



# A gentle introduction to Blockchain

*at the TechConnex Blockchain Peer Group*

By Marc Lijour

February 28, 2019

Metamesh Group, your Trusted Advisors for Technology Transformation across the globe



The Blockchain Peer Group is brought to you by



# Who am I?

<https://www.linkedin.com/in/marclijour/>



**COLLIDER-X**



**CONSENSYS**



# Team Experience

 CONSENSYS	 THE WORLD BANK	 TOYOTA	 NORTEL NETWORKS™
 Celestica™	 AMERICAN EXPRESS	 NATO OTAN	 BlueCross® BlueShield®
 McDonald's	 CISCO™	 NOKIA Bell Labs	 SLA SINGAPORE LAND AUTHORITY
 Educational Testing Service	 IES INSTITUTE OF EDUCATION SCIENCES	 RioTinto	 BirdLife SOUTH AFRICA Giving Conservation Wings



Access these slides

<https://bit.ly/2RV6BzL>

or find by date:

<https://github.com/marclijour/presentations>



# Table of Contents

1 What is Blockchain?

2 Hands-on Introduction to Crypto

3 ICOs, STOs, and other Token launches

4 Introduction to Smart Contracts



# The Trust Machine



# Source of Trust

World Economic Forum: *What is Blockchain* Youtube Video

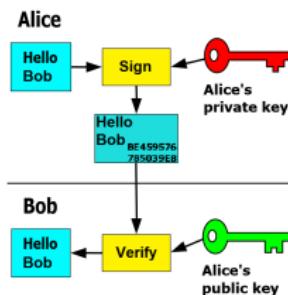


BLOCKCHAIN WILL BECOME A GLOBAL  
DECENTRALISED SOURCE OF TRUST



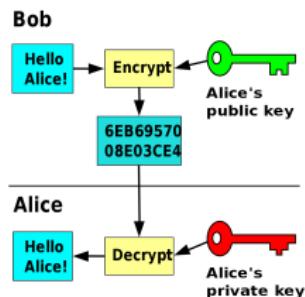
# Key Cryptographic Primitives

## Signing

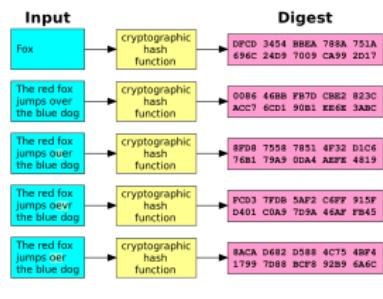


Credit: FlippyFlink

## Encrypting



## Hashing



# Where do we need more trust?

- payments



# Where do we need more trust?

- payments
- national identity



# Where do we need more trust?

- payments
- national identity
- supply chain (e.g. food: Spanghero horse meat trial, Opioids, expensive cargos)



# Where do we need more trust?

- payments
- national identity
- supply chain (e.g. food: Spanghero horse meat trial, Opioids, expensive cargos)
- contracts



# Where do we need more trust?

- payments
- national identity
- supply chain (e.g. food: Spanghero horse meat trial, Opioids, expensive cargos)
- contracts
- *real* news & historical events (e.g. bombings)



# Where do we need more trust?

- payments
- national identity
- supply chain (e.g. food: Spanghero horse meat trial, Opioids, expensive cargos)
- contracts
- *real* news & historical events (e.g. bombings)
- collaborative data reporting
- ...



# Table of Contents

- 1 What is Blockchain?
- 2 Hands-on Introduction to Crypto
- 3 ICOs, STOs, and other Token launches
- 4 Introduction to Smart Contracts



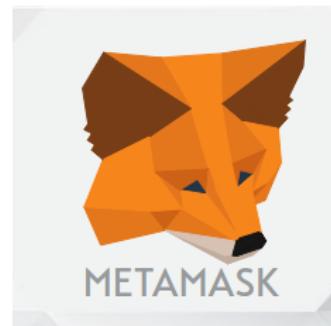
# Let's try things out!



# Install MetaMask

Follow step by step:

- ① Install the [Chrome/Chromium extension](#)
- ② Watch the [intro on Youtube](#)
- ③ Create an account
- ④ Switch to the Ropsten Testnet  
(top-right in MetaMask)
- ⑤ Fill your account with Ether from  
<https://faucet.metamask.io>



<https://metamask.io>



# Request Ether from the faucet (on the Ropsten network)

Do it several times; then donate 1 ether to the faucet

← → ⌛ 🔒 https://faucet.metamask.io

## MetaMask Ether Faucet

faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647  
balance: 4094302.54 ether

[request 1 ether from faucet](#)

user

address: 0x56e552eb5a9ab277d9eb841f92d473bf0cae7ebf  
balance: 1.00 ether  
donate to faucet:

[1 ether](#) [10 ether](#) [100 ether](#)

transactions



# Check the transaction on Metamask

Click on the transaction for a detailed view

The screenshot shows the Metamask wallet interface. At the top, it displays "0.9999 ETH" and "\$88.94 USD". Below this, there's a "History" section with a single entry: "Sent Ether" from address #0 to address 0x81b7E08F65Bdf5648606c89998A9CC8164397647 at 23:23 on 12/11/2018, confirmed. The transaction details show an amount of 1 ETH, gas limit of 21000, gas used of 21000, and a gas price of 5 GWEI. The total cost was 1.000105 ETH or \$88.96 USD. An "Activity Log" section lists three events: the creation of the transaction, its submission, and its confirmation. A "Details" section provides a breakdown of the transaction fees. On the left, there's a sidebar with a profile picture for "Marc 1", a balance of 0.9999 ETH, and a link to "View on Etherscan". A message encourages users to "Add Token" if they don't see their tokens.



