

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 the private blockchain, and interact with it.
- You must use either a university's **DICE machine** to connect to the blockchain or via VPN to Informatics, see here <http://computing.help.inf.ed.ac.uk/openvpn>
- 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 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://addons.mozilla.org/en-US/firefox/addon/ether-metamask/>

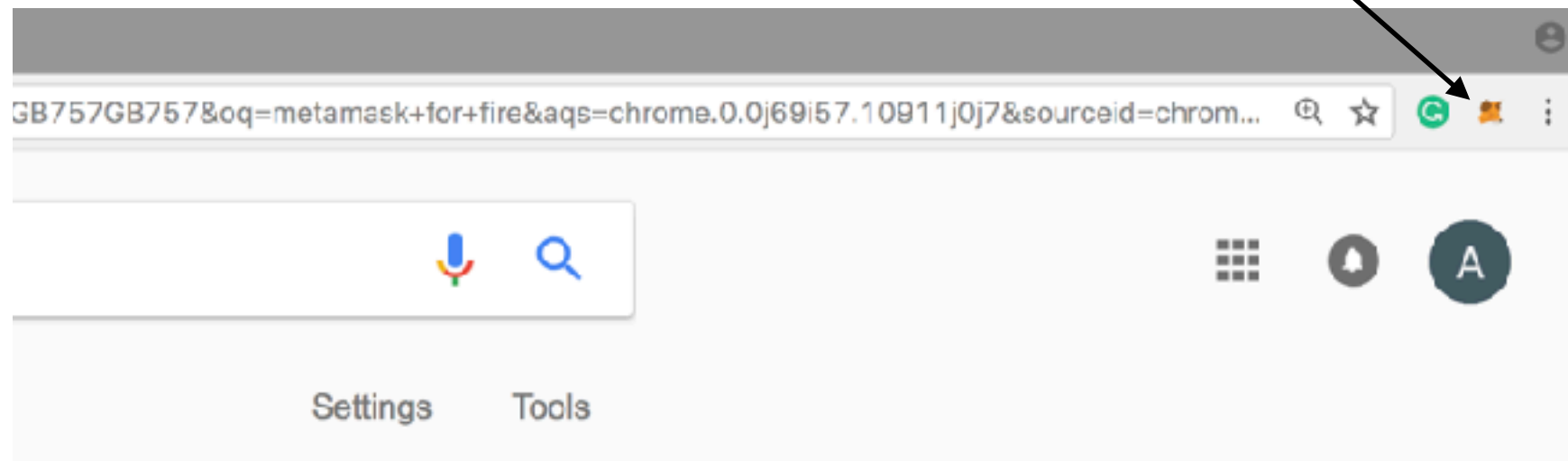
<https://metamask.io/>

- Follow the instructions to install it.

Step 1.1:

Set Up an Account in MetaMask

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



Step 1.2:

Create an Account in MetaMask

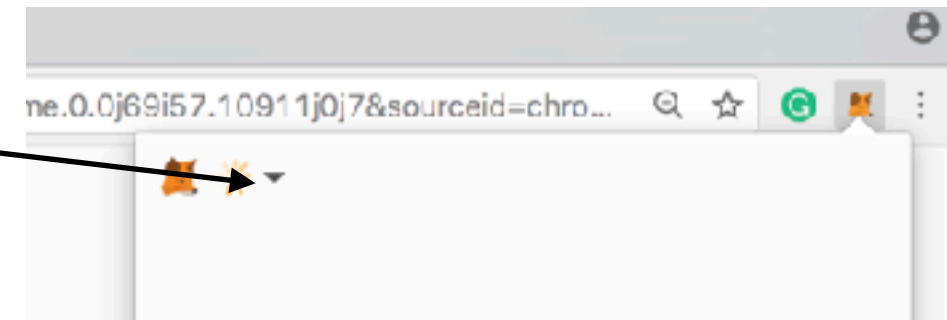
- Follow the instructions to create an account.
- After you provide a password, an account (i.e. an address, public and secret keys) will be created for you.

