

Ethereum Workshop

- Introductions
- Solidity presentation
- Setup & Extending contracts
- Lunch
- Tooling presentation
- Setup, Testing & deployment

Bank Account

- Create deposits.
- Transfer value.
- Query balance.

```
pragma solidity ^0.4.18;
```

```
contract Token {  
    // Contract code here  
}
```

Hashing

- SHA2 / Keccak
- Unique Fingerprint
- Message / Digest
- `keccak("Crypto") = 850a73150588979a5...`

Mapping

```
mapping(address => uint256) balances;
```

Balances

	A	B
1	Account	Balance
2	0x1313734d2D6625173278978DDaa7B63400462745	10
3	0xf5B1b23448F8f970ce3AEC7fd78aB8EEc819d161	12
4	0xdF6ffe1d380e273702140937FB67c6EEB52d34F9	5
5	0xe221cfC94E5f649c20276deC564B28E80f3C0538	4

Function

```
function transfer(address _to, uint256 _value) public returns (bool)
```


Events

```
// Definition  
event Transfer(address indexed from, address indexed to, uint256 value);
```

```
// Usage  
emit Transfer(msg.sender, _to, _value);
```

Numbers

Ethereum unit converter

1000000000000000000

Wei

1000000000000000

KWei

1000000000000

MWei

1000000000

GWei (Shannon)

1000000

Szabo

1000

Finney

1

Ether

0.001

KEther

0.000001

MEther

0.000000001

GEther

0.0000000000001

TEther

Types

- uint
- int
- string
- bytes
- address

uint & casting

```
uint8 public constant decimals = 18;
```

```
uint256 public constant INITIAL_SUPPLY = 10000 * (10 ** uint(decimals));
```

Arithmetic

- Addition: $x + y$
- Subtraction: $x - y$
- Multiplication: $x * y$
- Division: x / y
- Modulus: $x \% y$

