DECENTRALIZED
TRAINING SERIES

Day 1 - Session 3

Fundamentals on Ethereum Smart Contract

Part I

Klitos Christodoulou, Ph.D.

christodoulou.kl@unic.ac.cy

UNIC | Institute For the Future

CERTH CENTRE FOR RESEARCH & TECHNOLOGY HELLAS | iti Information Technologies Institute

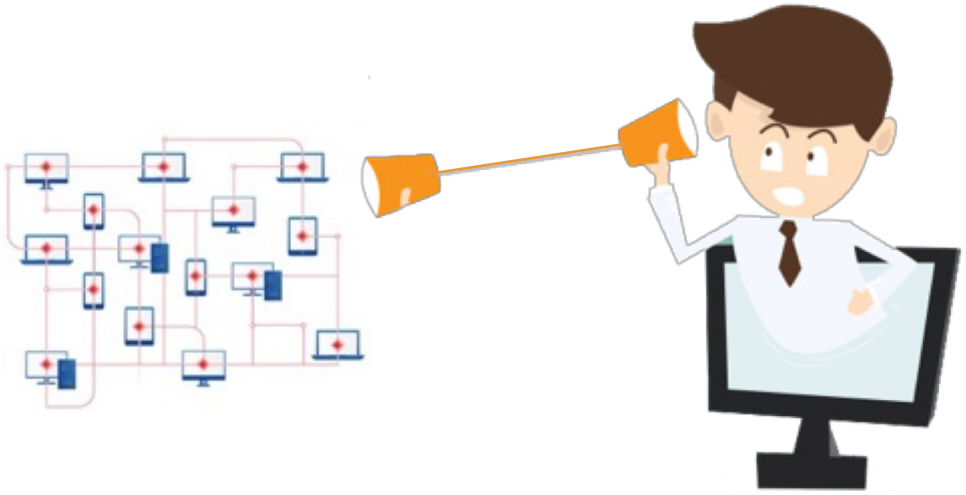
Agenda

- How do we interact with the Ethereum Network?
- The Mist Browser
- Metamask Installation
- Metamask Accounts
- Experimenting with Ethereum Networks
- Mnemonic Phrazes
- Gas System in Ethereum
- Transaction Objects in Ethereum
- Solidity vs Traditional Programming Languages
- Experimenting with Rinkeby

Advanced Features of Ethereum

Introduction

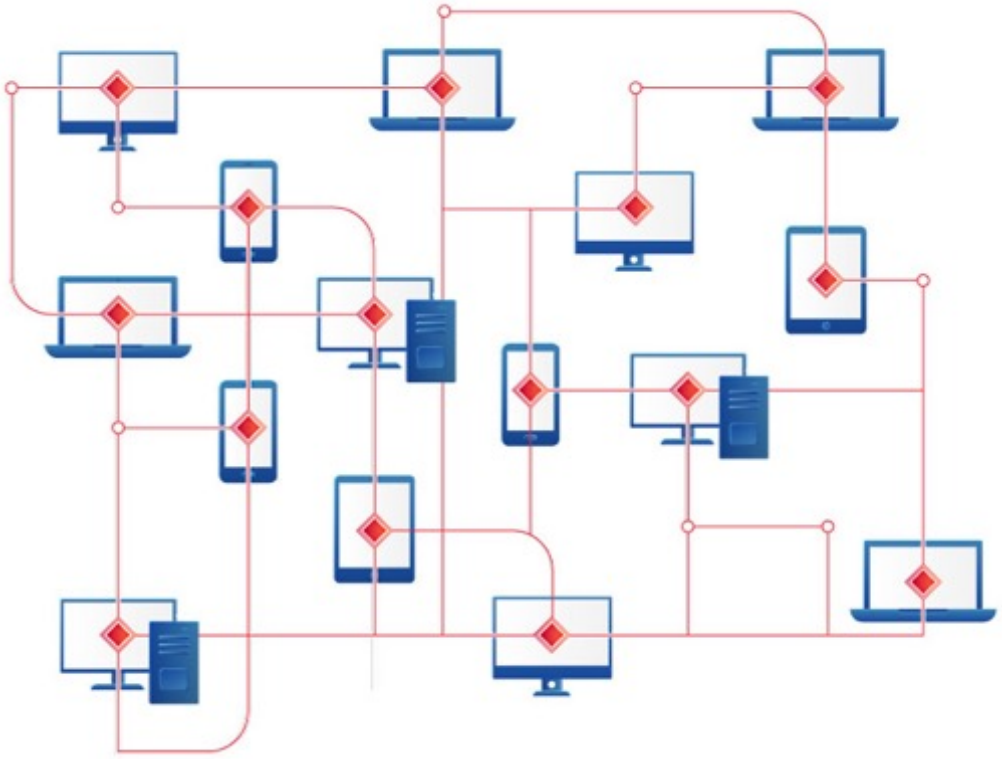
How do we interact with the Ethereum Network?



- How do we interact with the Network?
- How do we send money?
- How do we store data?
- How do we create smart contracts?