Step 1.3:

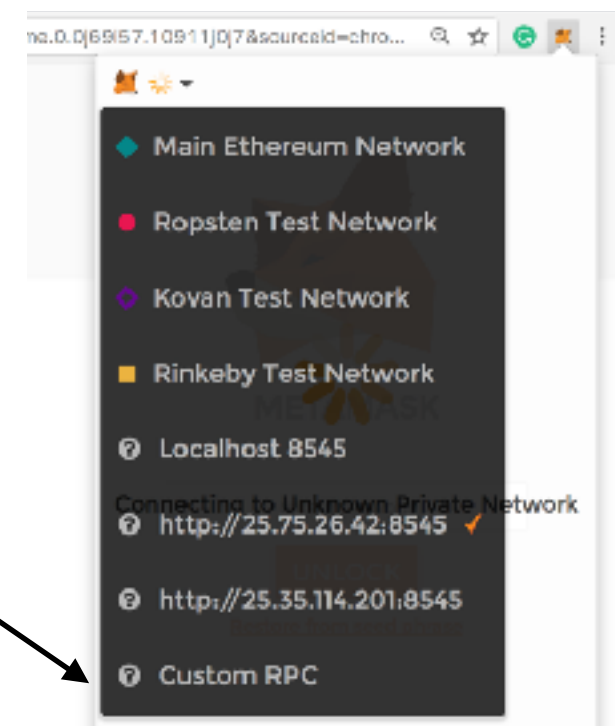
Connect MetaMask to the Private Blockchain

3.1. Click the MetaMask icon again.

3.2. Click on the black triangle.



3.3. Select the last option: Custom RPC



Step 1.3:

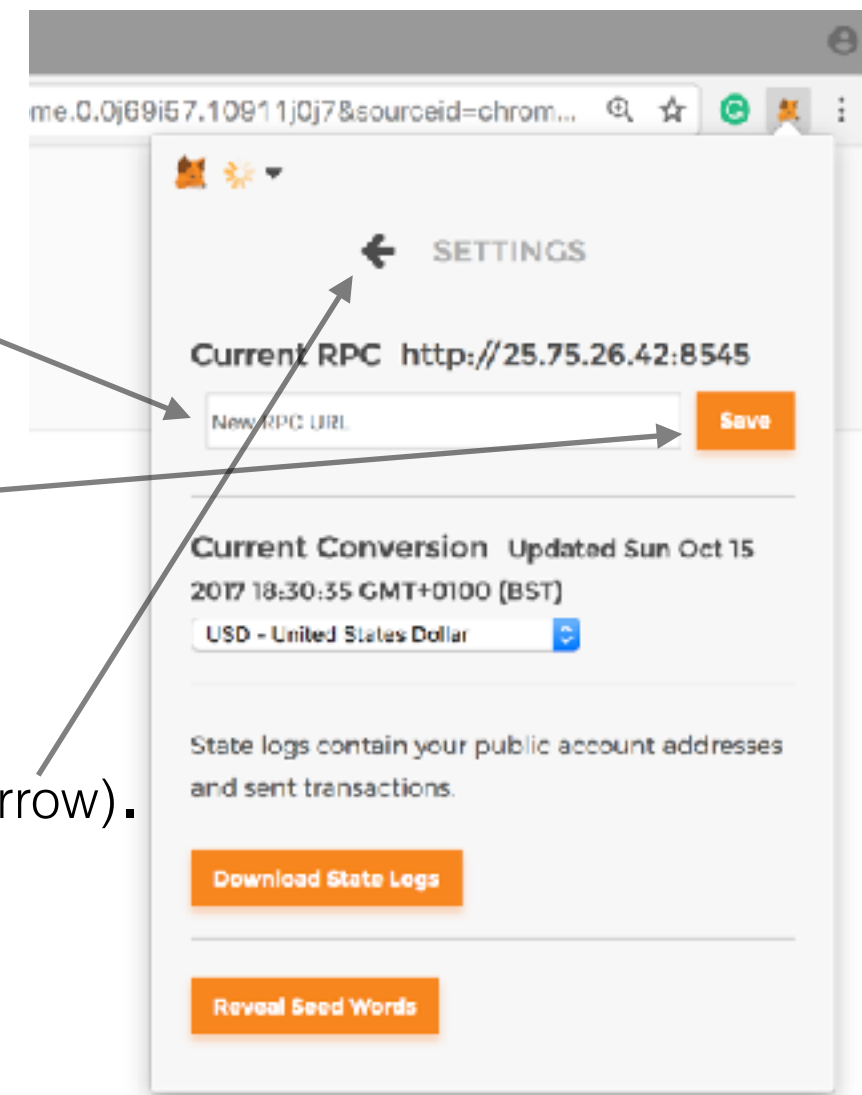
Connect MetaMask to the Private Blockchain

