Connecting our Ethereum private blockchain and interacting with it.

Tools:

1. A private blockchain: Setup and provided by the university.
2. MetaMask: Wallet.
3. Remix Ethereum: Online Solidity compiler.

Outline

- We show how to connect to our private blockchain and interact with it.
- Steps:

1. Install MetaMask. Create an account (i.e. an address and public-private key) via MetaMask.
2. Send us your account address, so we can give you some Ether.
3. Get familiar with Solidity and the Remix compiler:
   - Write smart contracts, debug and compile them online.
4. Send/deploy the latest version of the contract to the blockchain and interact with the deployed contract.

Step 1:
Install Metamask

- It is an extension for Firefox and Google Chrome.

- Allows us to create our public/private keys and connect to the blockchain.

- We recommend using MetaMask for Firefox or Chrome

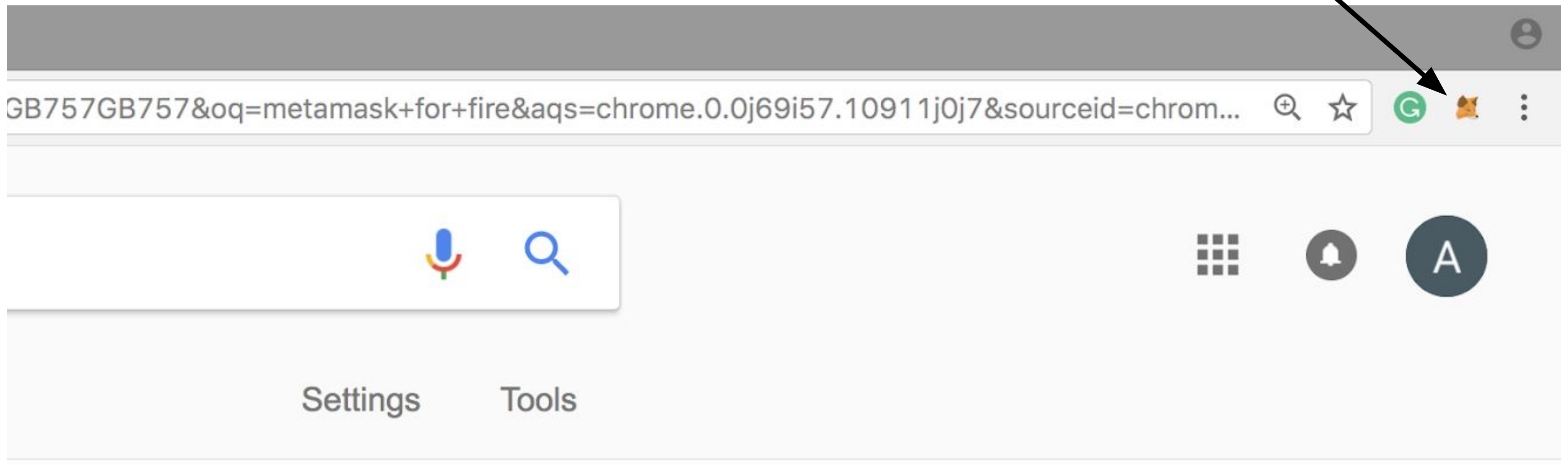
  - Download it from:

    https://metamask.io/

- Follow the instructions to install it.

Step1.1:
Set Up an Account in MetaMask

- Click on the MetaMask icon on the top right side of your Firefox browser.


GB757GB757&oq=metamask+for+fire&aqs=chrome.0.0j69i57.10911j0j7&sourceid=chrom...
A
Settings
Tools

Step1.2:
Create an Account in MetaMask

- Follow the instructions to create an account.
- After you provide a password, an account (i.e. an address, public and secret keys) will be created for you.
- **Store your seed**: you want it to restore your wallet in case you delete Metamask