# Check the transaction on Etherscan



ROPSTEN

ROPSTEN (Revival) TESTNET

Search by Address / Txhash / Block / Token / Ens

GO

Language

HOME

BLOCKCHAIN

TOKEN

MISC

Transaction 0xa8a8f32539cb69bd3e506b6527fcc7d44c43ccead09b30b10a9891c99355a3d

Home / Transactions / Tx Info

Overview

Transaction Information



Tools &amp; Utilities

[ This is a Ropsten Testnet Transaction Only ]

TxHash: 0xa8a8f32539cb69bd3e506b6527fcc7d44c43ccead09b30b10a9891c99355a3d

TxReceipt Status: Success

Block Height: 4610407 (348 Block Confirmations)

TimeStamp: 1 hr 12 mins ago (Dec-12-2018 04:23:14 AM +UTC)

From: 0x56e552eb5a9ab277d9eb841f92d473bf0cae7ebf

To: 0xb1b7e08f65bdf5648606c89998a9cc8164397647

Value: 1 Ether (\$0.00)

Gas Limit: 21000

Gas Used By Transaction: 21000 (100%)

Gas Price: 0.000000005 Ether (5 Gwei)

Actual Tx Cost/Fee: 0.000105 Ether (\$0.000000)

Nonce &amp; {Position}: 0 | {6}

Input Data:

0x



# A note about gas price

<https://ethgasstation.info>

ETH Gas Station

Estimates over last 1,500 blocks - Last update: Block 5164391

Change Currency ▾

**Std Cost for Transfer** \$0.056

**Gas Price Std (wei)** 3

**SafeLow Cost for Transfer** \$0.056

**Gas Price SafeLow (wei)** 3

**Median Wait (s)** 29

**Median Wait (blocks)** 2

**Gas-Time-Price Estimator:** For transactions sent at block: 5164391

Adjust confirmation time

Avg Time (m/s)	4.38
95% Time (m/s)	10.95
Gas Price (Wei)*	3
Tx Fee (Flat)	\$0.056

Gas Used*	21000
Avg Time (blocks)	18.02
95% Time (blocks)	45.05
Tx Fee (ETH)	0.00005

**Real Time Gas Use: % Block Limit (last 10)**

Last Block: 5164391

**Transaction Count by Gas Price**

**Confirmation Time by Gas Price**

**Recommended Gas Prices** (based on current network conditions)

Speed	Gas Price (wei)
SafeLow (<30m)	3
Standard (<5m)	3
Fast (<2m)	18

Note: Estimates not valid when multiple transactions are batched from the same address or for transactions sent to addresses with many (e.g. > 100) pending nonce confirmations.