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

<http://129.215.32.117:8545>

3.2. Save it.

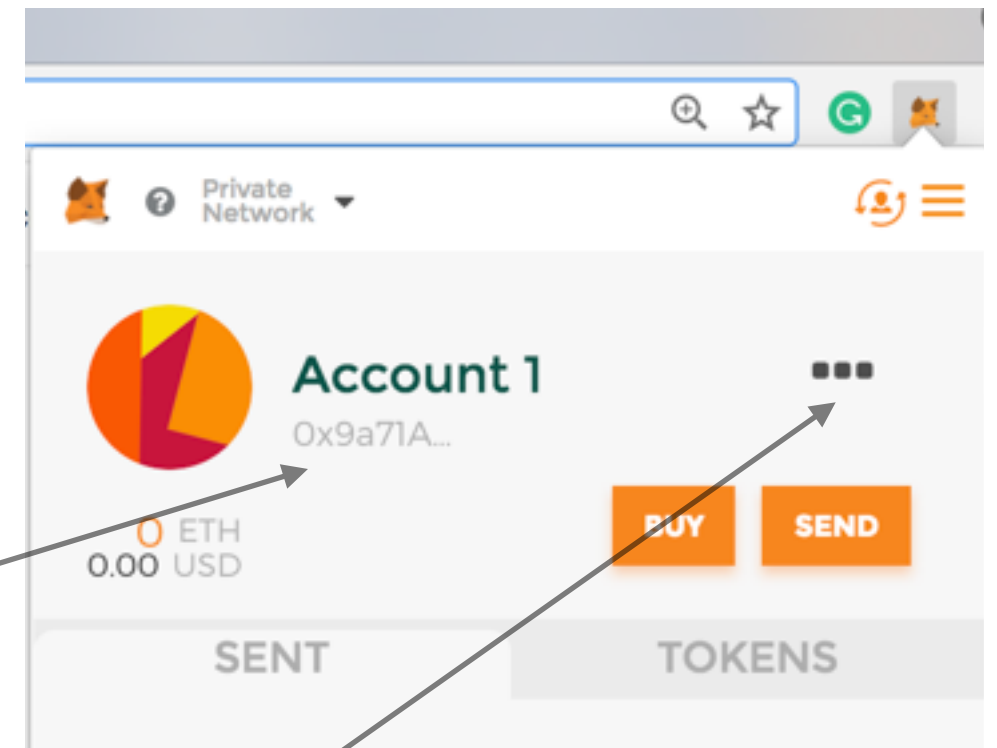
3.3. Return to the main page (click on the arrow).



Step 1.3:

Your Address

- When you're successfully connected to the chain, this page would appear.



- This is your address; you can 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 to a blockchain.
- We have created some Ether. Our Ether has not value outside of the private chain.
- Email your account address to this email address:

blockchain.lab.2017@gmail.com
- So we can send some Ether to you.

Step 3:

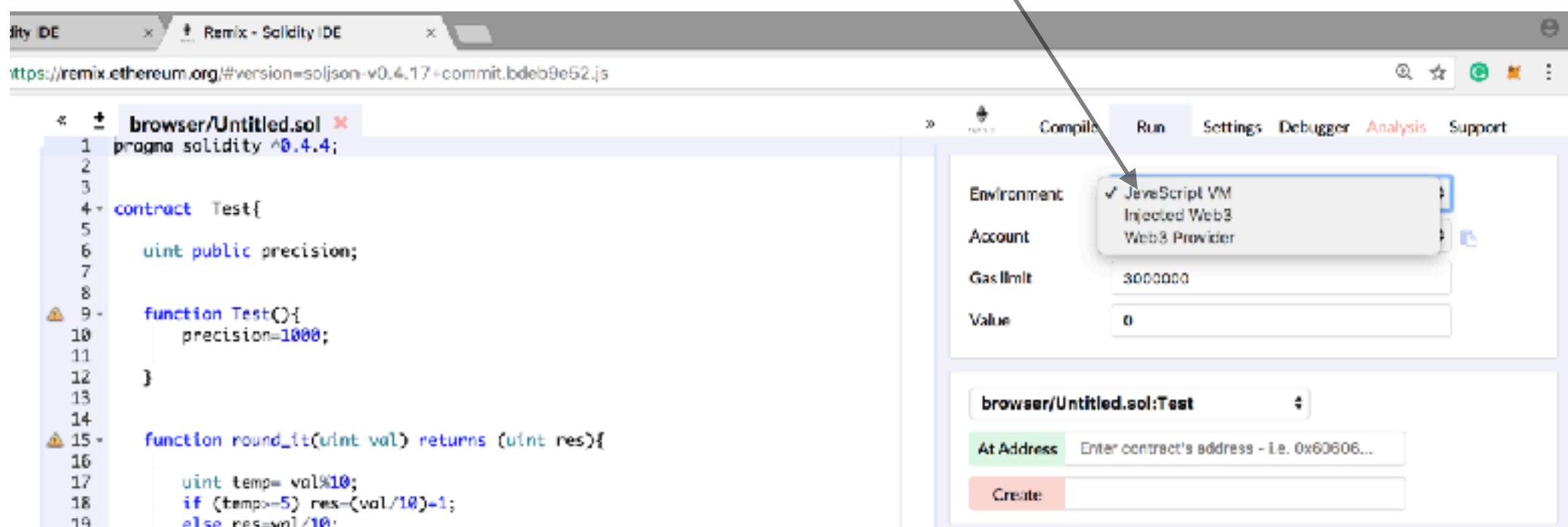
Getting Familiar with Remix Ethereum: Online Solidity Compiler

- 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

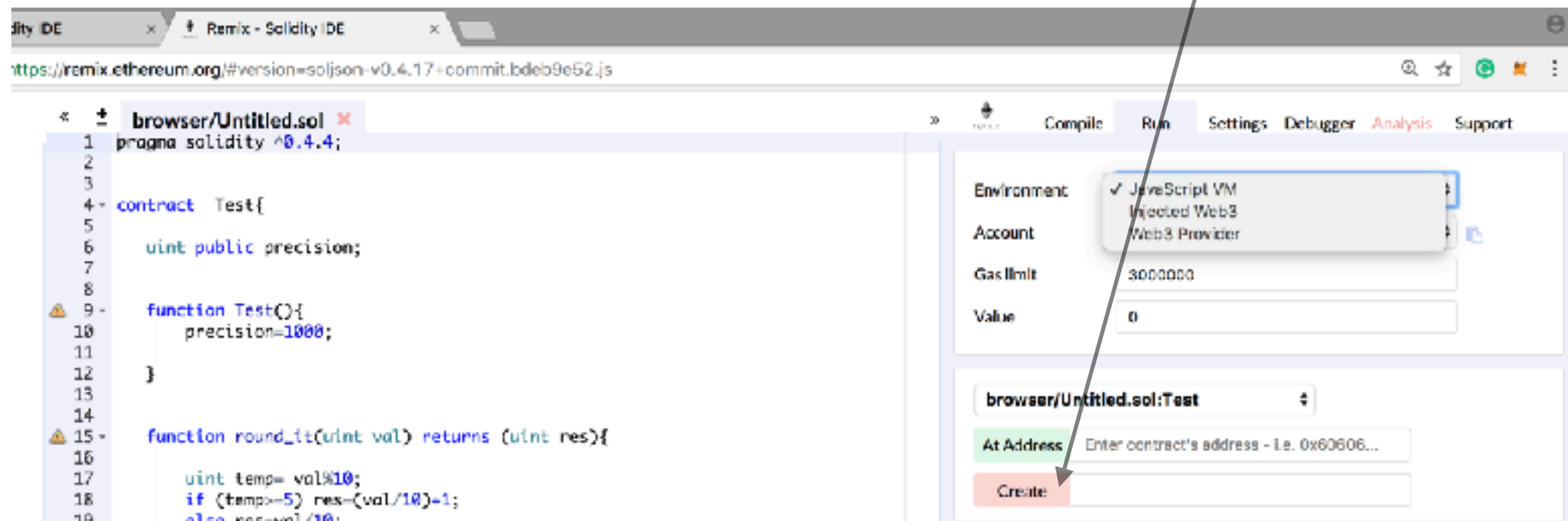
- Before you deploy your smart contract to the **private chain**, run and debug it online.
- In the case where you want to run it online, you should set environment to: **JavaScript VM**.



