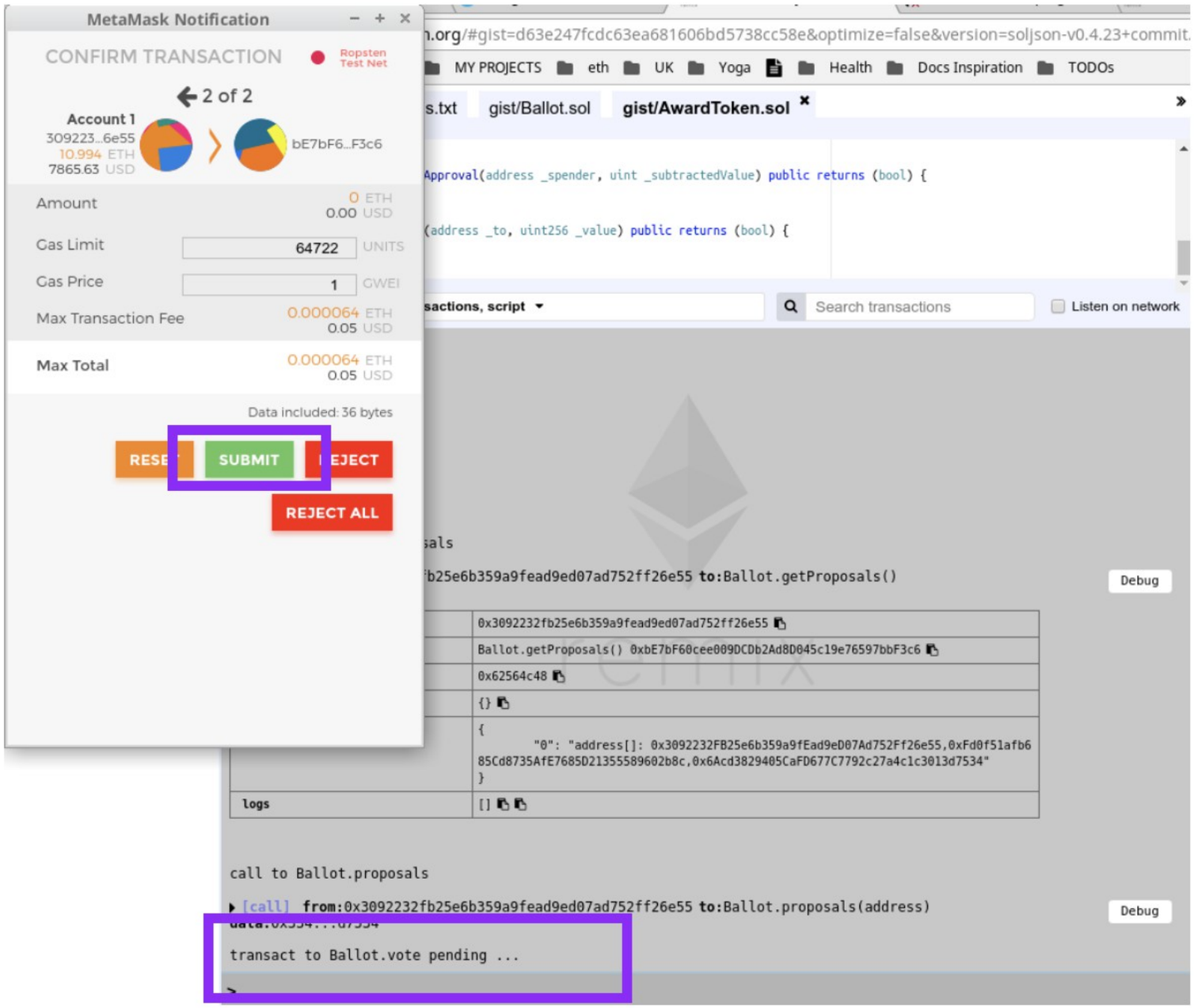
Execute vote transaction

vote button



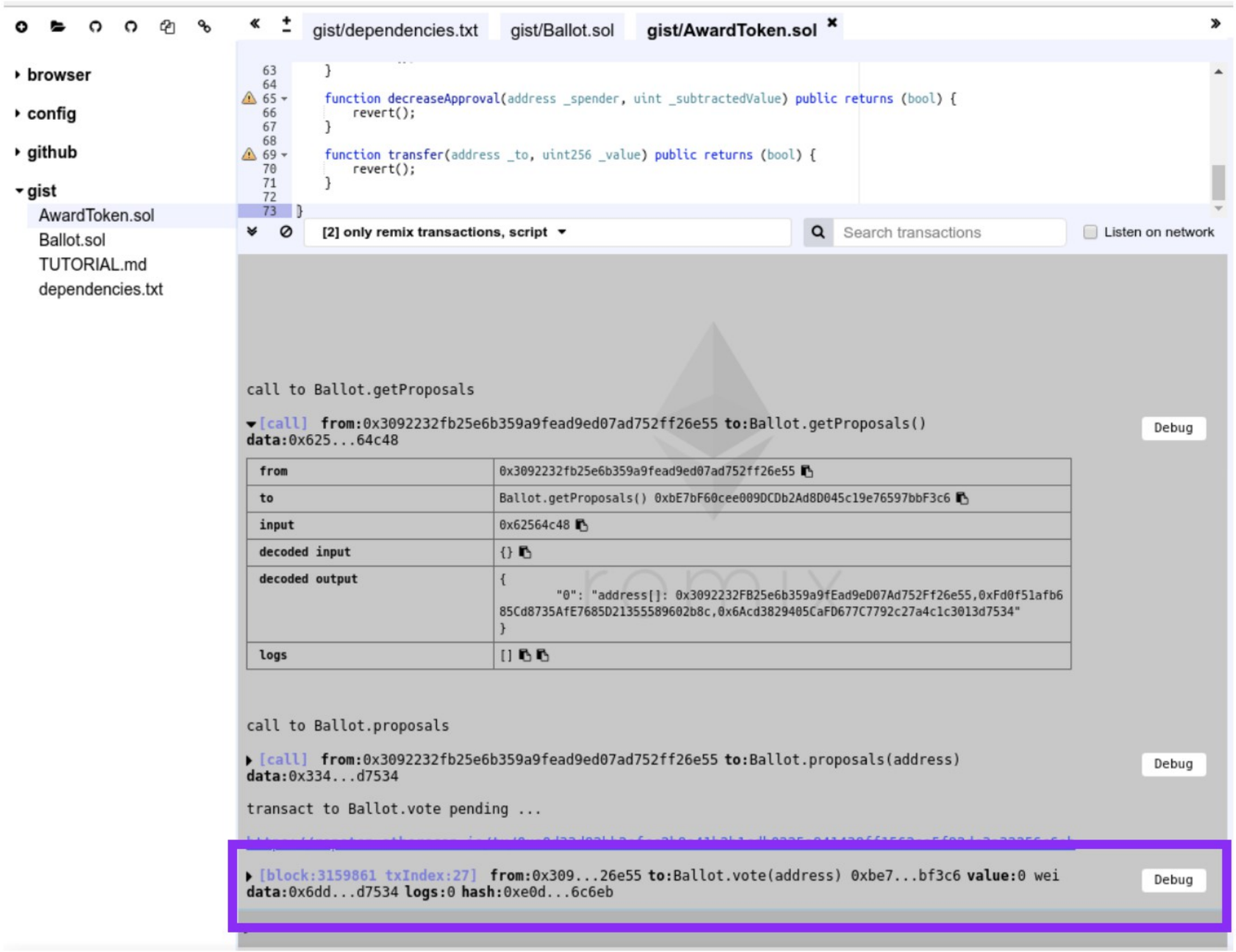
Confirm the transaction

Submit button



Check if tx succeeded

Terminal logs in Remix



Now let's try it out connecting a frontend

http://bit.ly/remix-voting

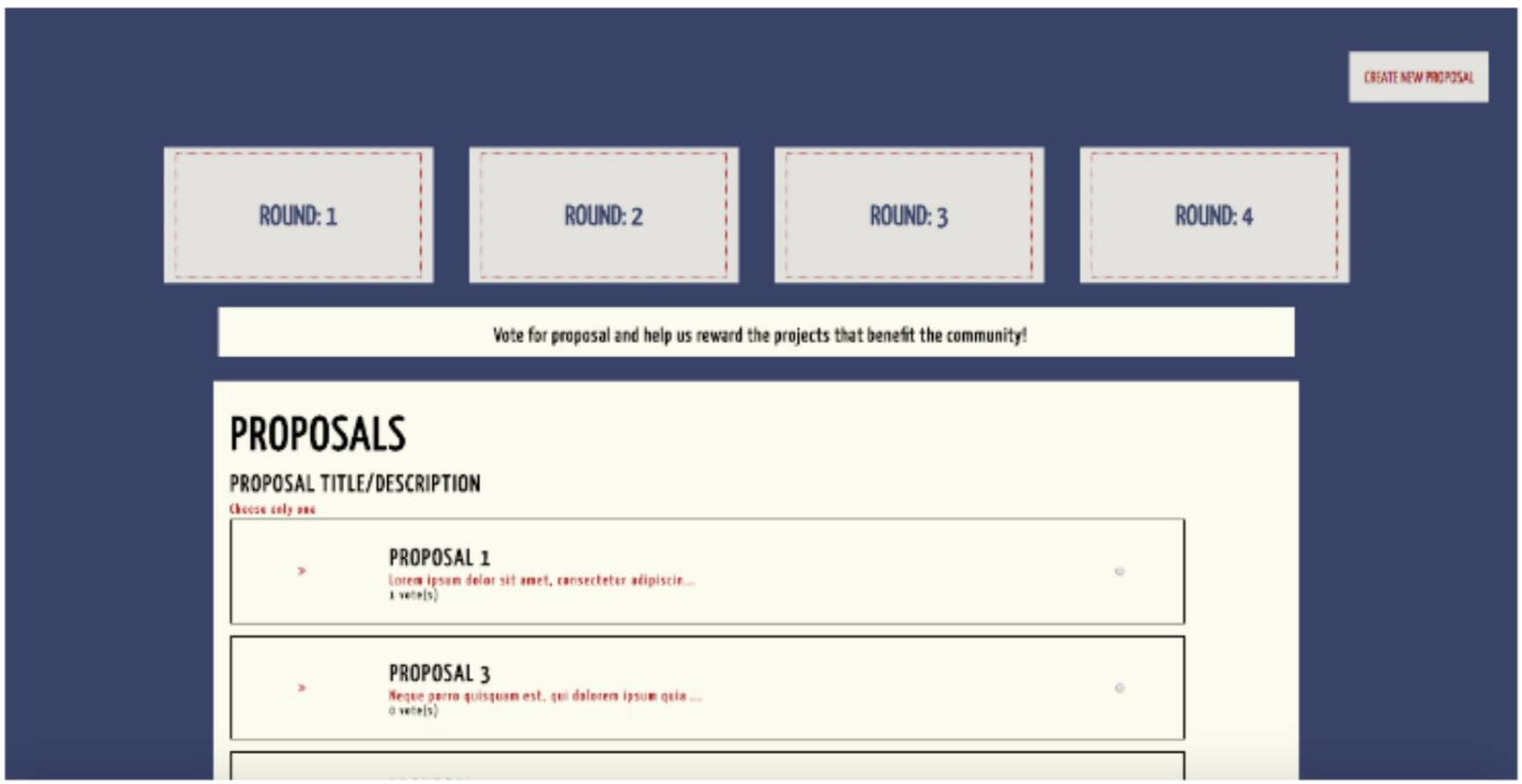
(when dependencies.js is the active file)