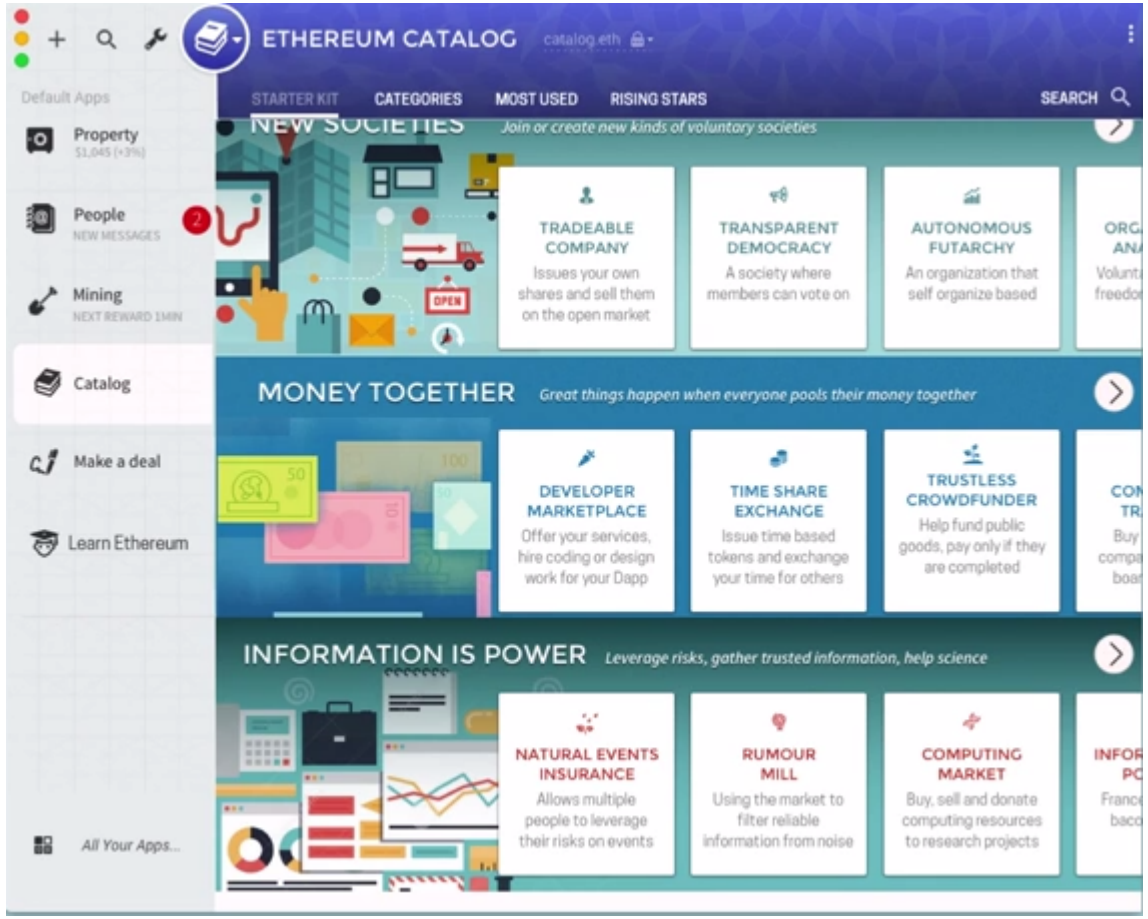
How do we interact with the Ethereum Network?

- Tools for Developers
  - web3.js
- Tools for Consumers
  - Metamask
  - Mist Browser



The Mist Browser

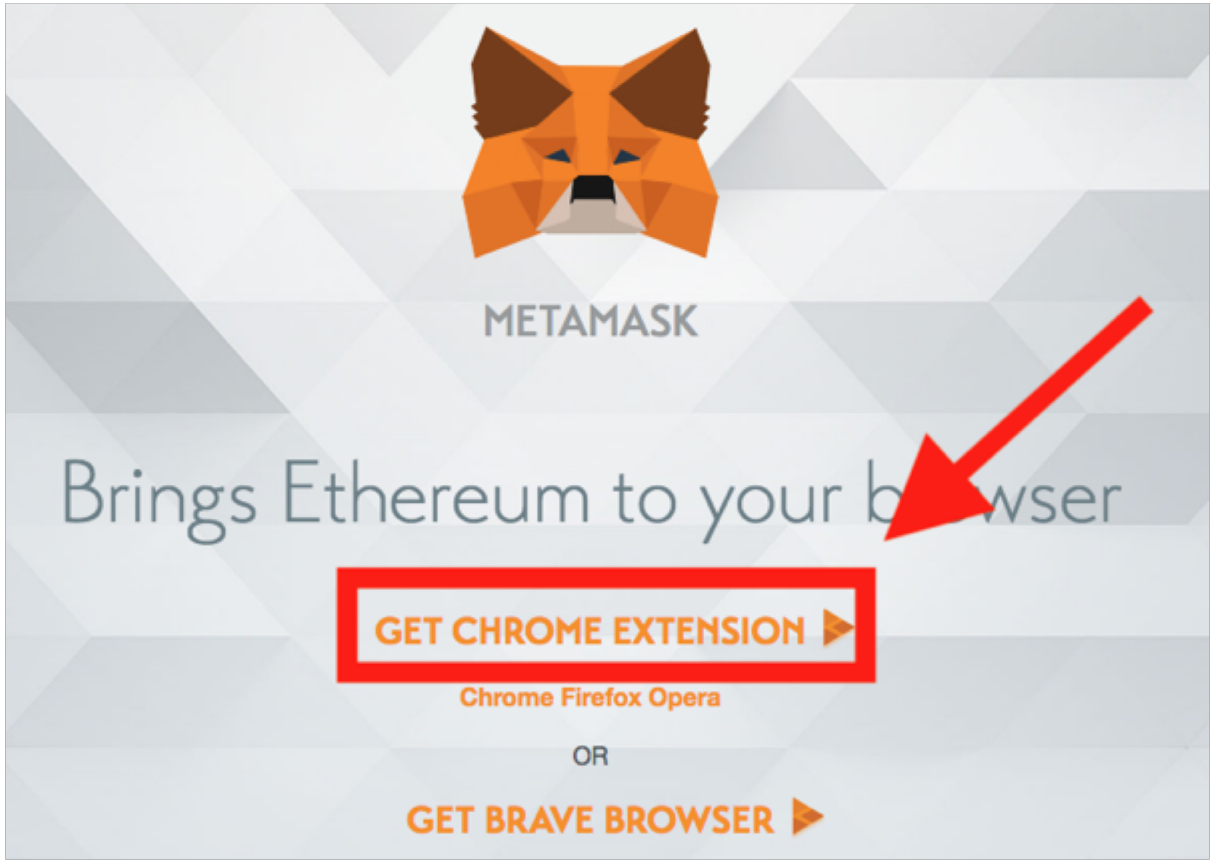
- Formerly known as the Ethereum Dapp Browser - is the end-user interface for Ethereum.
- A Graphical interface tool for browsing and using Dapps .
- Specifically designed for non-technical users.
- Requires a full node sync to run!
- Based on Decentralization of the p2p network.