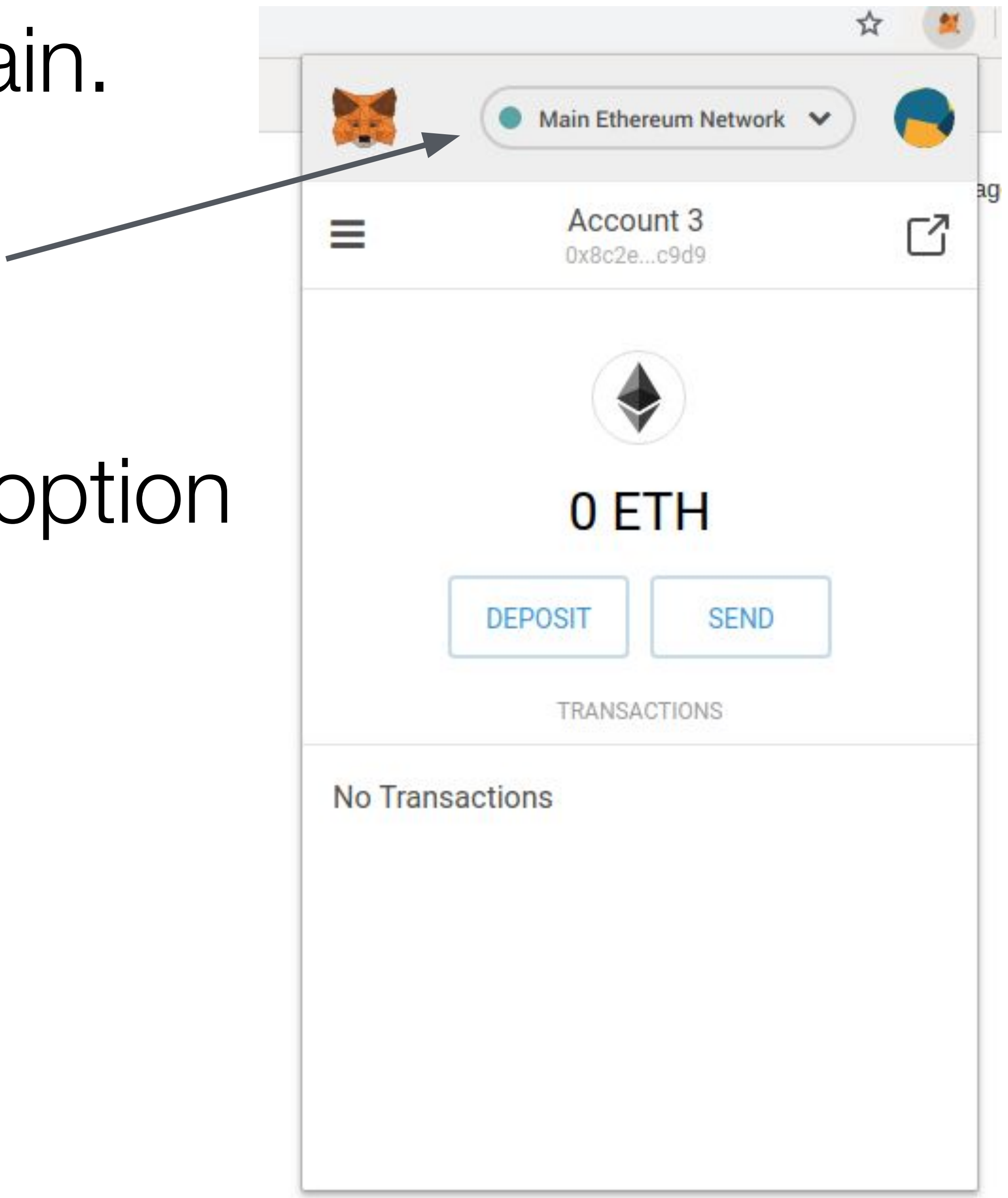
Step1.3:

Connect MetaMask to the Private Blockchain

3.1. Click the MetaMask icon again.

3.2. Click on the Network option

3.3. Click on the "Custom RPC" option



Step1.3:

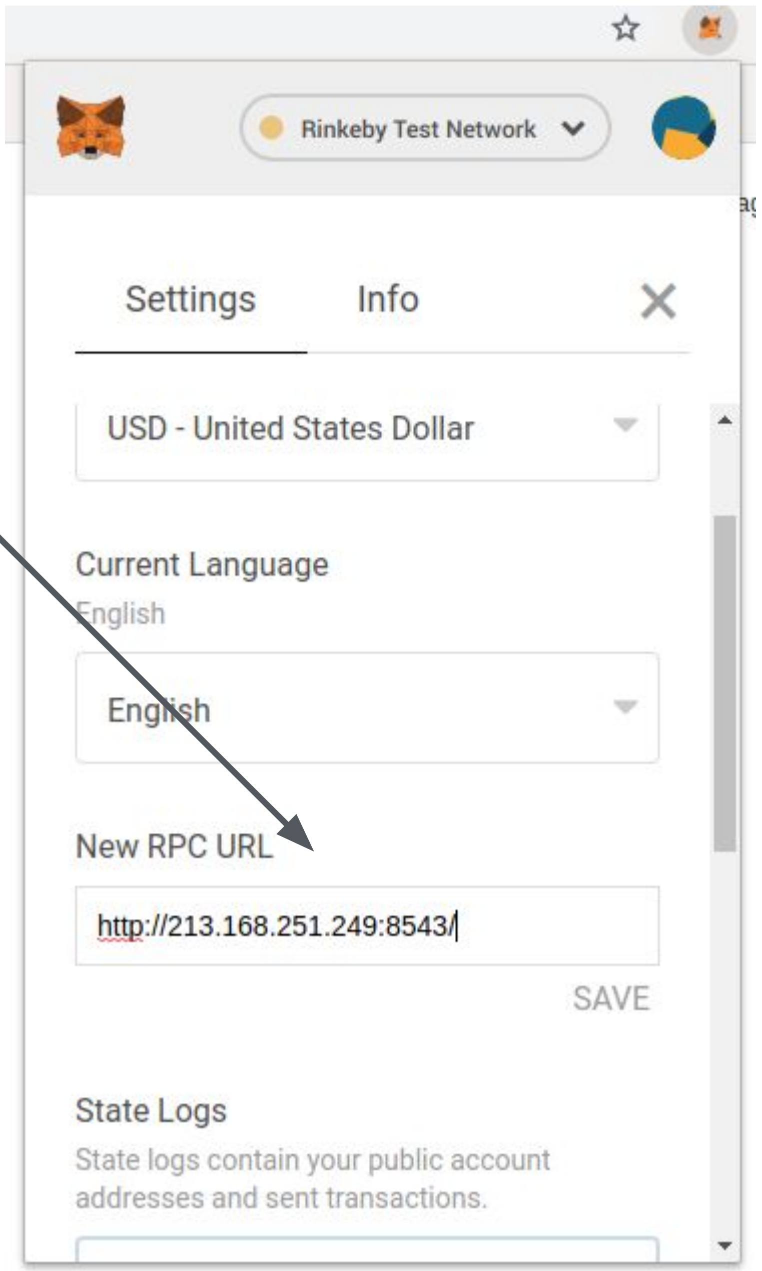
Connect MetaMask to the Private Blockchain