To access our Award Token from this frontend - you need the address of the Award Token.

Go to ethereum/remix-workshop to access the award token I just deployed

```
contract Ballot {

    uint _duration;
    uint _startTime;
    struct Proposal {
        string description;
        string title;
        uint voteCount;
```

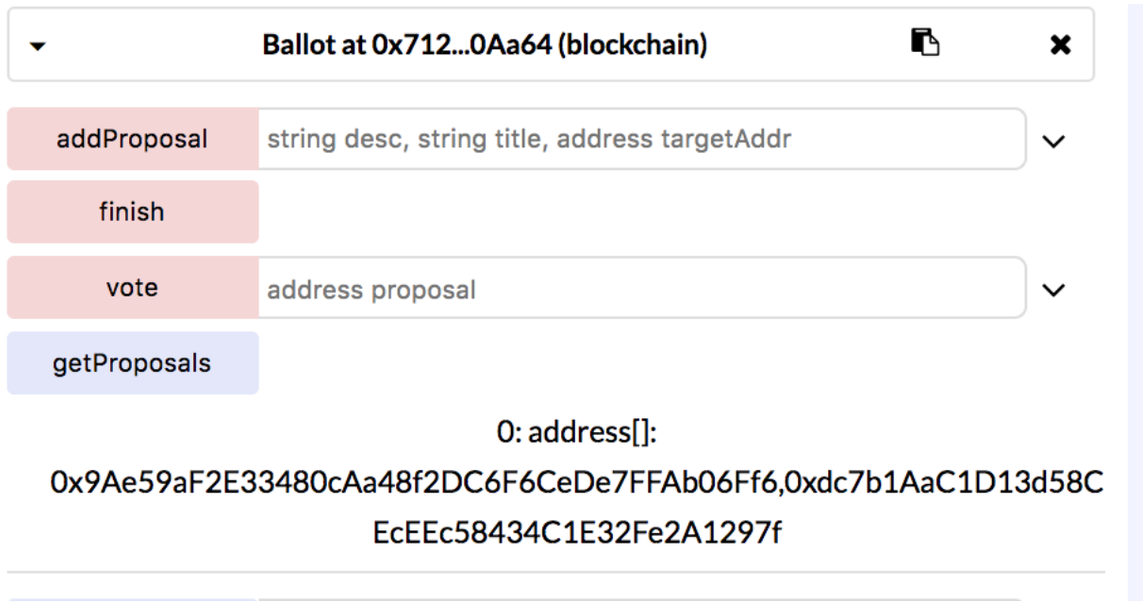
```
contract AwardToken is MintableToken {
    uint quantity;
    uint ballotPeriod = 7 hours;
    Ballot public currBallot;
    address[] public prevWinners;
```



Let's check results

http://bit.ly/remix-voting

Check the state of the contract



2 proposals have been added

Thank you

@ninabreznik @ryestew @yann300
@serapath @LianaHus @iurimatias

http://bit.ly/remix-workshop-repository