Inheritance

```
contract SimpleToken is BasicToken {  
    ...  
}
```

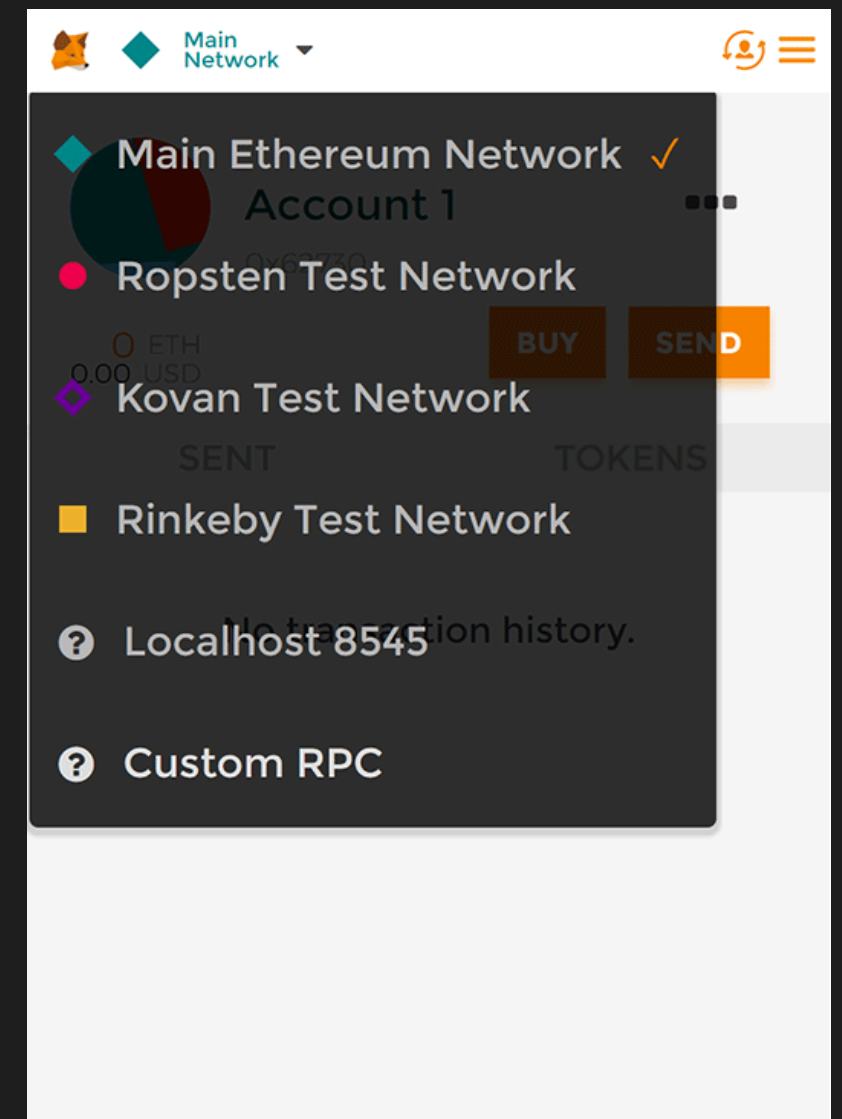
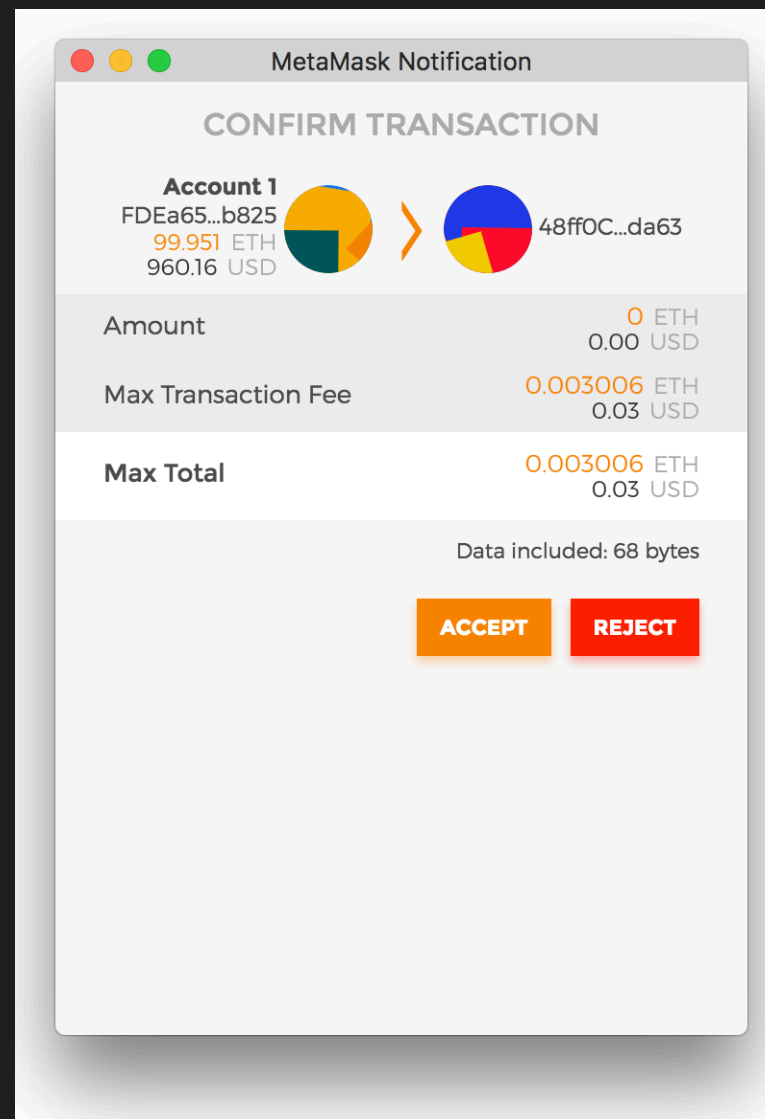
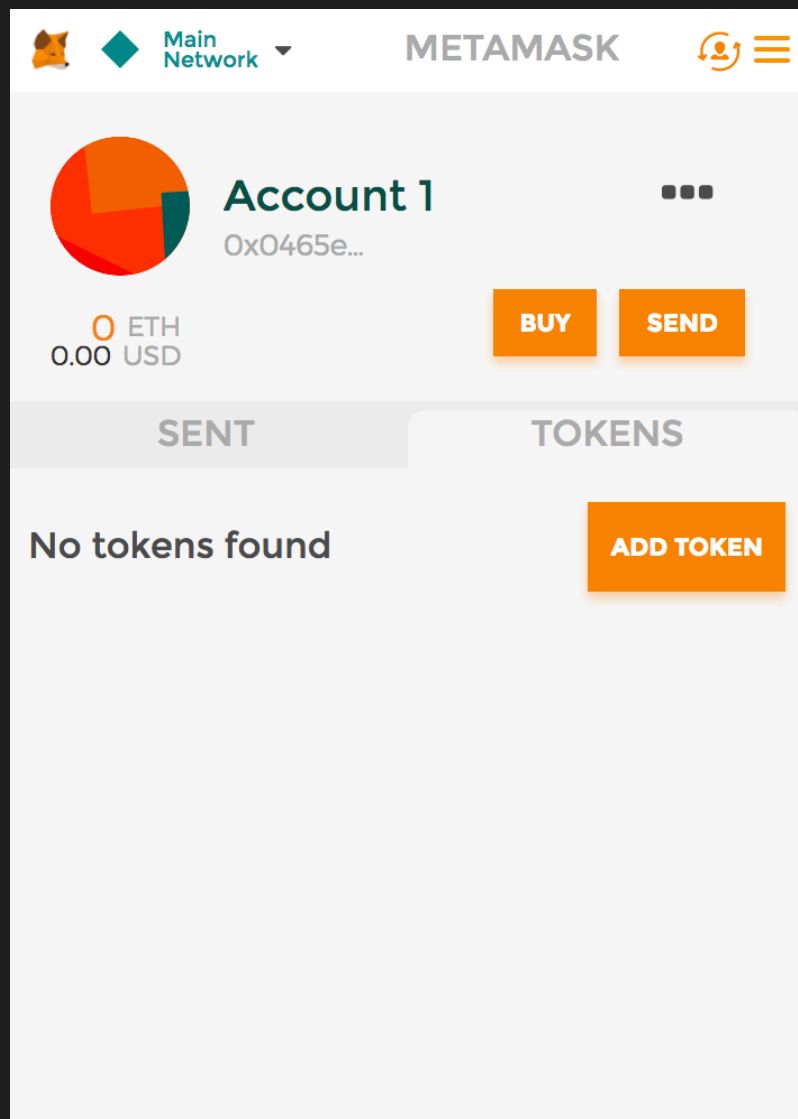
Constructor

```
contract Foo {  
    uint someValue;  
  
    constructor(uint _aValue) {  
        someValue = _aValue;  
    }  
}
```