3.1. In the box on the top, insert the following link:

http://213.168.251.249:8543/

3.2. Click on Save to save it.

3.3. Press X to return to the main page.



USD - United States Dollar

Current Language
English

English

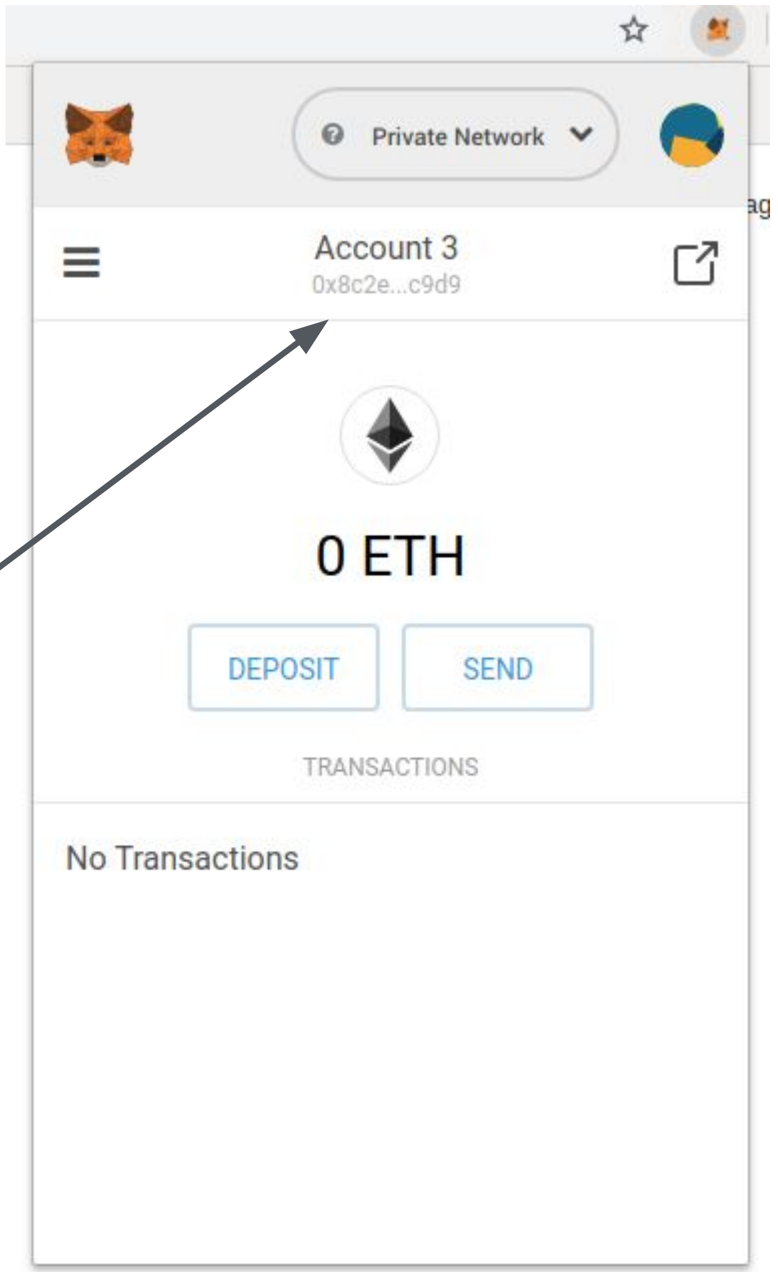
New RPC URL

http://213.168.251.249:8543/

State Logs
State logs contain your public account addresses and sent transactions.

Step1.3:
Your Address

- When you've successfully connected to the chain, this page will appear.



- This is your address; click on it to copy and send it to those who want to pay you.

Step 2:
Send us Your Account Address

- You need some Ether to send a transaction and interact with a smart contract.

- We have created a lot of Ether - you can also have some.

- Request some Ether by sending your account's address to this email address:

dimitris.karakostas@ed.ac.uk

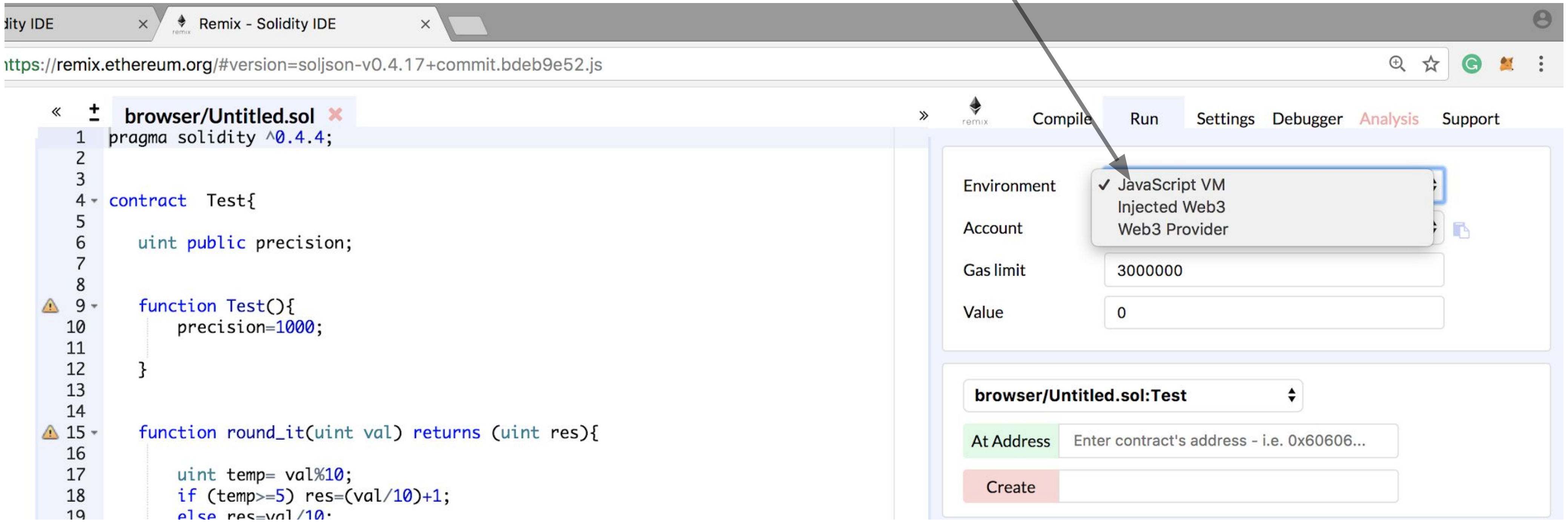
Step 3:
Getting Familiar with Remix Ethereum: Online Solidity Compiler