https://github.com/ethereum/mist/releases

Metamask

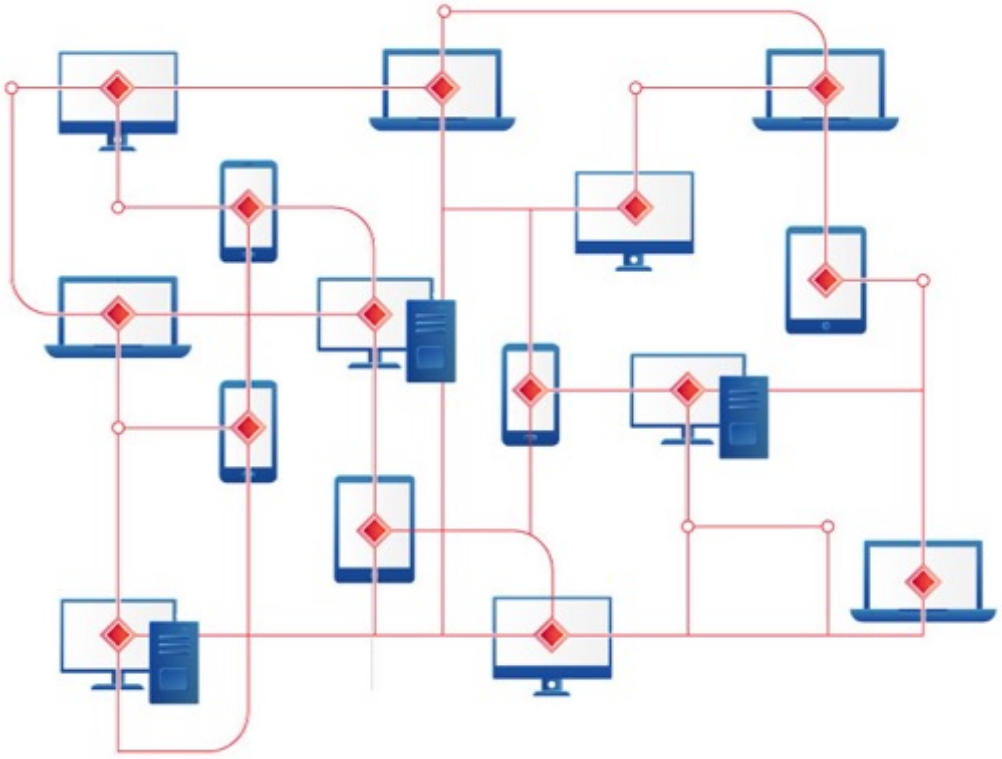
- Is an easy to use browser extension that enables browsing of Ethereum blockchain websites.
- Does not require you to run a full node.
- Relies on the Metamask server thus more centralized.
- May not support all ETH features.



https://metamask.io/

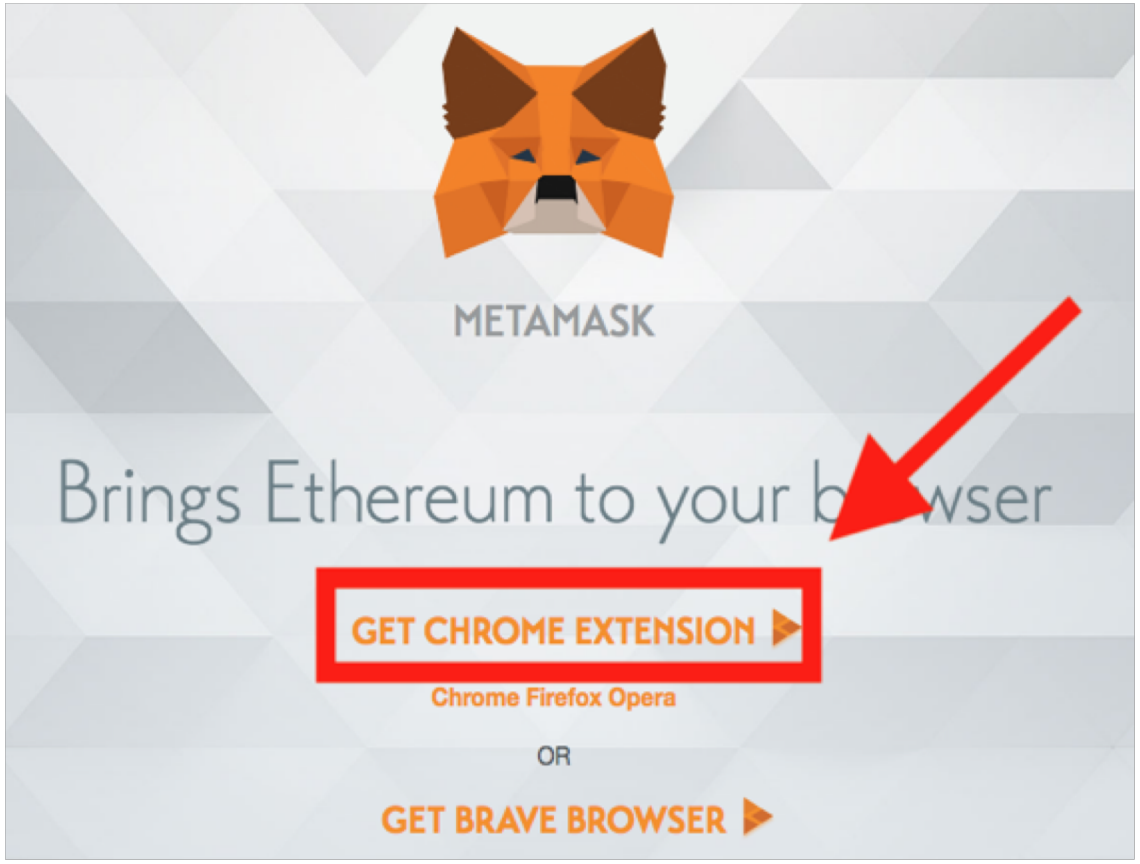
How do we interact with the Ethereum Network?