Setup for dev

- MetaMask
- Faucet
- Remix IDE

MetaMask



Remix IDE

The screenshot displays the Remix IDE interface. The browser address bar shows the URL `https://remix.ethereum.org/#optimize=false&version=soljson-v0.4.21+commit.dfe3193c.js`. The main editor area contains a Solidity contract named `SimpleToken` with the following code:

```
1 pragma solidity ^0.4.17;
2
3 /*
4 contract SimpleToken {
5
6     string public constant name = "SimpleToken";
7
8 }
```

The right sidebar contains several panels:

- Compile**: Shows the environment set to `Injected Web3` on the `Rinkeby (4)` network. The account is empty, the gas limit is `3000000`, and the value is `0 wei`.
- Run**: Contains a `Create` button and a `Load contract from Address` button with an `At Address` input field.
- Debugger**: Shows `0 pending transactions` and `0 contract instances`.

The left sidebar shows a file explorer with `browser` and `config` folders.

Faucet

Secure | <https://faucet.rinkeby.io>



Rinkeby Authenticated Faucet

Give me Ether ▼

5 peers 2211942 blocks 9.046256971665328e+56 Ethers 99387 funded

How does this work?

This Ether faucet is running on the Rinkeby network. To prevent malicious actors from exhausting all available funds or accumulating enough Ether to mount long running spam attacks, requests are tied to common 3rd party social network accounts. Anyone having a Twitter, Google+ or Facebook account may request funds within the permitted limits.



To request funds via Twitter, make a [tweet](#) with your Ethereum address pasted into the contents (surrounding text doesn't matter). Copy-paste the [tweets URL](#) into the above input box and fire away!











To request funds via Google Plus, publish a new **public** post with your Ethereum address embedded into the content (surrounding text doesn't matter). Copy-paste the posts URL into the above input box and fire away!



To request funds via Facebook, publish a new **public** post with your Ethereum address embedded into the content (surrounding text doesn't matter). Copy-paste the [posts URL](#) into the above input box and fire away!