**Misc Stats (Last 1,500 blocks)**



# Price of (real) ether: ETH

More information: <https://www.tradingview.com/symbols/ETHUSD/>

ETHUSD Crypto Chart



# Wallets

More information: <https://blockgeeks.com/guides/cryptocurrency-wallet-guide/>

Software



Hardware



Paper



# Exchanges

- ① Centralized Exchanges (Coinbase, Quadriga, ...)
- ② Decentralized Exchanges

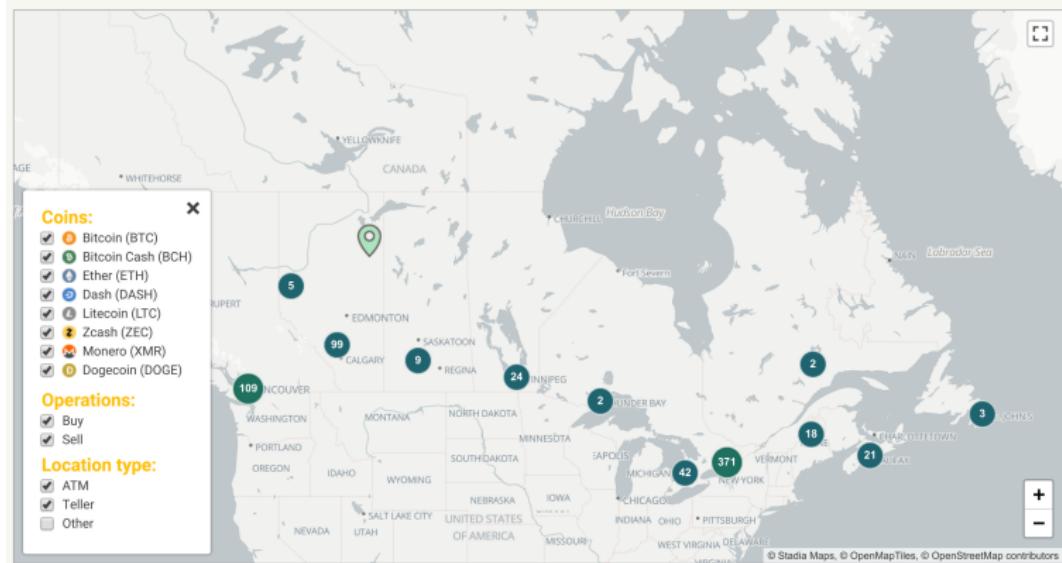


# ATMs

More information: <https://coinatmradar.com/country/38/bitcoin-atm-canada/>

## Bitcoin ATMs in Canada. 🇨🇦

Total number of Bitcoin ATMs / Tellers in Canada: 707

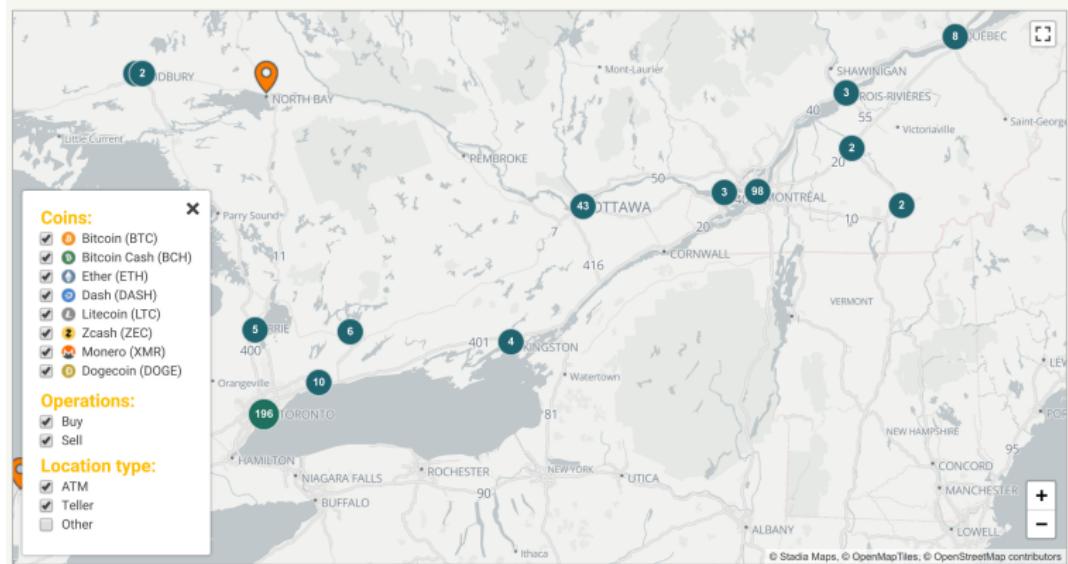


# ATMs

More information: <https://coinatmradar.com/country/38/bitcoin-atm-canada/>

## Bitcoin ATMs in Canada . 🇨🇦

Total number of Bitcoin ATMs / Tellers in Canada: 707



# Other crypto-assets

More information:

<https://www.tradingview.com/markets/cryptocurrencies/prices-all/>

NAME 301 matches	SEARCH	WKT CAP	FD WKT CAP	LAST	AVAIL COINS	TOTAL COINS	TRADED VOL	CHG %
Bitcoin		59.852B	72.158B	3436.1	17.419M	21M	4.712B	0.02%
XRP		12.167B	29.729B	0.29729	40.927B	100B	369.224M	0.17%
Ethereum		9.257B	9.257B	89.23	103.745M	101.745M	1.739B	0.10%
Stellar		2.165B	11.811B	0.11297826	19.165B	104.543B	197.41M	-0.34%
Tether		1.838B	2.554B	0.99000000	1.856B	2.58B	2.97B	-0.10%
Bitcoin Cash		1.729B	2.074B	98.760	17.506M	21M	71.709M	0.92%
EOS		1.692B	1.879B	1.8672	906.245M	1.006B	740.947M	0.44%
Bitcoin SV		1.548B	1.858B	88.45300000	17.505M	21M	59.813M	-2.60%
Litecoin		1.431B	2.028	24.045	59.532M	84M	380.518M	1.14%
TRON		877.572M	1.314B	0.013247	66.247B	99.218B	72.465M	-0.88%
Cardano		772.388M	1.341B	0.029791	25.927B	45B	9.766M	-0.09%
Monero		722.858M	722.858M	43.445	16.638M	16.638M	11.502M	1.38%
NEM		642.663M	642.663M	0.07141	9B	9B	8.23M	-1.51%
Binance Coin		635.009M	926.299M	4.8548350	130.799M	190.799M	20.772M	-2.52%
IOTA		631.509M	631.509M	0.2272	2.78B	2.78B	4.203M	1.07%
Dash		554.925M	1.234B	65.28257000	8.5M	18.9M	160.684M	2.12%
Ethereum Classic		402.997M	402.997M	3.777	106.698M	106.698M	106.521M	0.68%
NEO		387.602M	596.31M	5.96	65M	100M	100.484M	1.51%



# Table of Contents

- 1 What is Blockchain?
- 2 Hands-on Introduction to Crypto
- 3 ICOs, STOs, and other Token launches
- 4 Introduction to Smart Contracts

# Ethereum, Alt-Coins, and ICOs

- ① 94% of the top 100 tokens are built on top of Ethereum
- ② \$5.5 billion raised in 2017
- ③ \$6.5 billion raised in the first quarter of 2018
- ④ Ethereum has the largest developer community
- ⑤ Close to 2,000 dapps (decentralized applications) counted on Ethereum