- You can write, debug, deploy (i.e. send to a blockchain) your smart contract via remix Ethereum: remix.ethereum.org

- Also, you can interact with your deployed contract using remix.

Step 3:
Getting familiar with Remix Ethereum: Online Solidity Compiler

- Before you deploy your smart contract to the private chain, run and debug it online.
  - In the case where you want to run it online, you should set environment to: JavaScript VM.



https://remix.ethereum.org/#version=soljson-v0.4.17+commit.bdeb9e52.js

browser/Untitled.sol

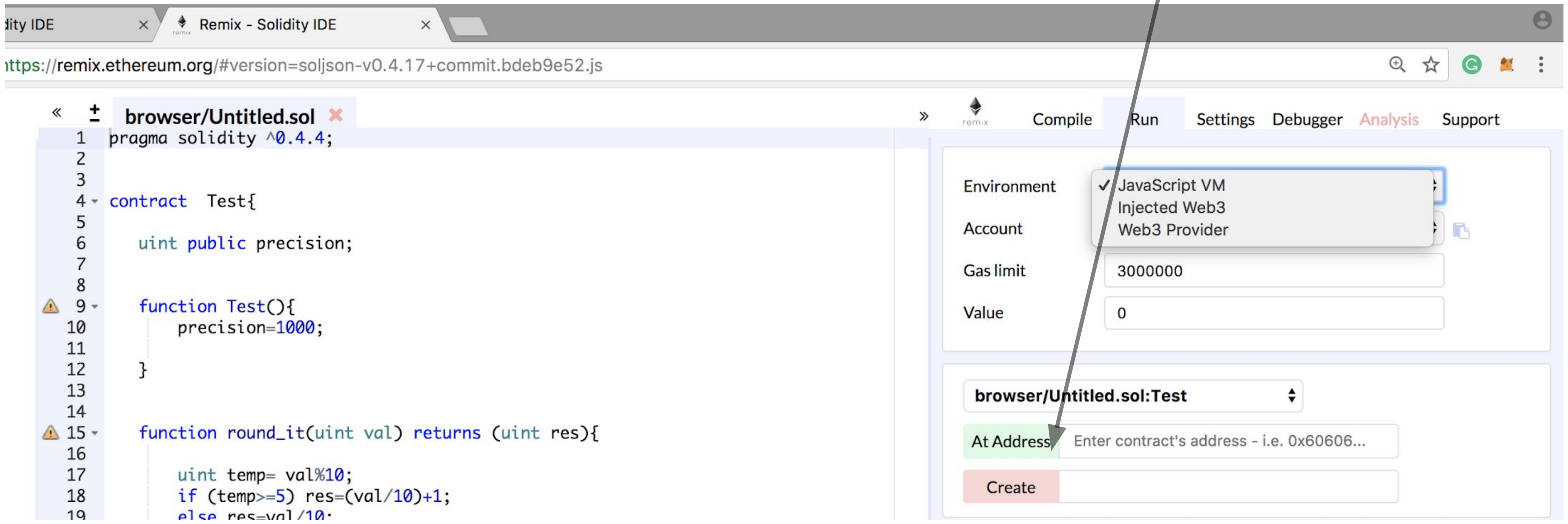
```
pragma solidity ^0.4.4;


contract  Test{

   uint public precision;


   function Test(){
       precision=1000;

   }


   function round_it(uint val) returns (uint res){

       uint temp= val%10;
       if (temp>=5) res=(val/10)+1;
       else res=val/10;
```

Environment: ✓ JavaScript VM / Injected Web3 / Web3 Provider
Account
Gas limit: 3000000
Value: 0

browser/Untitled.sol:Test
At Address: Enter contract's address - i.e. 0x60606...

Step 3:
Getting familiar with Remix Ethereum:
Online Solidity Compiler

- To compile your smart contract, click on Create button.
- After compiling the contract, remix creates a user interface for the functions you defined in the contract and you can pass parameters to it.



```
pragma solidity ^0.4.4;


contract  Test{

   uint public precision;


   function Test(){
        precision=1000;

   }


   function round_it(uint val) returns (uint res){

        uint temp= val%10;
        if (temp>=5) res=(val/10)+1;
        else res=val/10;
```