Deployment

Environment	Injected Web3  Rinkeby (4)  		
Account	0x131...62745 (76.19640348300277042   		
Gas limit	<input type="text" value="3000000"/>		
Value	<input type="text" value="0"/>	wei	



Create

At Address

Etherscan

Secure <https://rinkeby.etherscan.io/tx/0x6a34d423f9315b04ae8541a50697732dd1243294df59ebd2656f80d1d6432f0d>

Transaction 0x6a34d423f9315b04ae8541a50697732dd1243294df59ebd2656f80d1d6432f0d

[Home](#) / [Transactions](#) / [Transaction Information](#)

Overview

Transaction Information

Tools & Utilities ▼

TxHash: 0x6a34d423f9315b04ae8541a50697732dd1243294df59ebd2656f80d1d6432f0d

TxReceipt Status: **Success**

Block Height: **2206730** (1 block confirmation)

TimeStamp: 18 secs ago (May-01-2018 10:36:49 AM +UTC)

From: [0x1313734d2d6625173278978ddaa7b63400462745](#)

To: [Contract 0xff4a216cb7f451dc3287297ce7efaac7579e3576 Created]

Value: 0 Ether (\$0.00)

Gas Limit: 442267

Gas Used By Txn: 442267

Gas Price: 0.000000002 Ether (2 Gwei)

Actual Tx Cost/Fee: 0.000884534 Ether (\$0.000000)

Nonce: 2181

[illegible]

Convert To UTF8

This page intentionally left blank

Resources

- <https://solidity.readthedocs.io>
- <https://ethereum.stackexchange.com>
- <https://openzeppelin.org>
- [General blockchain for devs](#)


Add features


- Charge a fee for transfers
- Users can purchase by sending Ether
- Owner can mint new coins
- Owner can burn coins
- Owner can be changed
- Voting for token holders (hard)


Common dev tools


- Visual Studio
- Ganache
- TestRPC / ganache-cli
- Truffle
- Geth / Parity

Ganache


 ACCOUNTS

 BLOCKS

 TRANSACTIONS

 LOGS

SEARCH FOR BLOCK NUMBERS OR TX HASHES



CURRENT BLOCK
0


GAS PRICE
20000000000

GAS LIMIT
6721975






NETWORK ID
5777

RPC SERVER
HTTP://127.0.0.1:8545

MINING STATUS
AUTOMINING



MNEMONIC	HD PATH
candy maple cake sugar pudding cream honey rich smooth crumble sweet treat	m/44'/60'/0'/0/account_index

ADDRESS 0x627306090abaB3A6e1400e9345bC60c78a8BEf57	BALANCE 100.00 ETH	TX COUNT 0	INDEX 0	
ADDRESS 0xf17f52151EbEF6C7334FAD080c5704D77216b732	BALANCE 100.00 ETH	TX COUNT 0	INDEX 1	
ADDRESS 0xC5fdf4076b8F3A5357c5E395ab970B5B54098Fef	BALANCE 100.00 ETH	TX COUNT 0	INDEX 2	
ADDRESS 0x821aEa9a577a9b44299B9c15c88cf3087F3b5544	BALANCE 100.00 ETH	TX COUNT 0	INDEX 3	
ADDRESS 0x0d1d4e623D10F9FBA5Db95830F7d3839406C6AF2	BALANCE 100.00 ETH	TX COUNT 0	INDEX 4	

TestRPC

EthereumJS TestRPC v6.0.3 (ganache-core: 2.0.2)

Available Accounts

=====

(0) 0x99b8185dd98c5ed9c847de4ffa83e88091339ac
(1) 0x8dc440ed06e1972e243789f0ba2d993be13eff53
(2) 0x0f6fcea6a9227f6eb46e42bea33ea242358ed822
(3) 0x3ea8c346cc2de114a97e22ef9eb7b98bb2a864cb
(4) 0x6a755604acea8a2e4e2eb84f43401ea42d529bfd
(5) 0x8944d6d2a406bc5504d785395fc9eadaf6ad68bf
(6) 0xe6eae4bd3dbc9ef0e28065d594695c5f1b0a3827
(7) 0xdb697b38118d267978e756e23f2f17ff4abae875
(8) 0x81a76093909f956f41df98f45031f8f955dd3cd5
(9) 0x0321cd8cf7ea264b80d35b8ea170527426b609e2

Private Keys

=====

(0) a165bf13ba4d162de383678c25e025aa8971b704070914d9d2c6e2f445a4ec21
(1) 71dc41709cdcae1294c354d6b070a0842a843957c1411fcad6bbb5c46271a6c8
(2) d8f1bfa20b75247197ad6fd8721183538bbf218bc14d8478a79ee9771ba6bfbb
(3) 6aef0993cb99a0643ae99576322c00e119ad45553e373664548e739c2ecdb95c
(4) 1767ff888a588e0f3d9310600b2f9b5c0b6e2dc3127868a93c6f88a23ff16e21
(5) 6c075a710efa79d894e5210553b18e324271c4dfcb9a1b1a37ca965ab08aa136
(6) 8878d4b91868bc8eeb334b00131cf52ad1ec6096fc92d71d5644d7db9d6220db
(7) 58fc8768829adb8bb5fdec258bfbde16517c33e24ecb6c55054915d5955dbf0b
(8) cf46b39d80131761b10ceedd53fb4d625bb2d9f780d00d81dbeea1c49e6a46b2
(9) dfaff795cea2325005ca9aba2f633f6f9309e04d5000b5d597ef41642693e8ed