# Amount raised by ICOs

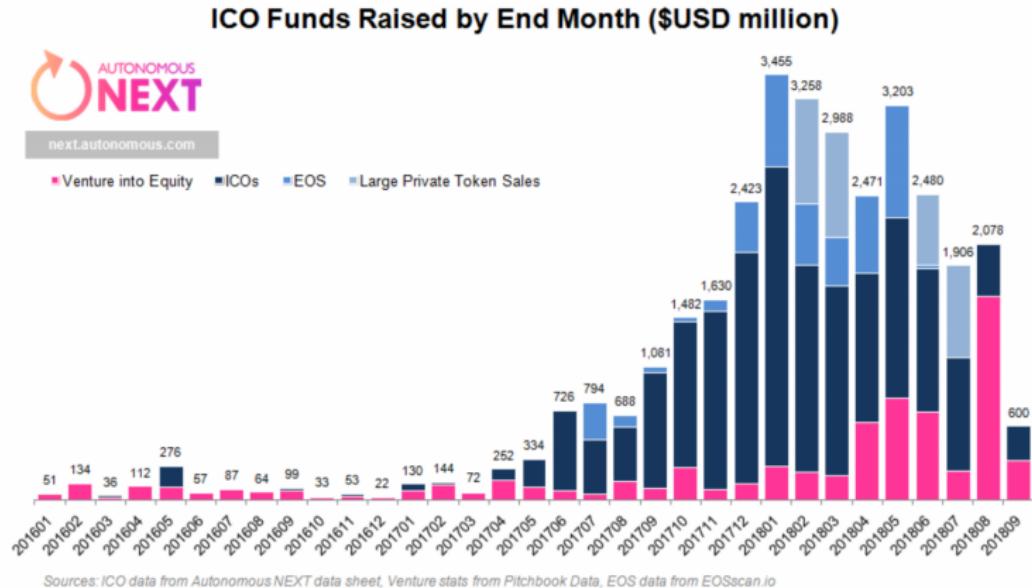
<https://cointelegraph.com/news/>

from-2-9-billion-in-a-month-to-hundreds-dead-trends-of-the-rollercoaster-ico-market-in-18-months



# ICOs fell down 90% in 2018

<https://www.investinblockchain.com/invest-in-icos/>



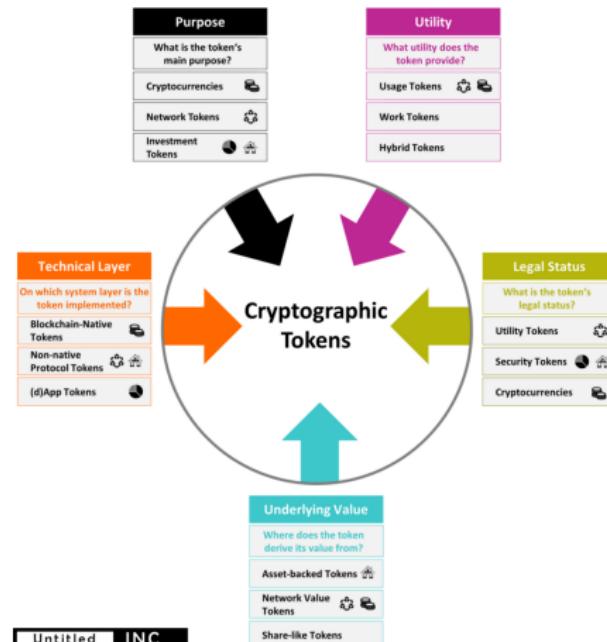
see also <https://www.coindesk.com/dont-throw-the-crypto-tokens-out-with-the-bathwater>



# Token Types

<http://www.untitled-inc.com/>

the-token-classification-framework-a-multi-dimensional-tool-for-understanding-and-classifying-crypto-tokens/



Untitled INC



# Token Types

<http://www.untitled-inc.com/>

the-token-classification-framework-a-multi-dimensional-tool-for-understanding-and-classifying-crypto-tokens/