Environment	✓ JavaScript VM / Injected Web3 / Web3 Provider
Account	
Gas limit	3000000
Value	0

browser/Untitled.sol:Test

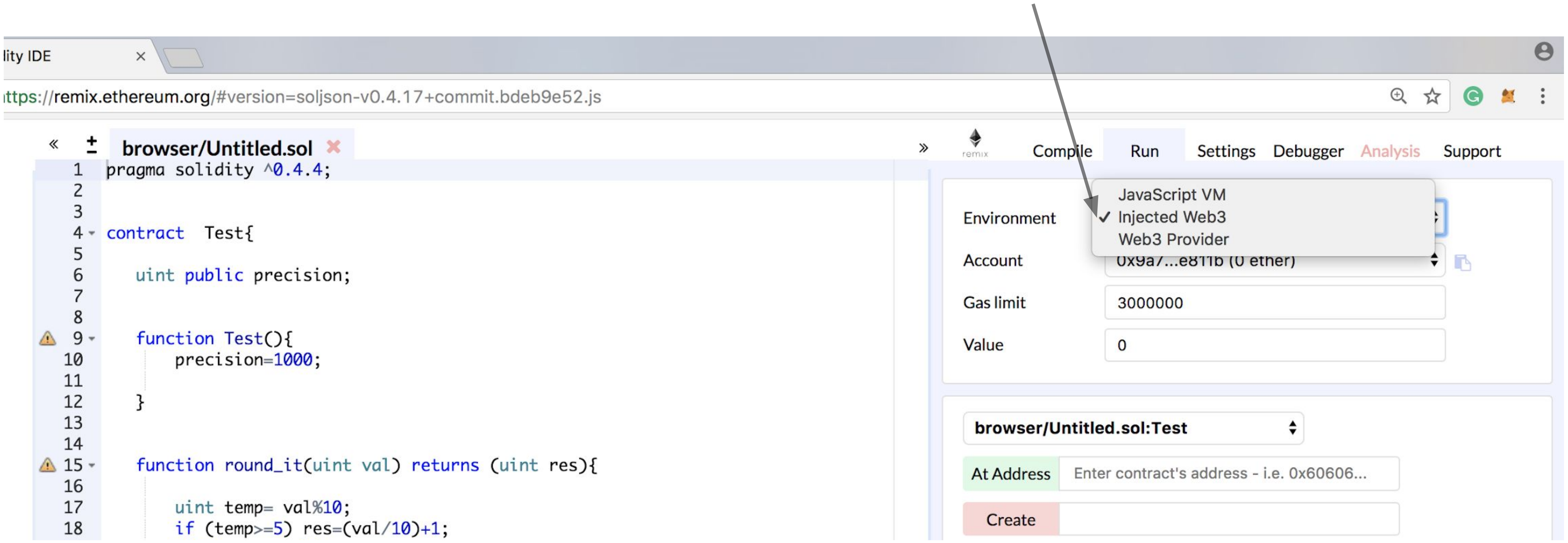
At Address — Enter contract's address - i.e. 0x60606...

Create

Step 4.1:
Deploying Smart Contract to the Private Chain

Configurations

- First, you need to connect Metamask to the blockchain, as we described in the earlier slides.
- In remix, set the environment to: Injected Web3.



lity IDE

ttps://remix.ethereum.org/#version=soljson-v0.4.17+commit.bdeb9e52.js

browser/Untitled.sol

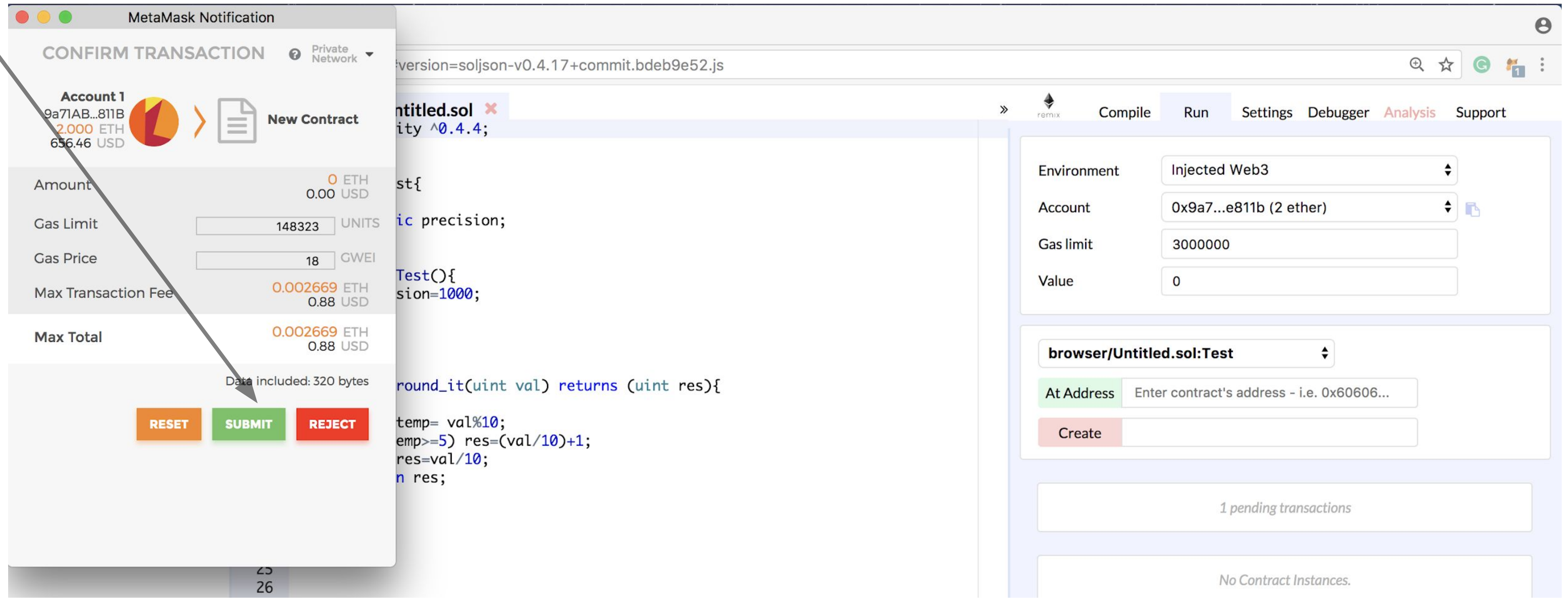
```
pragma solidity ^0.4.4;


contract  Test{

    uint public precision;


    function Test(){
        precision=1000;

    }


    function round_it(uint val) returns (uint res){

        uint temp= val%10;
        if (temp>=5) res=(val/10)+1;
```

Environment: Injected Web3
Account: 0x9a7...e811b (0 ether)
Gas limit: 3000000
Value: 0
browser/Untitled.sol:Test
At Address: Enter contract's address - i.e. 0x60606...