HD Wallet

=====

Mnemonic: party total update fatal kick below learn antenna spoil south oft
en cement

Base HD Path: m/44'/60'/0'/0/{account_index}

Listening on localhost:8545



Truffle

Truffle v4.1.7 – a development framework for Ethereum

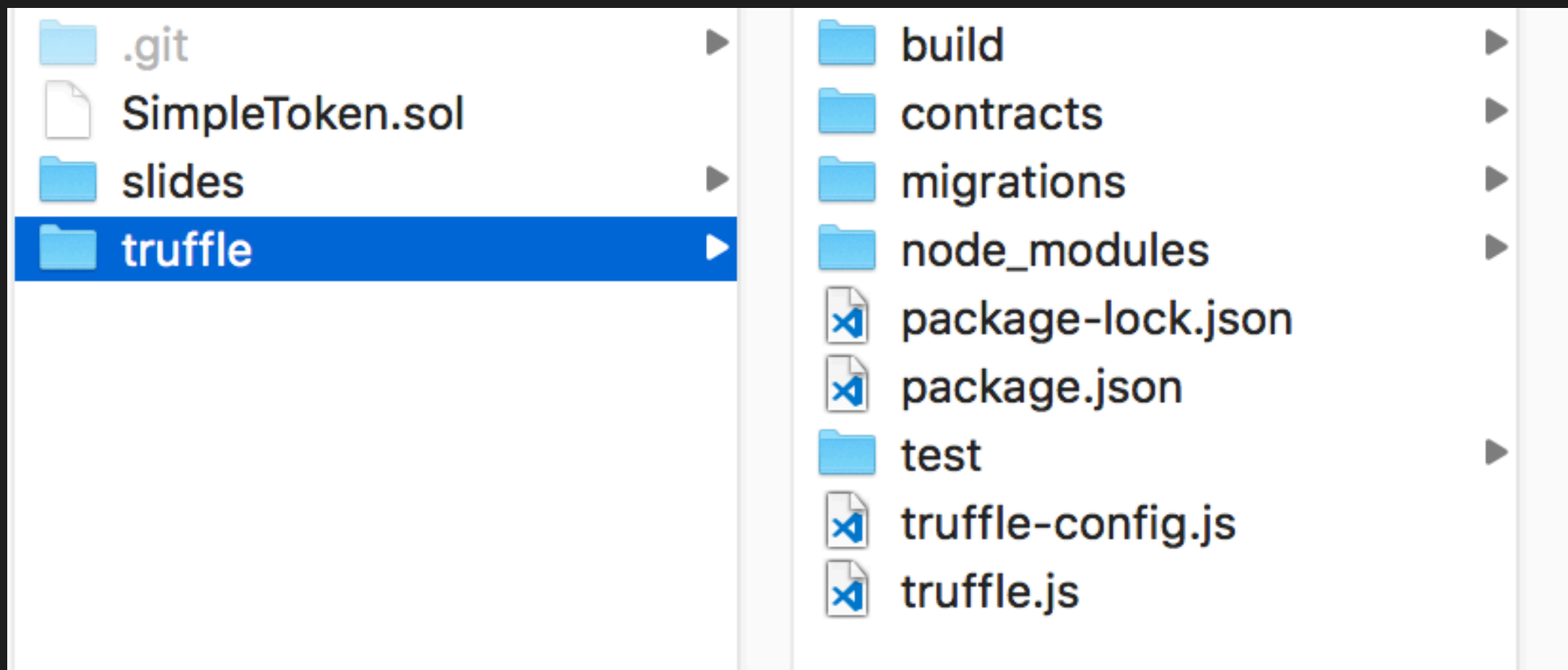
Usage: `truffle <command> [options]`

Commands:

<code>init</code>	Initialize new and empty Ethereum project
<code>compile</code>	Compile contract source files
<code>migrate</code>	Run migrations to deploy contracts
<code>deploy</code>	(alias for migrate)
<code>build</code>	Execute build pipeline (if configuration present)
<code>test</code>	Run JavaScript and Solidity tests
<code>debug</code>	Interactively debug any transaction on the blockchain (experimental)
<code>opcode</code>	Print the compiled opcodes for a given contract
<code>console</code>	Run a console with contract abstractions and commands available
<code>develop</code>	Open a console with a local development blockchain
<code>create</code>	Helper to create new contracts, migrations and tests
<code>install</code>	Install a package from the Ethereum Package Registry
<code>publish</code>	Publish a package to the Ethereum Package Registry
<code>networks</code>	Show addresses for deployed contracts on each network
<code>watch</code>	Watch filesystem for changes and rebuild the project automatically
<code>serve</code>	Serve the build directory on localhost and watch for changes
<code>exec</code>	Execute a JS module within this Truffle environment
<code>unbox</code>	Download a Truffle Box, a pre-built Truffle project
<code>version</code>	Show version number and exit

See more at <http://truffleframework.com/docs>

Truffle



Testing

- Happy path
- Code coverage
- Unit testing
- Journey testing



EXPLORER

OPEN EDITORS 2 UNSAVED



SimpleToken.sol 1

JS SimpleMintableToken.test.js



WORKSHOP_CO...

slides

truffle

build

contracts

migrations

node_modules

test

JS SimpleMintableToken.test.js

{ } package-lock.json

{ } package.json

JS truffle-config.js

JS truffle.js

SimpleToken.sol 1

SimpleToken.sol

JS SimpleMintableToken.test.js

```
1  contract('SimpleToken', function(accounts) {
2
3
4      const SimpleMintableToken = artifacts.require('SimpleMintableToken')
5
6      let token
7
8      beforeEach(async function () {
9          token = await SimpleMintableToken.new()
10      });
11
12      it("should return the correct totalSupply after construction", async function() {
13          let totalSupply = await token.totalSupply()
14          assert.deepEqual(totalSupply, new web3.BigNumber('100000000000000000000'))
15      })
16
17      it("should have correct balances after transfer", async function() {
18
19          let amount = '1000000000000000000'
20
21          let account0before = await token.balanceOf(accounts[0])
22
23          // Transfer
24          let transfer = await token.transfer(accounts[1], amount)
25
26          let event = transfer.logs[0].args
27
28          assert.equal(event.from, accounts[0])
29          assert.equal(event.to, accounts[1])
30          assert.equal(event.value.toString(), amount)
31
32          let account0After = await token.balanceOf(accounts[0])
33          let account1After = await token.balanceOf(accounts[1])
34
35          assert.deepEqual(account0After, new web3.BigNumber('999900000000000000000'))
36          assert.deepEqual(account1After, new web3.BigNumber(amount))
37
38      });
39
40      // Add test for minting
41      // Add test for any custom code
42      // Test any single lines of code
43      // Test boundaries eg attempting to send greater than balance
44      // Test expected default values are present and correct
45      // Test possible paths through a function
46
47      // And then...
48      // Post gist or pastie of working code in chat channel
49      // ```[code]``` Three back ticks should force
50
51  })
```

This page intentionally left blank

Transaction Receipts

[illegible]

Setup

- <https://nodejs.org/en/>
- <http://truffleframework.com/ganache/>
- `npm install -g truffle`
- [link to code]

Pair programming

Publish

- Run Geth
- Open Geth console
- Add private key from MetaMask
- Unlock private key
- Publish

Local node

```
geth --rinkeby --networkid=4 --datadir=$HOME/.rinkeby --cache=1024 --rpc --  
rpcapi="db,eth,net,web3,personal,web3, debug" --syncmode "light"
```

Import key

<https://ethereum.stackexchange.com/questions/465/how-to-import-a-plain-private-key-into-geth-or-mist>



Paste the key into a text file, save it to disk and use the path to that file with `geth account import .`. Here are some example Windows instructions that might help:

36



1. Open Notepad
2. Paste key into notepad without any extra characters or quotations
3. Save the file as `nothing_special_delete_me.txt` at `C:\`
4. Run the command, `geth account import C:\nothing_special_delete_me.txt`
5. After successful import, delete the file at `C:\nothing_special_delete_me.txt`

Unlock account

```
INFO [05-02|15:40:37] IPC endpoint opened           url=/Users/x/.rinkeby/geth.ipc
INFO [05-02|15:40:37] HTTP endpoint opened          url=http://127.0.0.1:8545
```

geth attach ipc:/some/custom/path

web3.personal.unlockAccount('<ethAddress>', '<pass>', 999999999)