- Tools for Developers
  - web3.js
- Tools for Consumers
  - Metamask
  - Mist Browser



Hands-on: Install Metamask

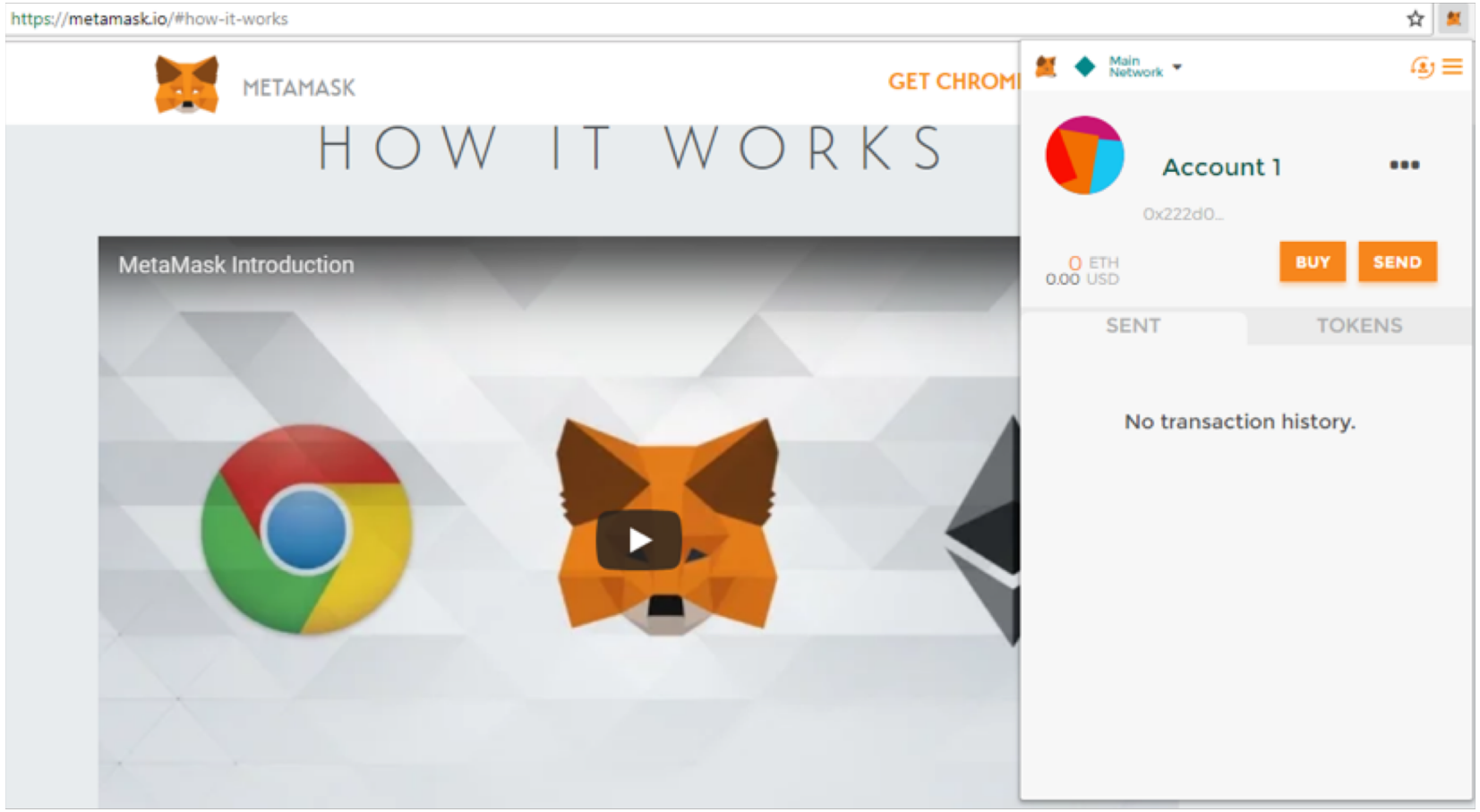
- Step 1: Navigate to
  - https://metamask.io
- Step 2: Click “Get Chrome Extension” to install Metamask.
- Step 3: Click “Add to Chrome” in the upper right.
- Step 4: Click “Add Extension” to complete the installation.