Step 4.2: Deploying Smart Contract to the Private Chain

Deploying a Contract to the Blockchain

- Click on Create button.
- Next, MetaMask page will appear and by clicking on submit, you send your contract to the blockchain.



Account 1
9a71AB...811B
2.000 ETH
656.46 USD

Amount	0 ETH 0.00 USD
Gas Limit	148323 UNITS
Gas Price	18 GWEI
Max Transaction Fee	0.002669 ETH 0.88 USD
Max Total	0.002669 ETH 0.88 USD

Data included: 320 bytes

version=soljson-v0.4.17+commit.bdeb9e52.js

ntitled.sol

```
ity ^0.4.4;

st{

ic precision;

Test(){
sion=1000;

round_it(uint val) returns (uint res){

temp= val%10;
emp>=5) res=(val/10)+1;
res=val/10;
n res;
```

Environment	Injected Web3
Account	0x9a7...e811b (2 ether)
Gas limit	3000000
Value	0

browser/Untitled.sol:Test

At Address: Enter contract's address - i.e. 0x60606...

Create

1 pending transactions

No Contract Instances.

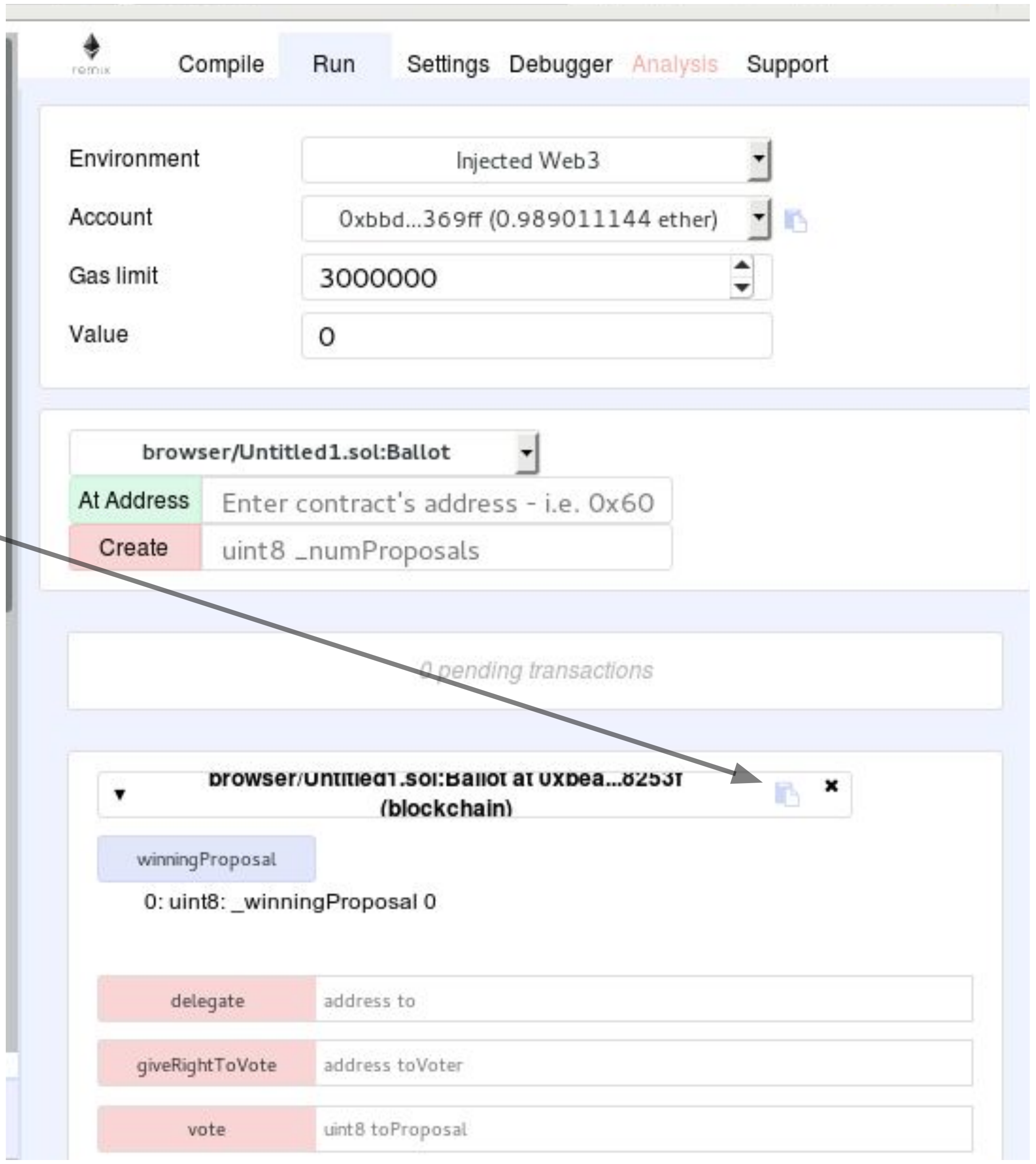
Step 4.3: Deploying Smart Contract to the Private Chain

Saving the Deployed Contract's Address

- When, your contract is successfully submitted/deployed, remix provides the contract address on the blockchain.