MAIN TOKEN TYPES PER DIMENSION				
Technical Layer	Purpose	Underlying Value	Utility	Legal Status*
<b>Blockchain-Native Tokens</b>  Description: A token that is implemented on the protocol-level of a blockchain.  Characteristics: <ul style="list-style-type: none"><li>• Critical to operate the blockchain</li><li>• Integral component of the blockchain's consensus mechanism</li><li>• Part of the protocol's incentive mechanisms for block validators/fee nodes</li></ul> Examples: BTC (Bitcoin), Bitcoin; ETH (Ether, Ethereum), STEEM (Steem, Steemit)	<b>Cryptocurrencies</b>  Description: A token that is intended to be a "pure" cryptocurrency  Characteristics: <ul style="list-style-type: none"><li>• Intended as a global medium of exchange</li><li>• Functions as a store of value</li></ul> Examples: BTC (Bitcoin), LTC (Lisk), KNC (Ku, KNC)	<b>Asset-backed Tokens</b>  Description: A token that functions as a claim on an underlying asset  Characteristics: <ul style="list-style-type: none"><li>• Allows trading via IOUs without physically having to move the underlying asset</li><li>• The issuer is responsible to hold the assets in escrow</li><li>• Introduces counterparty risk</li></ul> Examples: USDT (Tether USD, Tether), GOLD (Gold, GoldMint), Ripple (XRP, Ripple)	<b>Usage Tokens</b>  Description: A token that provides access to a digital service, similar to a paid API key  Characteristics: <ul style="list-style-type: none"><li>• Grants holders access to exclusive functionality of the service</li></ul> Examples: BTC (Bitcoin), STX (Stox, Blockstack)	<b>Utility Tokens</b>  Description: A token offering owners clearly defined utility within a network or platform application  Characteristics: <ul style="list-style-type: none"><li>• Closely tied to the functionality of the issuing network or application</li><li>• Internal network/app currency but not necessarily a general purpose currency</li><li>• Grants owners the right to actively contribute to the system vs. passive investors</li><li>• Aren't security-like features</li></ul> Examples: GNO (Gnosis), STEEM (Steem)
<b>Non-native Protocol Tokens</b>  Description: A token that is implemented in a crypotnomic protocol on top of a blockchain  Characteristics: <ul style="list-style-type: none"><li>• Integral component of the protocol's consensus mechanism</li><li>• Part of the protocol's incentive mechanisms for nodes</li><li>• Not directly part of the blockchain to which it is not integral (e.g. ERC20 Tokens on Ethereum)</li></ul> Examples: REP (Decentralized Oracle Protocol, Augur)	<b>Network Tokens</b>  Description: A token that is tied to the value and development of a network system (e.g. network, application)  Characteristics: <ul style="list-style-type: none"><li>• Token functionality within the system</li><li>• Not intended as a general cryptocurrency</li></ul> Examples: GNO (Gnosis), STX (Stacks, Blockstack)	<b>Network Value Tokens</b>  Description: A token that is tied to the value and development of a network system  Characteristics: <ul style="list-style-type: none"><li>• Tied to the value generated and distributed on the network (e.g. transaction fee volume)</li><li>• Closely intertwined with key interactors of network participants</li></ul> Examples: ETH (Ether, Ethereum) STEEM (Steem)	<b>Work Tokens</b>  Description: A token that provides the right to contribute to a system  Characteristics: <ul style="list-style-type: none"><li>• Owning a Work Token is the precondition for contributing to the system</li><li>• Contributions are either incentivized with a rewards system or holders get utility from the system/decentralization organization</li></ul> Examples: REP (Reputation, Augur), MMR (Market, Maker DAO)	<b>Security Tokens</b>  Description: A token that behaves like a security  Characteristics: <ul style="list-style-type: none"><li>• Shows classic security-like features, e.g. listing on exchanges regarding the issuing entity, dividends, or profit shares</li><li>• Holders are regarded as owners</li><li>• Utility or insufficient utility</li></ul> Examples: SPICE (SPICE VC), Bithala (Bla)
<b>(d)App Tokens</b>  Description: A token that is implemented on the application-level on top of a blockchain (and potentially protocol)  Characteristics: <ul style="list-style-type: none"><li>• Integrated with the application to provide specific incentives for nodes and/or users</li><li>• Based on an underlying blockchain to which it is not integral (e.g. ERC20 Token on Ethereum)</li></ul> Examples: WMI (Widnows, Gnosis), SAFE (SafeScore, SAFE Network)	<b>Investment Tokens</b>  Description: A token with share-like properties  Characteristics: <ul style="list-style-type: none"><li>• The issuer promises token owners a share in the success of the issuing entity (e.g. dividends, profit-shares)</li><li>• May or may not come with voting rights</li><li>• Mostly no physical legal basis</li></ul> Examples: Newfund (Gnosis) DEX (Digin Gold, DiginDAO)	<b>Share-like Tokens</b>  Description: A token with share-like properties  Characteristics: <ul style="list-style-type: none"><li>• The issuer promises token owners a share in the success of the issuing entity (e.g. dividends, profit-shares)</li><li>• May or may not come with voting rights</li><li>• Mostly no physical legal basis</li></ul> Examples: DGD (SigndDAO), LXR (Lykke) (likely to be classified as a security token)	<b>Hybrid Tokens</b>  Description: A token featuring traits of both usage and work tokens  Characteristics: <ul style="list-style-type: none"><li>• Grants access to system functionalities</li><li>• Allows owners to contribute to the system</li></ul> Examples: ETH (Ether, Ethereum, after Casper), DASH (Dash)	<b>Cryptocurrencies</b>  Description: A token that is a pure cryptocurrency  Characteristics: <ul style="list-style-type: none"><li>• Acts as a store of value and medium of exchange</li><li>• Regulated by a central authority against which owners have claims in Germany (according to BaFin)</li><li>• Considered as legal, functional currency</li><li>• not regulated by a currency laws</li></ul> Examples: BTC (Bitcoin), ZEC (Zcash), LTC (Litecoin)

\*details dependent on respective jurisdiction

# Crypto as seen by the Regulator (and the taxman)

- ① Cryptocurrency
- ② Utility Token
- ③ Security Token

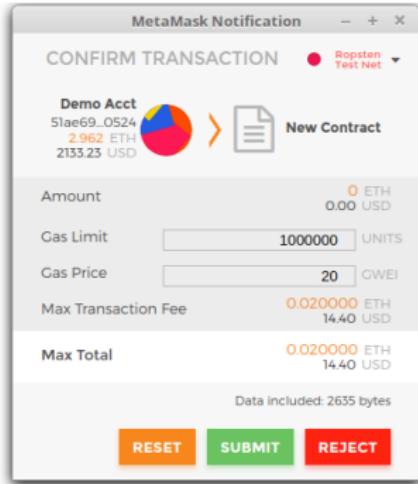


# Create your own (ERC-20) token

## Create Token

Create Token Contract with the following parameters.

100
Marc's Coin #2
8
MLD
<input type="button" value="Create Token"/>



- ① Use the Token Factory Dapp at <https://tokenfactory.surge.sh/#/factory>
- ② MetaMask will pop up (see picture above)
- ③ Submit the transaction (on the Ropsten Testnet)
- ④ Check your transaction on <https://ropsten.etherscan.io>

# Check your Smart Contract

A screenshot of a blockchain transaction status card. At the top, there are two tabs: "SENT" (highlighted in grey) and "TOKENS". Below the tabs, on the left, is a document icon with the number "2". In the center, the date and time are displayed as "December 29 2017 04:49". Below the date is the text "Contract Published". On the right, there is an orange copy icon with the number "0" and the text "ETH".