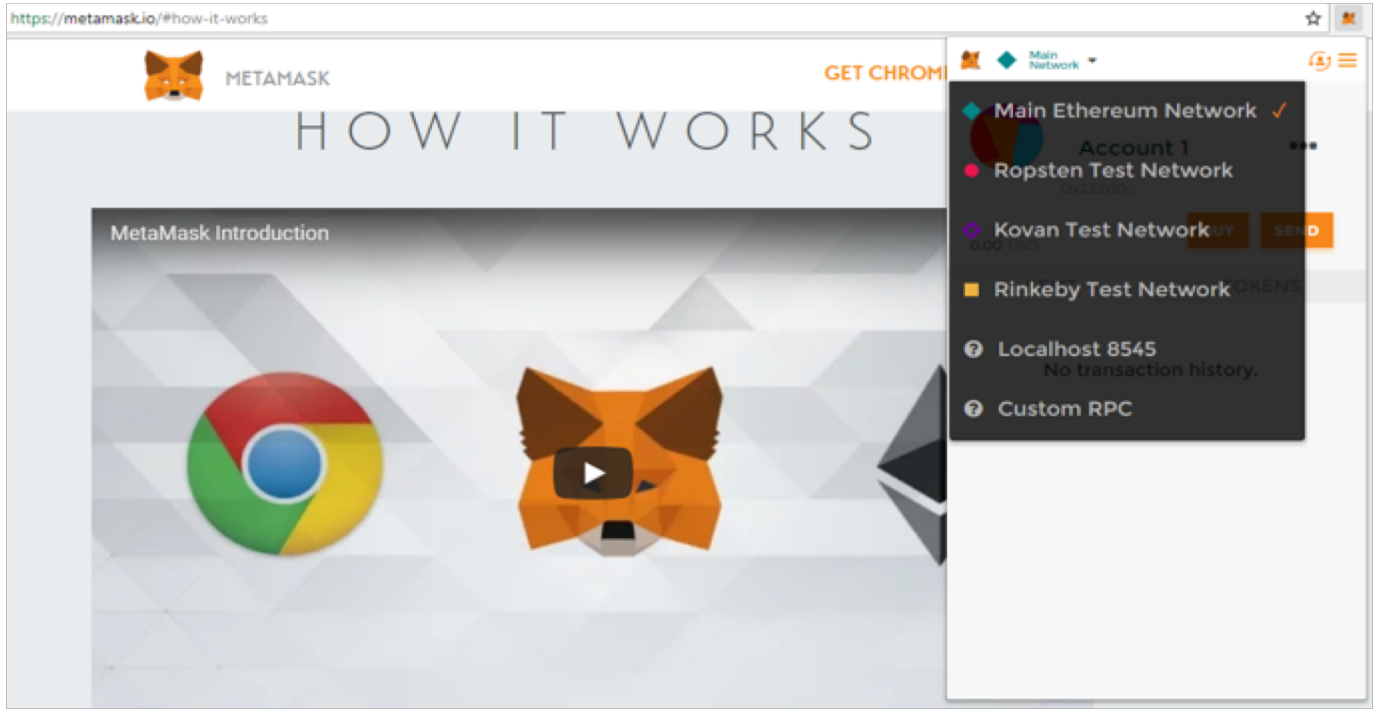
Hands-on: Using Metamask

- Aim: Now that we have installed Metamask on our browser, the next step is to create a wallet (or a “vault” using Metamask’s terminology), to do so follow these steps:
  - Step 1: Locate the Metamask logo in the upper right hand corner of your browser and click on it.
  - Step 2: Accept the terms and conditions.
  - Step 3: Enter a password and click the “Create” button to create your wallet.
  - Step 4: You will see a set of 12 “seed words” for your vault. Click “Save Seed Words as File” and copy the “MetaMask Seed Words” file that is downloaded to a safe place. You will need it to access/recover your vault (wallet) in the future.
  - Step 5: Metamask asks for confirmation click on “I’ve Copied It Somewhere Safe”

Congrats! Your are now connected on the Mainnet



Congrats! Your are now connected on the Mainnet



- Many Different Ethereum Networks to connect!
- From these Test Networks, you can safely “buy” and “send” test Ether from a faucet and begin experimenting with the blockchain.

What does it mean to have a Metamask Account?

- 3-pieces of information stored as Hex Numbers

Account Address

`0xcf512AeB3Cb4778DEFaC78B92E616F290D3d5448A9`

Public Key

`0x03f35f4efc18b3826ddb966a9a939293ab769b2eb`
`e11feb53ab4cc0506059eb094`

- Note: We can share the public key

Private Key

`0x178d5ec8d6e817813897bfeaae71f3eebbb6ceccd`
`a23d74c4ab7c31b98c9fcb4`

- Note: We do NOT share the private key

What does it mean to have a Metamask Account?

- 3-pieces of information stored as Hex Numbers

Account Address	`0xcf512AeB3Cb4778DEFaC78B92E616F290D3d5448A9`
Public Key	`0x03f35f4efc18b3826ddb966a9a939293ab769b2eb e11feb53ab4cc0506059eb094`
Private Key	`0x178d5ec8d6e817813897bfeaae71f3eebbb6ceccd a23d74c4ab7c31b98c9fcb4`

- Base 10 number = 4.574566855215592e+77
- Base 10 number = 1.06529751323271 19e+76

- It is incredibly unlikely that someone will guess your private key as this is a very large number, in fact you may have more chances to win a lottery than ever guess someone else's private key :-)

Experimenting with Ethereum Networks

- A distinct email account for each provider!

klitos@yahoo.com

pass_1

klitos_chr@gmail.com

pass_2

chrklitos@aol.com

pass_3

- On the Ethereum eco-system, **one** Account Address exists the same on all networks.
- One account is used **across all** Networks (we can have more that one account if we wish)