Deploy

truffle deploy



Rinkeby
Test Net



Rinkeby
Test Net



ADD TOKEN

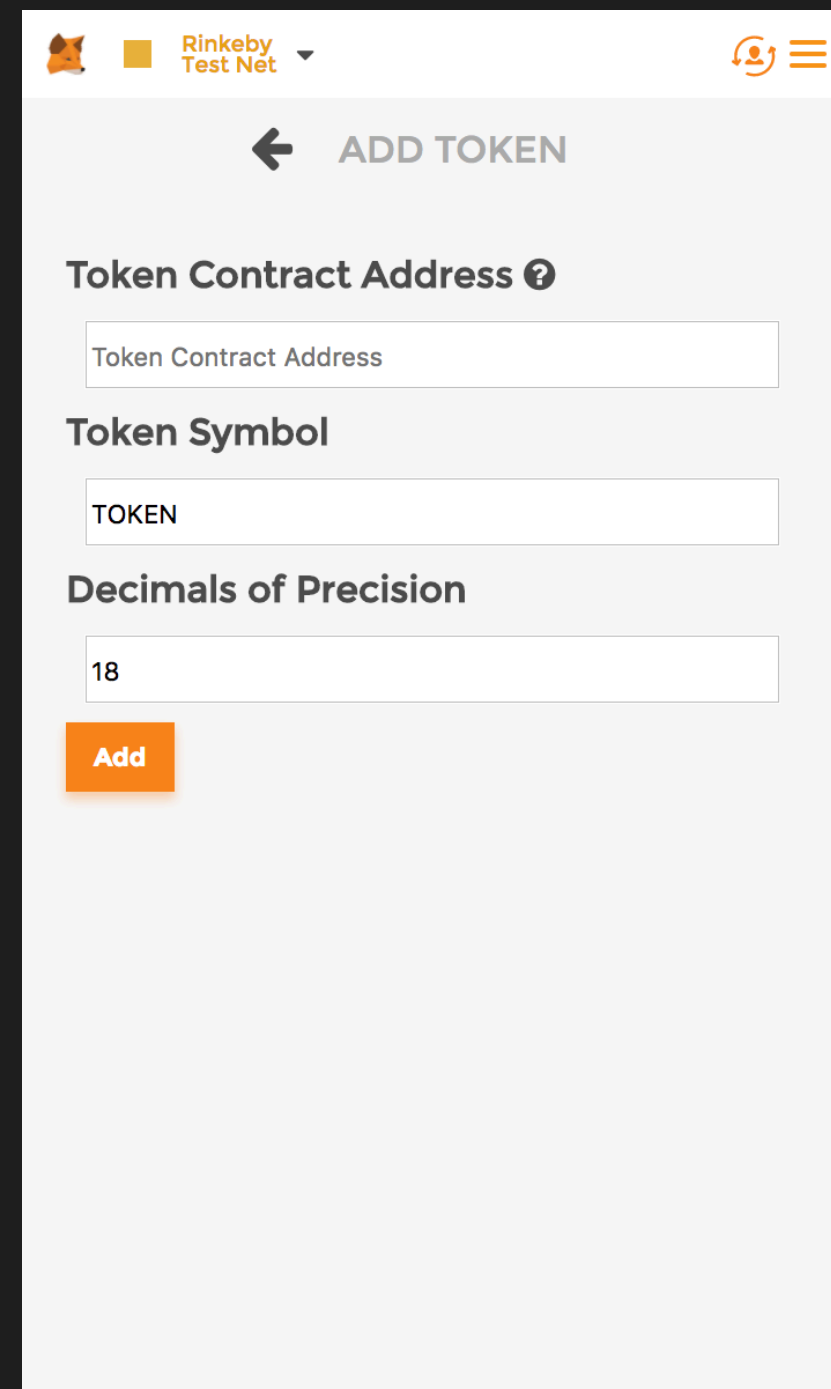
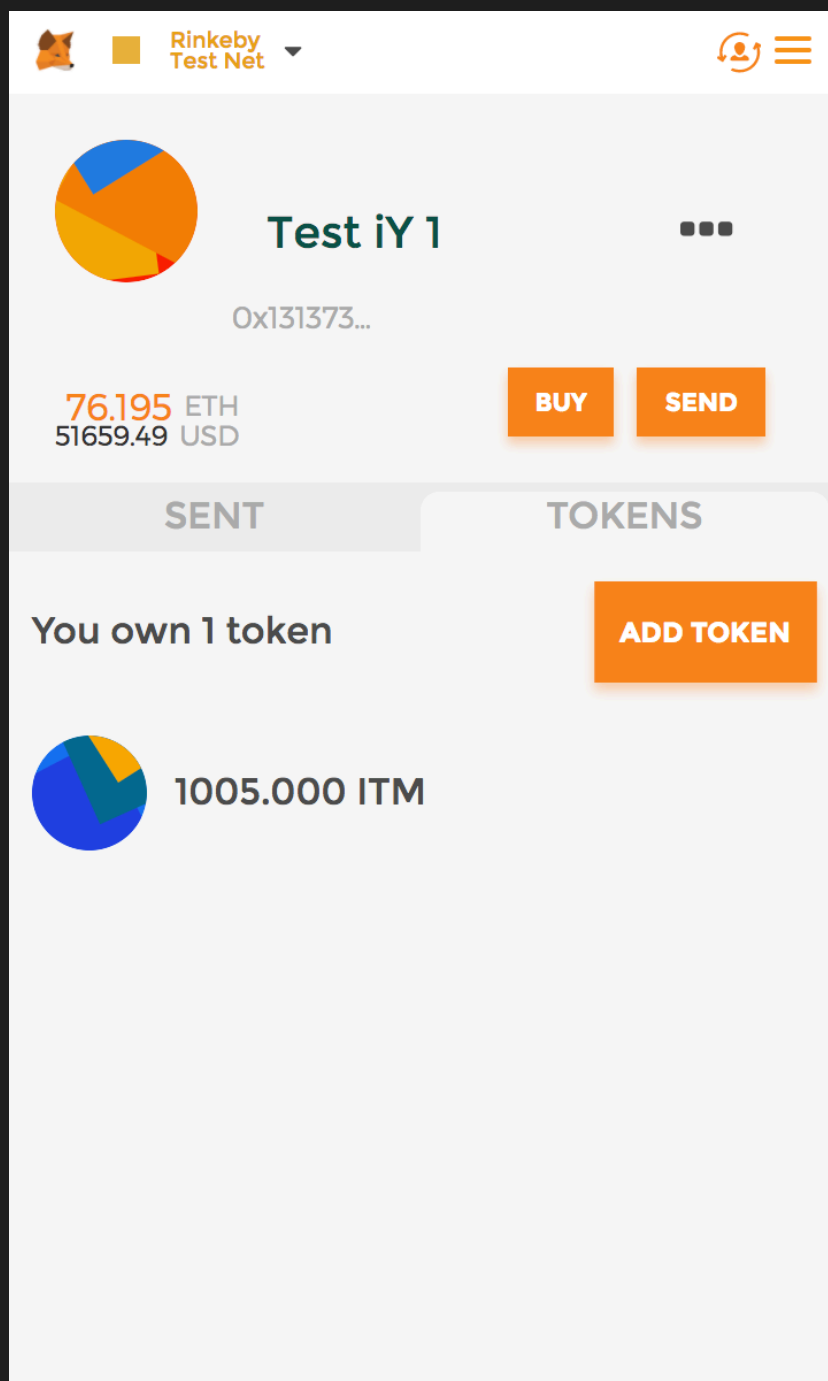
Token Contract Address ?