- You can copy the address from here.

- You need the contract code and the address next time you want to interact with your deployed contract.



Environment: Injected Web3
Account: 0xbbd...369ff (0.989011144 ether)
Gas limit: 3000000
Value: 0

browser/Untitled1.sol:Ballot

Enter contract's address - i.e. 0x60
uint8 _numProposals

0 pending transactions

browser/Untitled1.sol:Ballot at 0xbea...8253f (blockchain)

0: uint8: _winningProposal 0

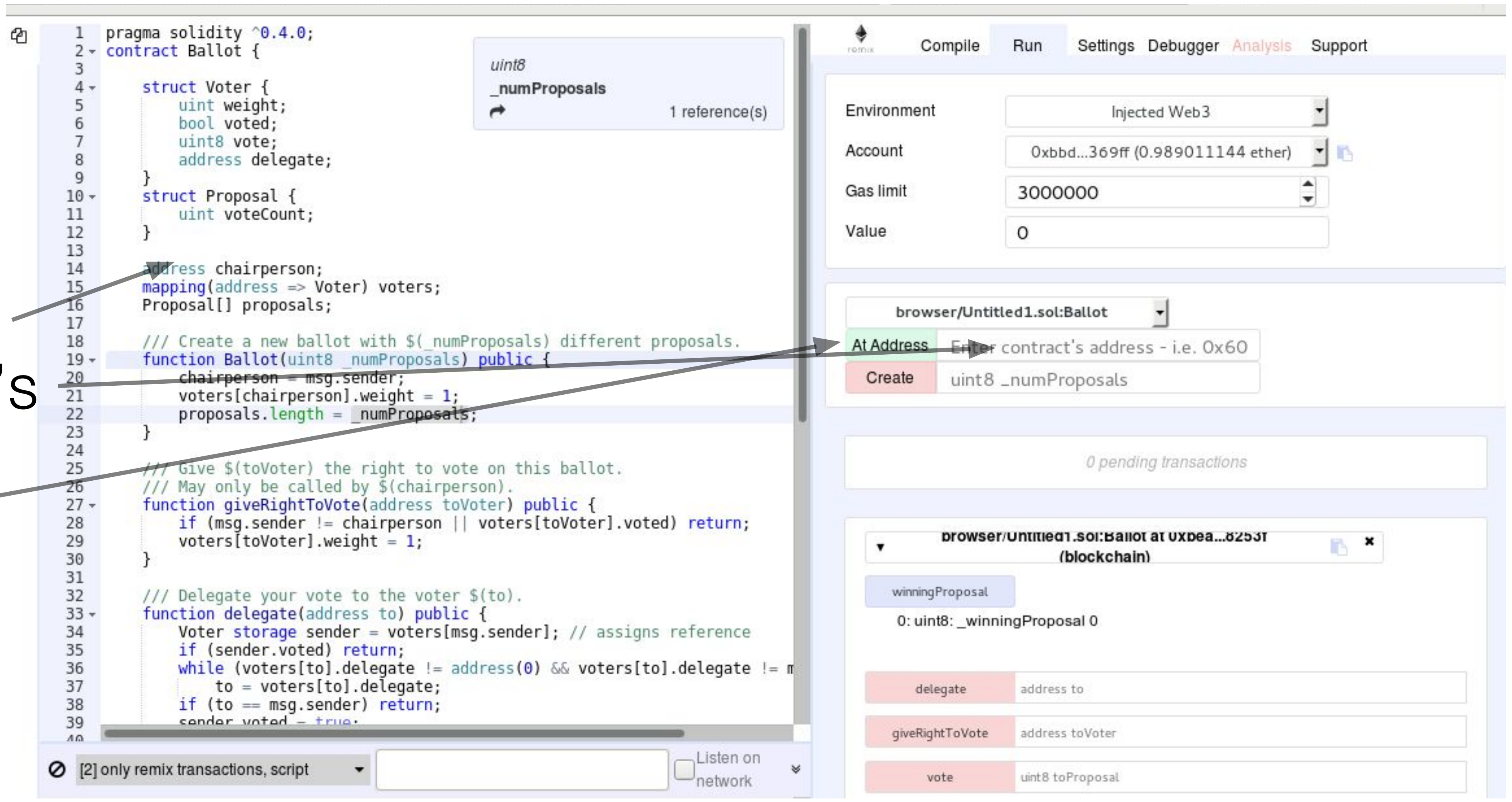
address to
address toVoter
uint8 toProposal

Step 4.3:
Deploying Smart Contract to the Private Chain

Interacting with a Deployed Contract

1. Log in to MetaMask and connect to the blockchain (as previously explained)
2. In remix, set the environment to: Injected Web3.