Account Address

Private Key

Public Key

Main Network

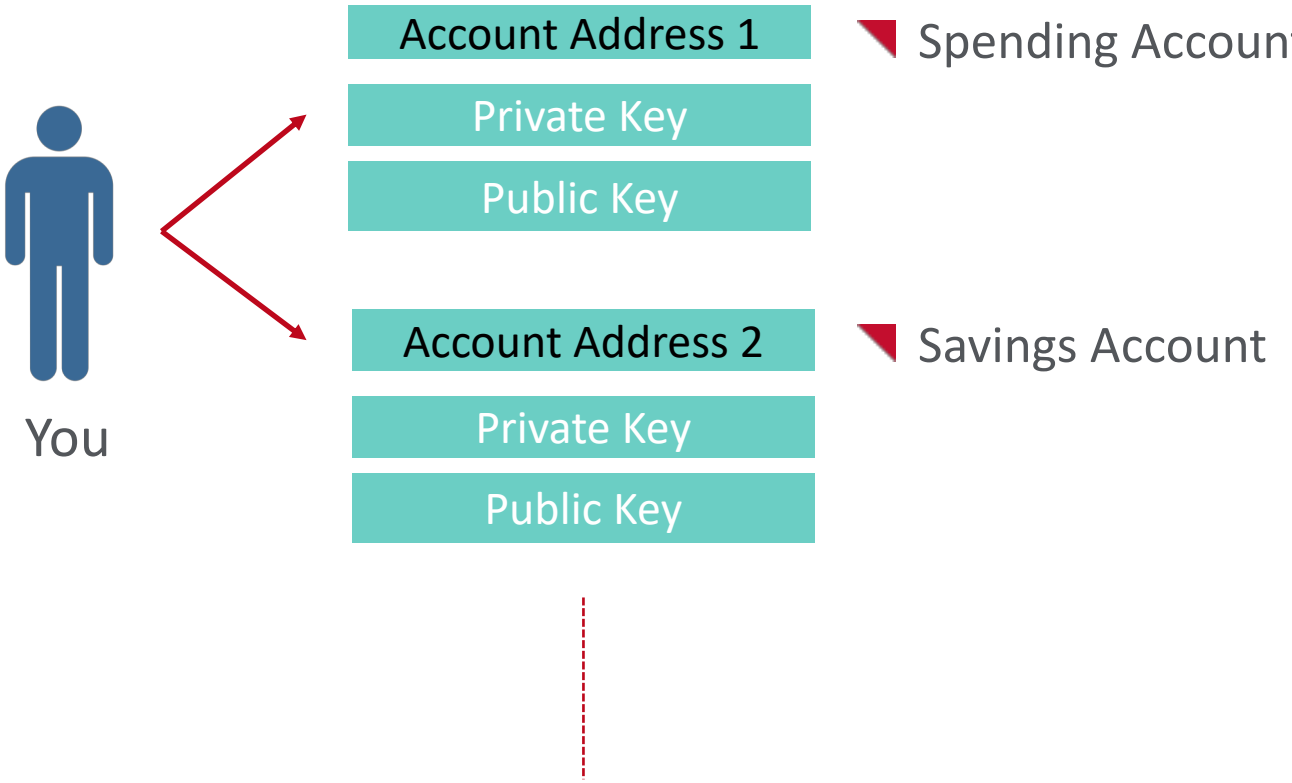
Rinkeby Network

Ropsten Network

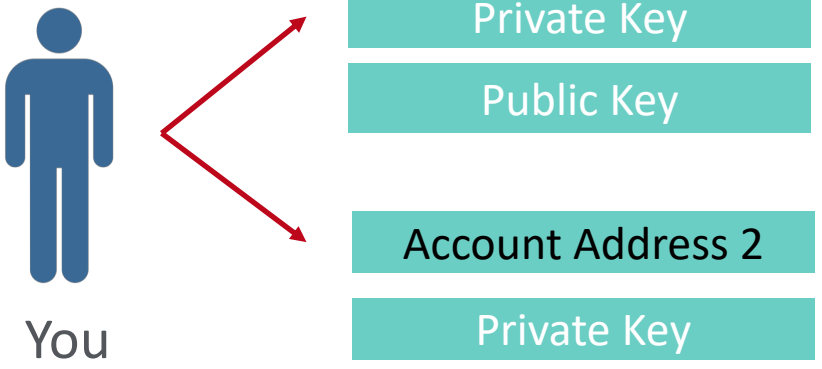
Kovan Network

Local Network

Mnemonic Phrazes: What about multiple accounts then?



Mnemonic Phrazes: What about multiple accounts then?



0x3B8c4c43a78e29805D849c1230892a58b1DE1012

0x178d5ec8d6e817813897bfeaae71f3eebbb6ceccda23d74c4ab7c31b98c9fcb4

0x03f35f4efc18b3826ddb966a9a939293ab769b2ebe11feb53ab4cc0506059eb094

0xdbca004E1B70378E2591145626f3c174d0252D1b

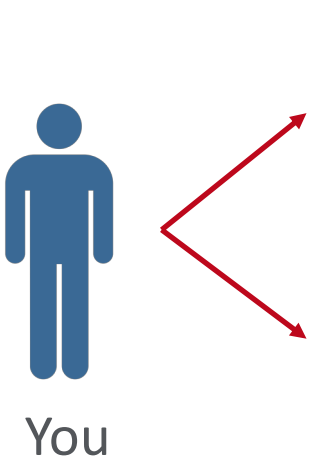
0x03680dc71cb84290dac868819e139e4d5e4b783f8d3ea8e7788f157ad91f7ea3a7

0xbad54e1201d88948b66f24932e486ef6beb8a48a418669abd481aa549fa9bfd3

What if there was a way to automatically generate multiple accounts with their own pair of <Public-Key, Private-Key>

Note: these addresses are derived from the BIP32 Extended Key

Mnemonic Phrazes: What about multiple accounts then?