Step 3:

Getting familiar with Remix Ethereum: Online Solidity Compiler

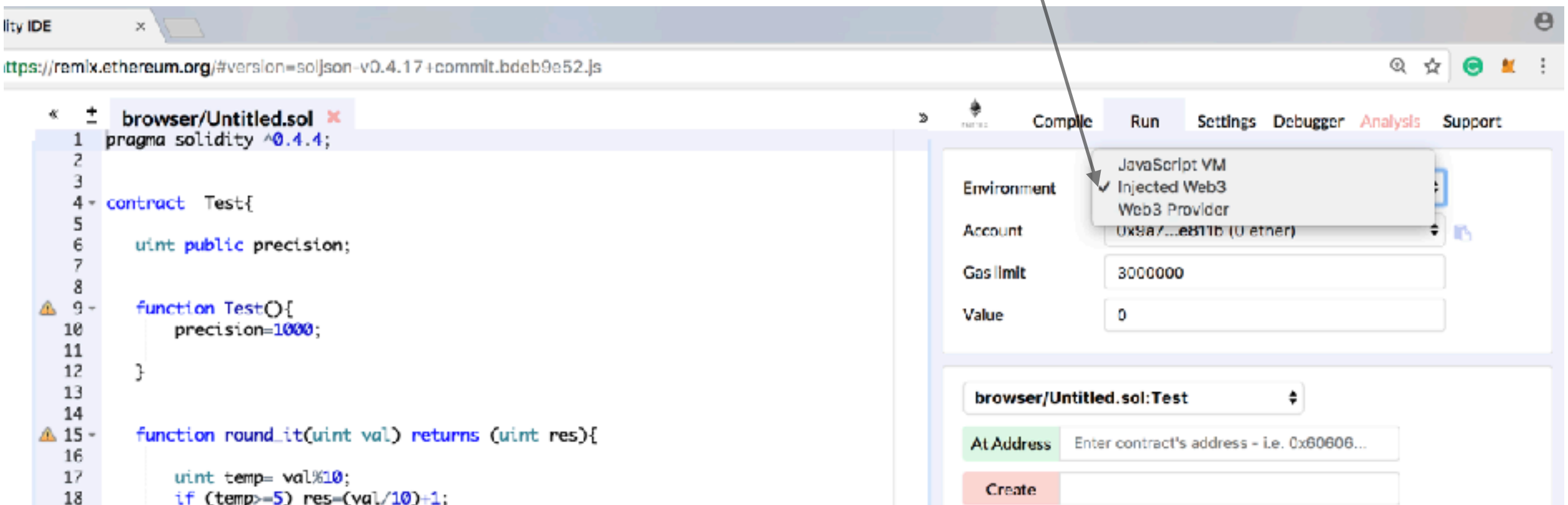
- To compile your smart contract, click on **Create** button.
- After compiling the contract, remix creates a user interface for the functions you defined in the contract and you can pass parameters to it.



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**.