Token Symbol

Decimals of Precision

Add

View tokens



MyCrypto

MyCrypto, Inc [US] | https://mycrypto.com/#contracts



GAS PRICE
41 Gwei

NETWORK
ETH (mycryptoapi.com)

LANGUAGE
English

LATEST
Try the MyCrypto Beta!

New Wallet Send Swap Send Offline Contracts ENS DomainSale TX Status View Info Help

Interact with Contract or Deploy Contract

Contract Address

0x1313734d2D6625173278978DDaa7B63400462745



Select Existing Contract

Select a contract...

ABI / JSON Interface

```
[{ "type": "constructor", "inputs": [{ "name": "param1", "type": "uint256", "indexed": true }], "name": "Event" }, {  
  "type": "function", "inputs": [{ "name": "a", "type": "uint256" }], "name": "foo", "outputs": [] }]
```





Access

ABI

EXPLORER

OPEN EDITORS

- `{} SimpleMintableToken.json ...`

WORKSHOP_CO...    

- slides ●
- truffle ●
 - build
 - contracts
 - `{} BasicToken.json`
 - `{} Migrations.json`
 - `{} SimpleMintableToken.json`

```
{} SimpleMintableToken.json x
```

```
1  {
2    "contractName": "SimpleMintableToken",
3    "abi": [
4      {
5        "constant": true,
6        "inputs": [],
7        "name": "name",
8        "outputs": [
9          {
10           "name": "",
11           "type": "string"
12         }
13      ]
14    }
15  ]
16 }
```

Beer

:)

[]

• X

• X

• X