Account Address 1

Private Key

Public Key

- Spending Account

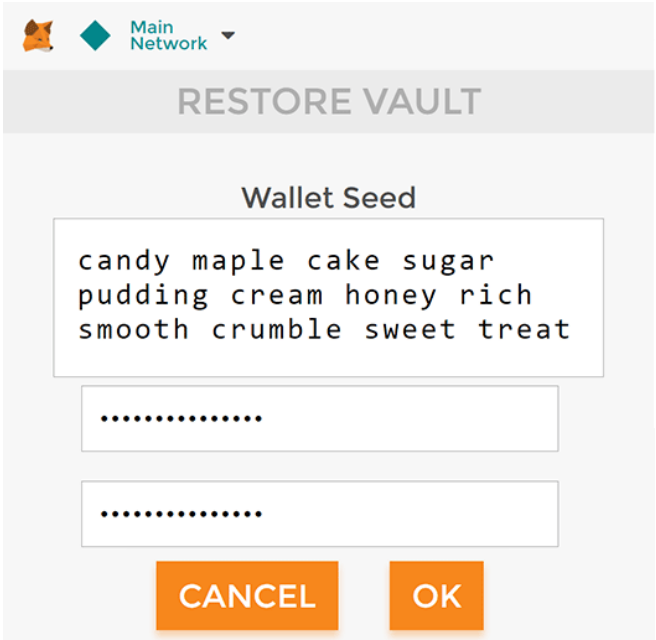
Account Address 2

Private Key

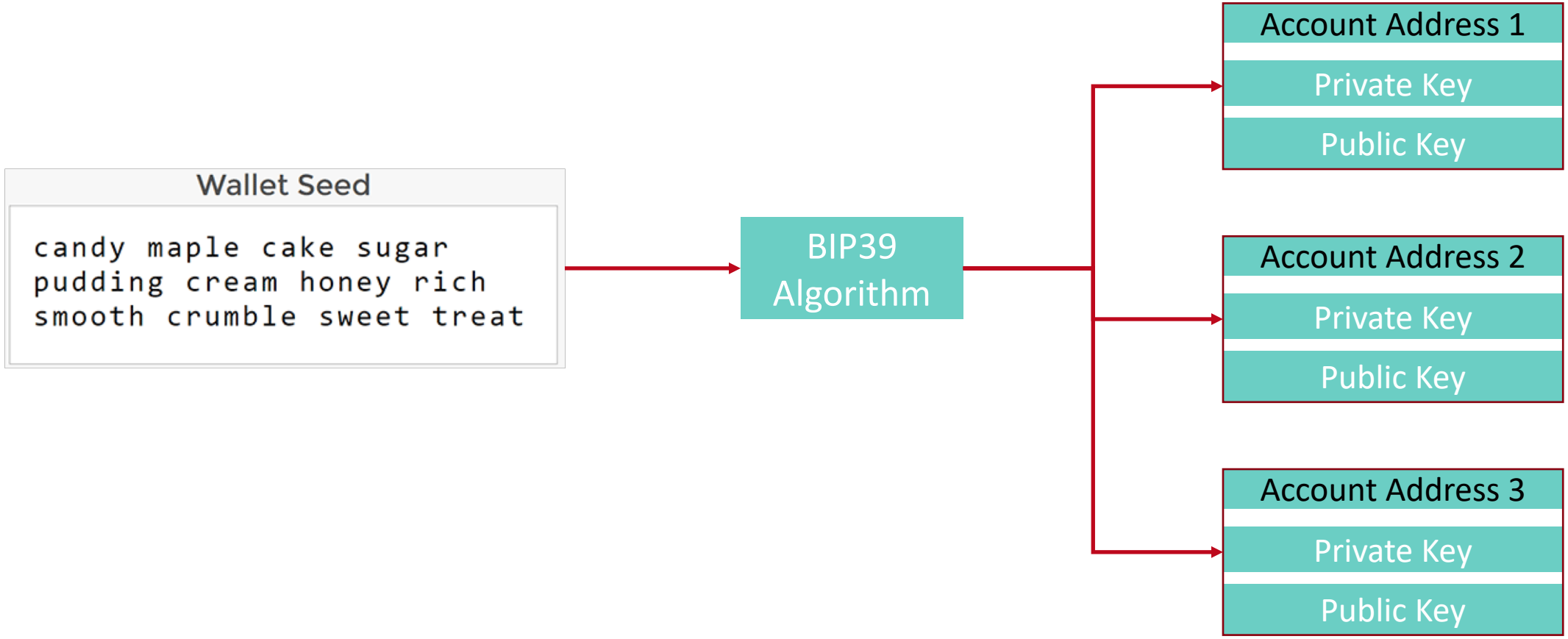
Public Key

- Savings Account

- Recommendation:
  - Memonic Phrazes (= A set of random words - seed)
  - Allows a user to generate a series of Accounts.



Mnemonic Phrazes: A solution to multiple accounts

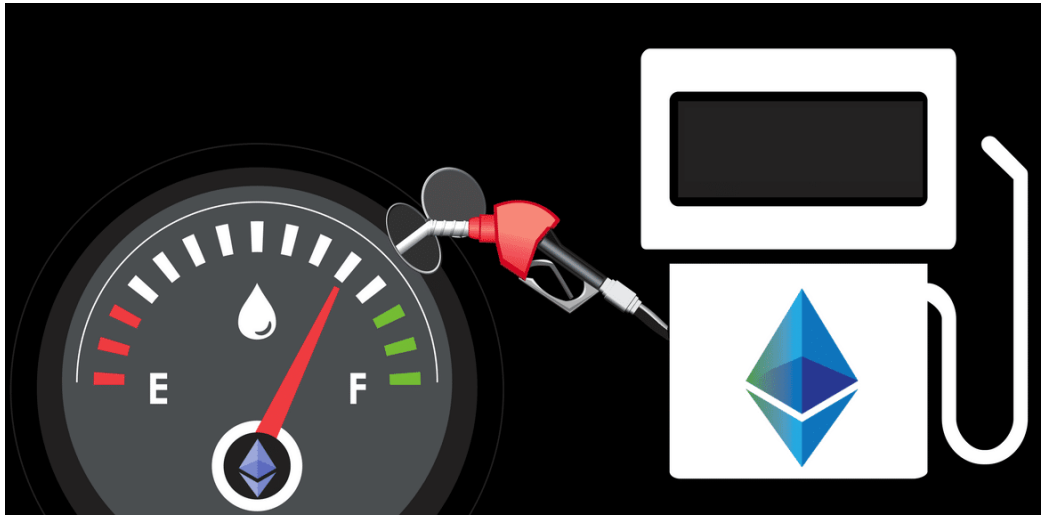


Gas System in Ethereum

Ether as fuel to the eco-system

Gas System: Ether as fuel

- You will probably heard the Gas System in Ethereum.
- This is based on a similar logic on how Amazon Web Services works
  - If you are using resources to run code you pay a fee
- Every time you send a transaction to the Ethereum Network we have to pay some amount of **Ether** as **Gas**.
  - A fraction of Ether most of the times!
- This is similar to the fuel our car needs to function but in this case we are using Ether as fuel.



Gas System: Ether Units of Measurement

- The most common Ether units used

Wei	10^{18}
Gwei	10^{9}
Finney	10^{3}
ether	1

Online tool for conversion:

https://etherconverter.online/

Gas System: Ether as fuel

- Let us demonstrate the notion of ”Gas” as a fuel with a working example:
  - Can we calculate the amount of gas we need to pay someone to run our code?

```
function doSimpleMath(int a, int b) {
   b – a;
   a + b;
   a * b;
   a == 15;
}
```

Gas Costs from EIP-150 Yellow Paper:

https://docs.google.com/spreadsheets/d/1n6mRqkBz3iWcOlRem_mO09GtSKEKrAsfO7Frgx18pNU/edit#gid=0

Gas System: Ether as fuel

- Let us demonstrate the notion of ”Gas” as a fuel with a working example:
  - Can we calculate the amount of gas we need to pay someone to run our code?

```
function doSimpleMath(int a, int b) {
    b – a;
    a + b;
    a * b;
    a == 15;
}
```

			*Gas
0x03	SUB	Subtraction operation.	3
0x01	ADD	Addition operation	3
0x02	MUL	Multiplication operation.	5
0x14	EQ	Equality comparison.	3

* Gas Costs: EIP-150 Yellow Paper

What does it mean for the Transaction Object?

- Before experimenting with Ethereum Smart Contracts we need some

nonce	An arbitrary number .e.g., of how many times the sender has send a transaction
to	The receiver's account (Public Key)
value	Amount of Ether to send
gasPrice	?
gasLimit	?
v	
r	
s	

- Complex pieces of Cryptographic stuff – these are generated by the sender's private key!