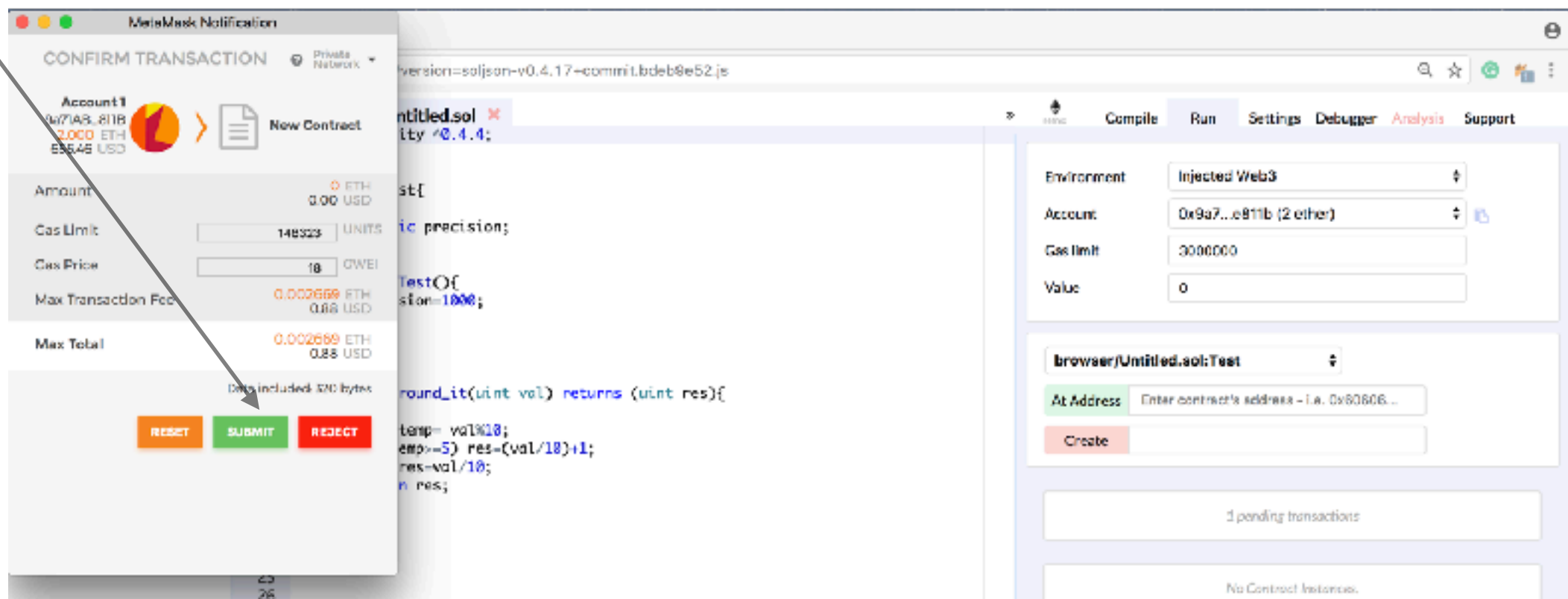


Step 4.2:

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.