3. In remix, insert the contract code, insert the deployed contract's address and click on: At Address.



```
pragma solidity ^0.4.0;
contract Ballot {

    struct Voter {
        uint weight;
        bool voted;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        uint voteCount;
    }

    address chairperson;
    mapping(address => Voter) voters;
    Proposal[] proposals;

    /// Create a new ballot with $(_numProposals) different proposals.
    function Ballot(uint8 _numProposals) public {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
        proposals.length = _numProposals;
    }

    /// Give $(toVoter) the right to vote on this ballot.
    /// May only be called by $(chairperson).
    function giveRightToVote(address toVoter) public {
        if (msg.sender != chairperson || voters[toVoter].voted) return;
        voters[toVoter].weight = 1;
    }

    /// Delegate your vote to the voter $(to).
    function delegate(address to) public {
        Voter storage sender = voters[msg.sender]; // assigns reference
        if (sender.voted) return;
        while (voters[to].delegate != address(0) && voters[to].delegate != m
            to = voters[to].delegate;
        if (to == msg.sender) return;
        sender.voted = true;
```

uint8 _numProposals 1 reference(s)

[2] only remix transactions, script

Listen on network

Environment: Injected Web3
Account: 0xbbd...369ff (0.989011144 ether)
Gas limit: 3000000
Value: 0

browser/Untitled1.sol:Ballot
At Address: Enter contract's address - i.e. 0x60
Create: uint8 _numProposals

0 pending transactions

browser/Untitled1.sol:Ballot at 0xbea...8253f (blockchain)

winningProposal
0: uint8: _winningProposal 0

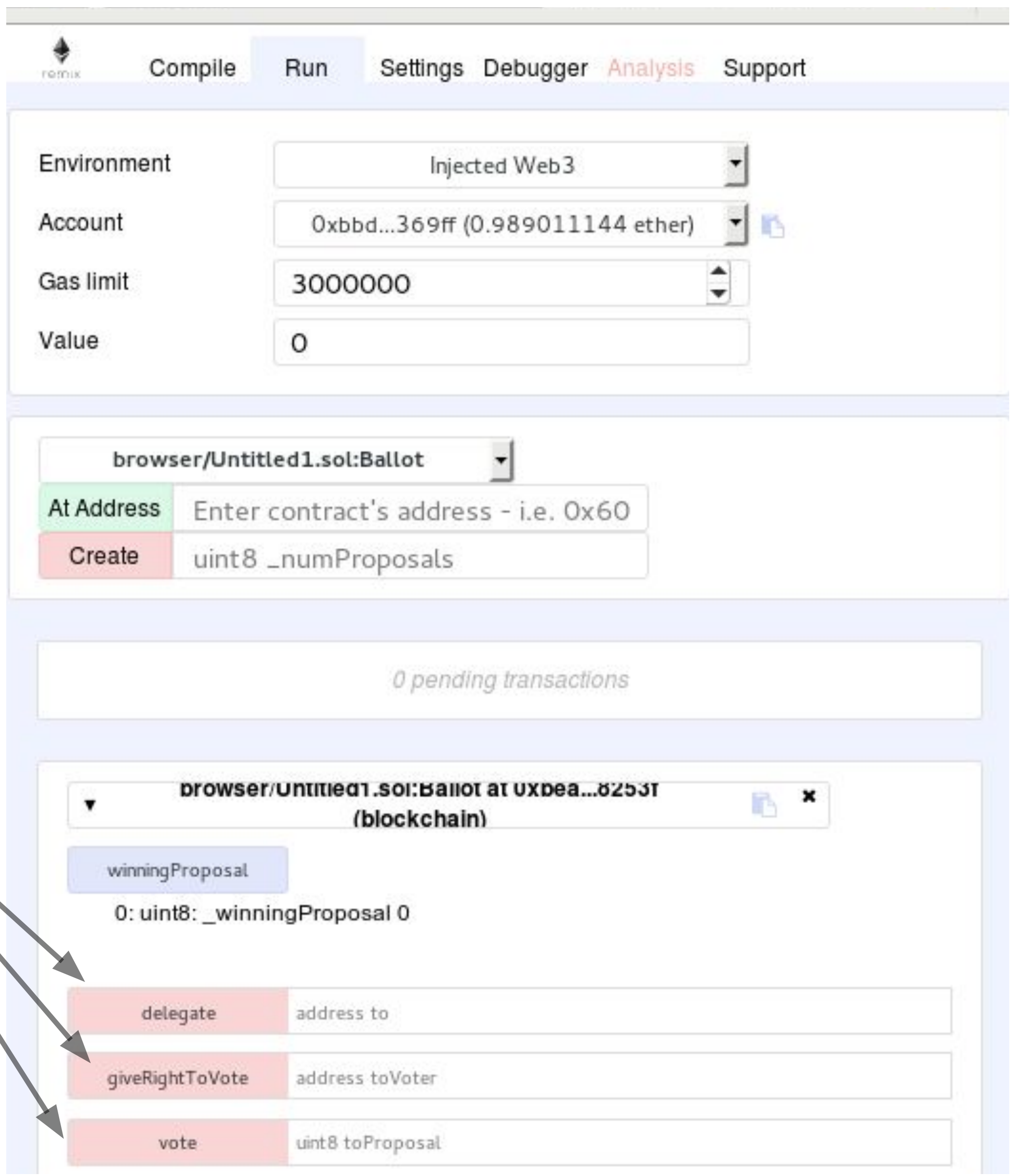
delegate: address to
giveRightToVote: address toVoter
vote: uint8 toProposal

Step 4.3:
Deploying Smart Contract to the Private Chain

Interacting with a Deployed Contract

4- All the public/external functions in the contract are provided and you can pass arguments on them and invoke them.

- The invocation of a function, that changes the contract state, will result in new transaction.



Environment: Injected Web3
Account: 0xbbd...369ff (0.989011144 ether)
Gas limit: 3000000
Value: 0

browser/Untitled1.sol:Ballot
At Address: Enter contract's address - i.e. 0x60
Create: uint8 _numProposals

0 pending transactions

browser/Untitled1.sol:Ballot at 0xbea...8253f (blockchain)

winningProposal
0: uint8: _winningProposal 0

delegate: address to
giveRightToVote: address toVoter
vote: uint8 toProposal