What does it mean for the Transaction Object?

- Before experimenting with Ethereum Smart Contracts we need some

nonce	An arbitrary number .e.g., of how many times the sender has send a transaction
to	The receiver's account (Public Key)
value	Amount of Ether to send
gasPrice	Amount of Wei (fraction of Ether) we are offering per unit of gas to get this transaction
gasLimit	The total number of gas units we are willing to spend with this transaction
v	
r	
s	

- Complex pieces of Cryptographic stuff – these are generated by the sender's private key!

Gas System: Ether as fuel

gasPrice	300 Wei
gasLimit	10 Gas Units x 300 Wei = 3000 Wei

```
function doSimpleMath(int a, int b) {
    b – a;
    a + b;
    a * b;
    a == 15;
}
```

			*Gas
0x03	SUB	Subtraction operation.	3
0x01	ADD	Addition operation	3
0x02	MUL	Multiplication operation.	5
0x14	EQ	Equality comparison.	3

* Gas Costs: EIP-150 Yellow Paper

Gas System: Ether as fuel

gasPrice	300 Wei
gasLimit	10 Gas Units x 300 Wei = 3000 Wei

- In order for the code to execute we need to spend at most 14 Gas Units and if we are willing to pay 300 Wei, thus: 14 x 300 = 4200 Wei is the cost

```
function doSimpleMath(int a, int b) {
    b – a;
    a + b;
    a * b;
    a == 15;
}
```

			*Gas
0x03	SUB	Subtraction operation.	3
0x01	ADD	Addition operation	3
0x02	MUL	Multiplication operation.	5
0x14	EQ	Equality comparison.	3

* Gas Costs: EIP-150 Yellow Paper

Invoking Functions in Ethereum Smart Contracts

- In reality there are **two** ways to **invoke** a **function** in Ethereum Smart Contracts with each way resulting in a different behavior.

Way 1 (just a call like in many traditional Programming Langs)	Way 2 (modify Contract's Data)
Just call a function – regular to any programming language	It requires a Transaction Object to be created and broadcast to the Network
It is **not** indented to modify the contract's data	It is indented to modify the contract's data
Runs **instantly** because it does not involve any transaction (no mining is required)	It requires **time** to execute – because a Transaction Object is broadcasted and need to be mined (*proof-of-work)*
Free to do! It is just to return data (e.g., getter method in Java)	***Not Free!*** *It costs money (i.e., ether)*

Solidity vs Traditional Programming Languages

- So every time we need or intend to modify or update the state of data in the contract's instance (like data in a Java Object) we need to submit a **Transaction Object**

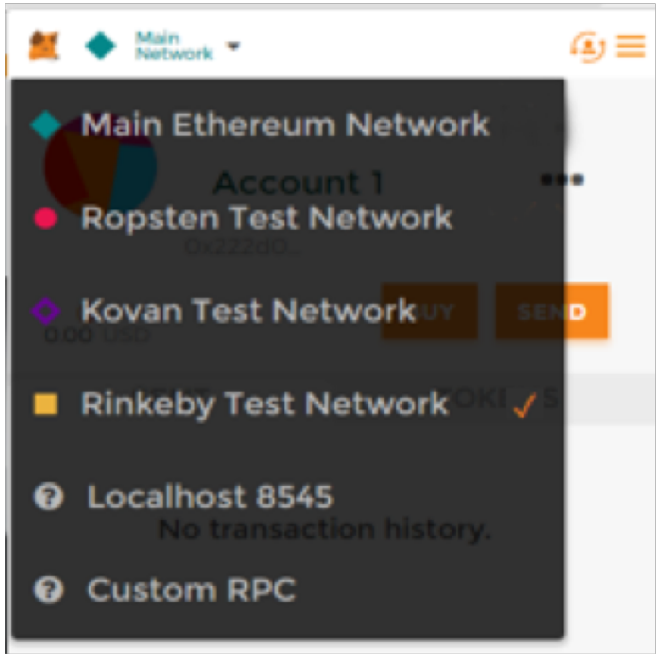
	Solidity Tip
	▪ Every time we submit a Transaction we need to wait for that Transaction to be mined (using proof-of-work) • This takes time (10-15 secs), it does not run instantly!

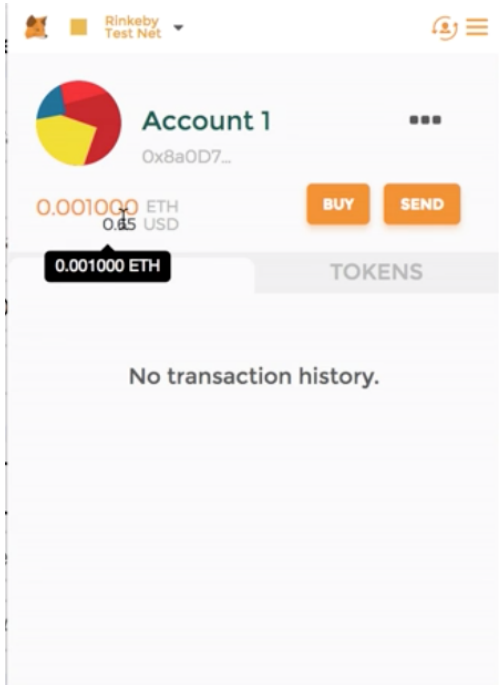
Experimenting with Rinkeby

Receive Ether from Rinkeby Faucet

Request Ether from the Rinkeby Faucet

- On **Metamask** make sure that we are connected to the Rinkeby Network
  - Navigate to: https://faucet.rinkeby.io/





- We then need to copy our Public Ethereum address and then request funds from either:
  - Twitter
  - Google +
  - Facebook
- How? By just posting our Public Key on one of the above channels.
- For surrounding text please use #DecentralizedTrainingSeries (any text is ok)
- Then copy the link from your Post to the text-box on faucet.rinkeby.io and choose “Give me Ether” :-)

Decentralized Training Series Workshop: Certified Ethereum Specialist

19-20 November 2018, Athens, Greece

UNIC | Institute For the Future

CERTH CENTRE FOR RESEARCH & TECHNOLOGY HELLAS | iti Information Technologies Institute