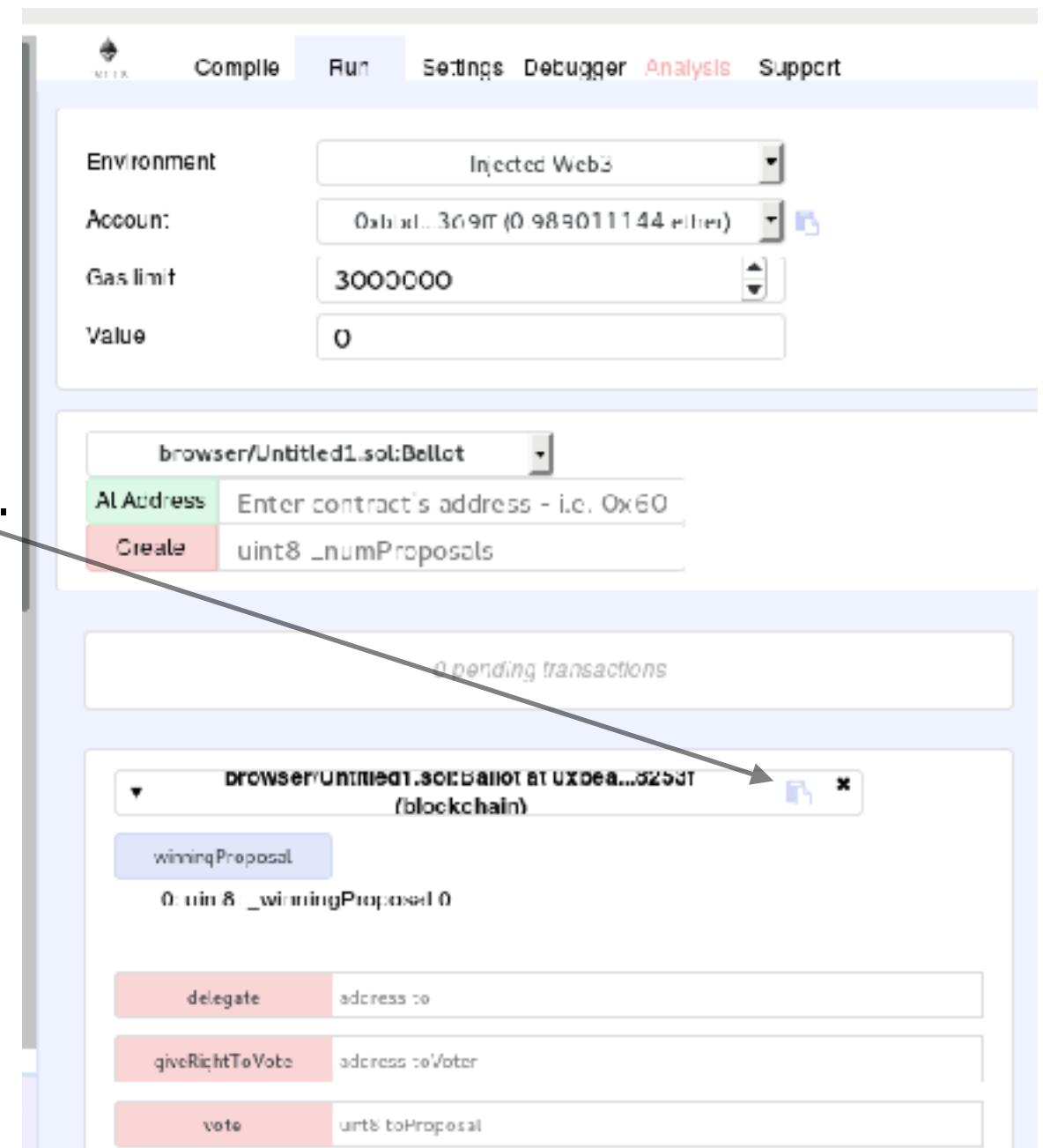


Step 4.3:

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.



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** and the deployed contract address and click on: **At Address**.

The screenshot displays the Remix IDE interface. On the left, the Solidity code for a 'Ballot' contract is visible, including a pragma statement for Solidity 0.4.0, a 'contract Ballot' definition, and functions for creating a ballot, giving voting rights, and delegating votes. A variable '_numProposals' is defined as '1 referendumis;'. On the right, the 'Run' tab is active, showing the 'Environment' set to 'Injected Web3'. Below this, the 'Account' is '0xbad...369M (0.983011144 ether)', 'Gas limit' is '3000000', and 'Value' is '0'. The 'Create' button is highlighted, and the 'At Address' field is populated with '0x60'. The 'Create' button is also highlighted. Below the 'Create' button, the '0 pending transactions' section is visible. At the bottom, the 'Interact' tab is active, showing the 'winningProposal' variable and the 'delegate', 'giveRightToVote', and 'vote' functions with their respective parameters.

```
1 pragma solidity ^0.4.0;
2 contract Ballot {
3
4     struct Voter {
5         uint weight;
6         bool voted;
7         uint8 vote;
8         address delegate;
9     }
10    struct Proposal {
11        uint voteCount;
12    }
13
14    address chairperson;
15    mapping(address => Voter) voters;
16    Proposal[] proposals;
17
18    // Create a new ballot with $( _numProposals ) different proposals.
19    function Ballot(uint8 _numProposals) public {
20        chairperson = msg.sender;
21        voters[chairperson].weight = 1;
22        proposals.length = _numProposals;
23    }
24
25    // Give $(toVoter) the right to vote on this ballot.
26    // May only be called by $(chairperson).
27    function giveRightToVote(address toVoter) public {
28        if (msg.sender != chairperson || voters[toVoter].voted) return;
29        voters[toVoter].weight = 1;
30    }
31
32    // Delegate your vote to the voter $(to).
33    function delegate(address to) public {
34        Voter storage sender = voters[msg.sender]; // assigns reference
35        if (sender.voted) return;
36        while (voters[to].delegate != address(0) && voters[to].delegate != r
37            to = voters[to].delegate;
38        if (to == msg.sender) return;
39        sender.voted = true;
```


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 results in new transaction.