- ① Select the “Sent” tab
- ② Check the orange Copy icon (Tx Hash)
- ③ Click on “Contract Published”
- ④ That should bring you to Etherscan (see next page)





# Watch your Token



- ① Click on the “Add Token” button
- ② Wait for the next window (picture on the right)
- ③ Copy your contract address (from Etherscan)
- ④ Go back to your Token Factory tab, which should show an UI to interact with your contract or go to the URL:  
<https://tokenfactory.surge.sh/#/token/0x...> (replace 0x... by your contract address)
- ⑤ Move coins around
- ⑥ In MetaMask, click on your token to check the tx on Etherscan



Want to sell to token (ICO)?  
Consult your lawyer now...



# Table of Contents

- 1 What is Blockchain?
- 2 Hands-on Introduction to Crypto
- 3 ICOs, STOs, and other Token launches
- 4 Introduction to Smart Contracts



# Remember: Ethereum

Ethereum is a **decentralized platform that runs smart contracts**: applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third party interference.

— <https://ethereum.org>



# Coding your first ERC-20 Smart Contract

The screenshot shows the Remix IDE interface. On the left, there's a file tree with a single item: 'browser/ballot.sol'. The main area displays the Solidity code for the 'Ballot' contract. The code defines a struct 'Voter' with fields for weight, voted status, vote, and delegate. It also defines a struct 'Proposal' with a vote count. The contract has a mapping of addresses to voters and a mapping of proposals to their addresses. A constructor initializes the proposals mapping. A function 'Ballot' takes a uint8 parameter representing the number of proposals and creates a new ballot with that many proposals. The right side of the interface has tabs for 'Compile', 'Run', 'Settings', 'Debugger', 'Analysis', and 'Support'. Below the tabs, there are buttons for 'Start to compile' (with 'Auto compile' checked), 'Details', and 'Publish on Swarm'. A message box indicates 'Static Analysis raised 2 warning(s) that require fixing'. A green bar at the bottom says 'Ballot'.

```
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 {
```

- ① Open the Remix IDE at <https://remix.ethereum.org>
- ② Close the ballot file
- ③ Create a new file named TokenRecipient.sol
- ④ Copy the code from <https://ethereum.org/token> (second white box, under “The Code”, starting with “pragma”)
- ⑤ Switch to the “Run” tab (top-right bar, after Compile)

Reference:

ERC-20 Token Standard



# Compiling Successfully

The screenshot shows the Truffle UI interface. On the left, there's a file browser with a file named 'TokenRecipient.sol' selected. The main area displays the Solidity code for the contract. On the right, there's a toolbar with 'Compile', 'Run', 'Settings', 'Debugger', 'Analysis', and 'Support' buttons. A modal window titled 'TokenERC20' is open, showing the contract details. It includes a 'Start to compile' button, an 'Auto compile' checkbox (unchecked), and tabs for 'Details' and 'Publish on Swarm'. Below this, a message says 'Static Analysis raised 4 warning(s) that require attention'. Two green boxes are visible: one for 'TokenERC20' and another for 'tokenRecipient'.

```
1 pragma solidity ^0.4.16;
2
3 interface tokenRecipient { function receiveApproval(address _from, uint256 _value, string _tokenName, string _tokenSymbol);
4
5+ contract TokenERC20 {
6+     // Public variables of the token
7+     string public name;
8+     string public symbol;
9+     uint8 public decimals = 18;
10+    // 18 decimals is the strongly suggested default, avoid changing it
11+    uint256 public totalSupply;
12+
13+    // This creates an array with all balances
14+    mapping (address => uint256) public balanceOf;
15+    mapping (address => mapping (address => uint256)) public allowance;
16+
17+    // This generates a public event on the blockchain that will notify clients
18+    event Transfer(address indexed from, address indexed to, uint256 value);
19+
20+    // This notifies clients about the amount burnt
21+    event Burn(address indexed from, uint256 value);
22+
23+}
```

- ① Two green boxes should show on the right
- ② TokenERC20 is the name of the contract (class)
- ③ tokenRecipient is the name of the interface
- ④ Switch to the “Run” tab (top right)



# Submitting the Smart Contract

The screenshot shows the Truffle UI interface. On the left is a code editor window titled "browser/TokenRecipient.sol" containing Solidity code for a TokenERC20 contract. The code defines a public variable `totalSupply` and a mapping of balances for each address. It includes event definitions for approvals, transfers, and burns. A constructor function initializes the supply. On the right is a control panel with tabs for "Compile", "Run", "Settings", "Debugger", "Analysis", and "Support". The "Run" tab is selected. Under "Environment", it shows "Injected Web3" and an account address "0x51a...e0524". The "Gas limit" is set to 3000000 and the "Value" is 0 wei. A dropdown menu shows "TokenERC20". Below it, there's a "Create" button and a "Load contract from Address" field with "At Address". A transaction history section shows "1 pending transactions" and "0 contract instances".

```
1 pragma solidity ^0.4.16;
2
3 interface TokenRecipient {
4     function receiveApproval(address _from, uint256 _value, c
5
6     contract TokenERC20 {
7         // Public variables of the token
8         string public name;
9         string public symbol;
10        uint256 public decimals = 18;
11        // 18 decimals is the strongly suggested default, avoid changing it
12        uint256 public totalSupply;
13
14        // This creates an array with all balances
15        mapping (address => uint256) public balanceOf;
16        mapping (address => mapping (address => uint256)) public allowance;
17
18        // This generates a public event on the blockchain that will notify clients
19        event Transfer(address indexed From, address indexed To, uint256 value);
20
21        // This notifies clients about the amount burnt
22        event Burn(address indexed from, uint256 value);
23
24        /**
25         * Constructor function
26         * Initializes contract with initial supply tokens to the creator of the contract
27         */
28        function TokenERC20(
29            uint256 initialSupply,
30            string tokenName,
31            string tokenSymbol
32        ) public {
33            totalSupply = initialSupply * 10 ** uint256(decimals); // Update total supply
34            balances[From] = initialSupply; // Give the creator all
35        }
36    }
37}
```

- ① Under the dropdown showing “TokenERC20”, add a number (total amount of tokens to issue) and two strings (the latter is the token symbol)
- ② Add enough gas (top right, try 30)
- ③ Click Create and check whether MetaMask needs confirmation



# Interacting with the contract

- ① A new interface will pop up on the bottom right corner of the IDE

The screenshot shows a blockchain IDE interface. At the top, there is a header bar with the text "0 pending transactions" and three icons: a square, a document, and a play button. Below this is a list of functions for a contract named "TokenERC20 at 0xaca5...ee675 (blockchain)". The functions listed are:

- totalSupply
- symbol
- name
- decimals
- allowance address , address
- balanceOf address
- transferFrom address \_from, address \_to,
- burnFrom address \_from, uint256 \_val
- approve address \_spender, uint256 \_
- approveAndCall address \_spender, uint256 \_
- transfer address \_to, uint256 \_value
- burn uint256 \_value



# Tools for Developers



# Truffle Framework

<http://truffleframework.com>

## YOUR ETHEREUM SWISS ARMY KNIFE

Truffle is the most popular development framework for Ethereum with a mission to make your life a whole lot easier.

 Star 4,734

 Fork 594

 gitter [join chat](#)

### INSTALL VIA NPM

```
$ npm install -g truffle
```

*Requires NodeJS 5.0+. Works on Linux, macOS, or Windows.*

[DOCUMENTATION](#)

[TUTORIALS](#)

Don't know where to start? Get yourself a [Truffle Box!](#)



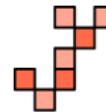
# Infura

<http://infura.io>



## BLOCKCHAIN BASED

We eliminate the need to install, configure, and maintain costly Ethereum infrastructure.



## RELIABLE AND SCALABLE

Our Ferryman™ middleware improves reliability and helps us scale quickly to meet your demand.



## DISTRIBUTED STORAGE

Access IPFS seamlessly without the hassle of managing the infrastructure.

## POWERFUL AND SECURE

5B+

Requests Per Day

1.6PB

Data Transferred Per Month

9000+

Developers and DApps Served



# Mythril

<https://github.com/ConsenSys/mythril>

Mythril is a security analysis tool for smart contracts.

It comes as a Python package that requires a solidity compiler and a C++ compiler.

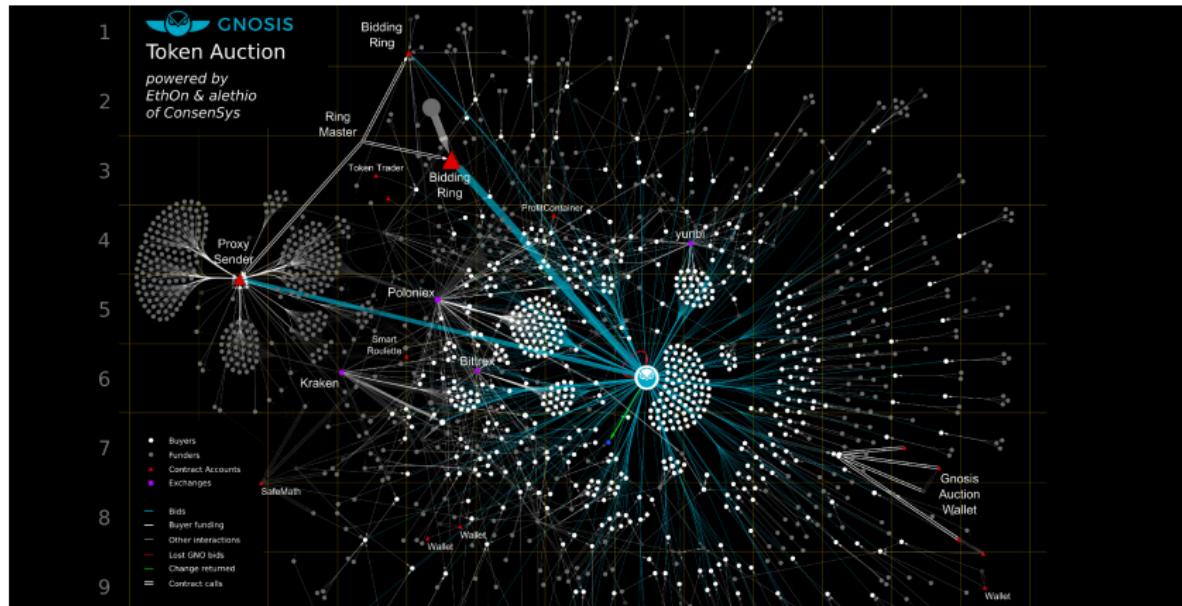
```
$ sudo apt install libssl-dev  
$ sudo apt install gcc g++  
$ sudo add-apt-repository ppa:ethereum/ethereum  
$ sudo apt install solc  
$ sudo pip3 install mythril  
$ myth -x contracts/higherbidder.sol
```

See also <https://hackernoon.com/introducing-mythril-a-framework-for-bug-hunting-on-the-ethereum-blockchain-9dc5588>



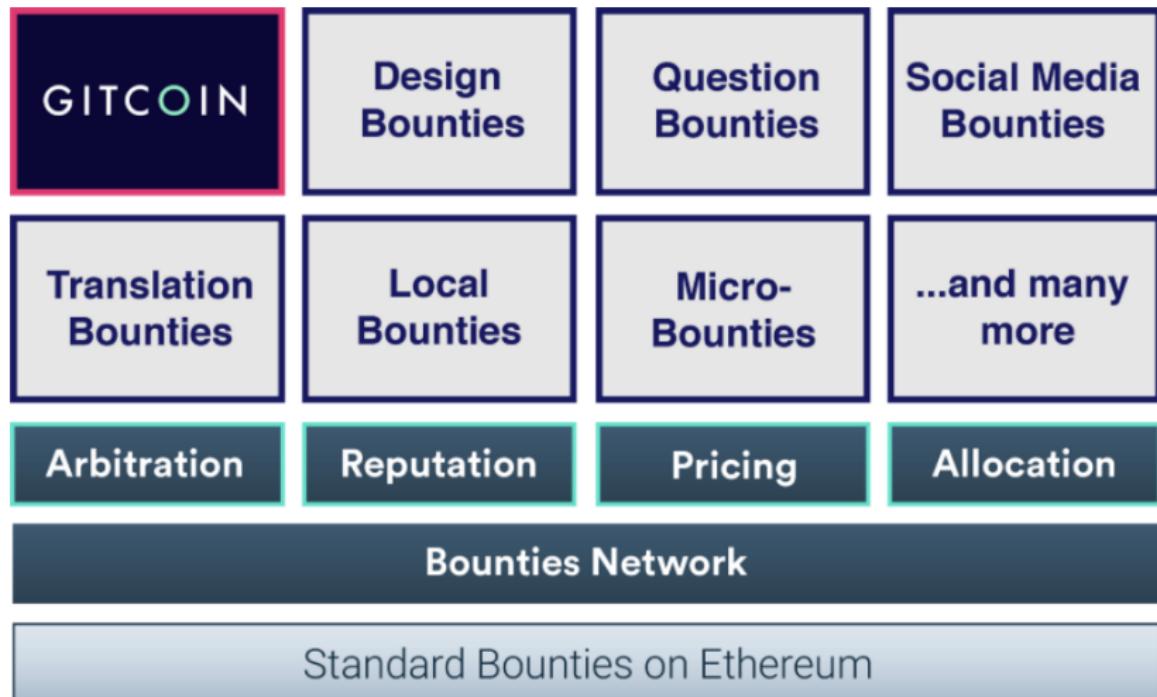
# Big Data and Analytics on Ethereum

<https://aleth.io>



# Getting paid for your work: The Bounties Network

<https://bounties.network>

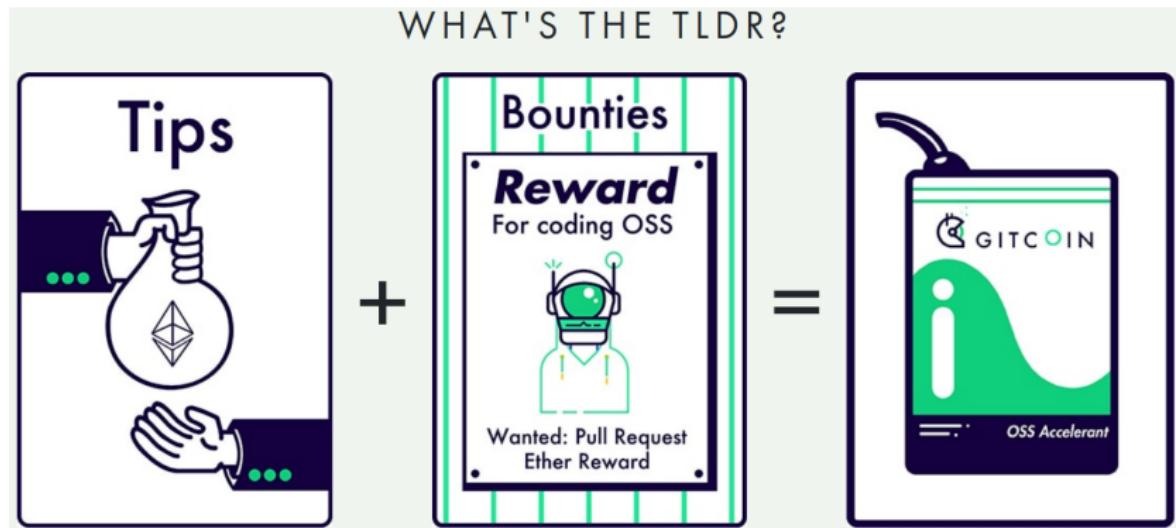


Gitcoin (depth-first) and Bounties Network (breadth-first) have integrated!



# Gitcoin

<https://gitcoin.co>



# Thank you!

Email: [marc@metameshgroup.com](mailto:marc@metameshgroup.com)

Twitter: [@marclijour](https://twitter.com/marclijour)



# References

