Why Cryptocurrency?

@babudahal

Agenda: To explain why, so you can learn what and how.

Chapter 1

Cryptocurrency

A Beginners Guide on Technology and Investment

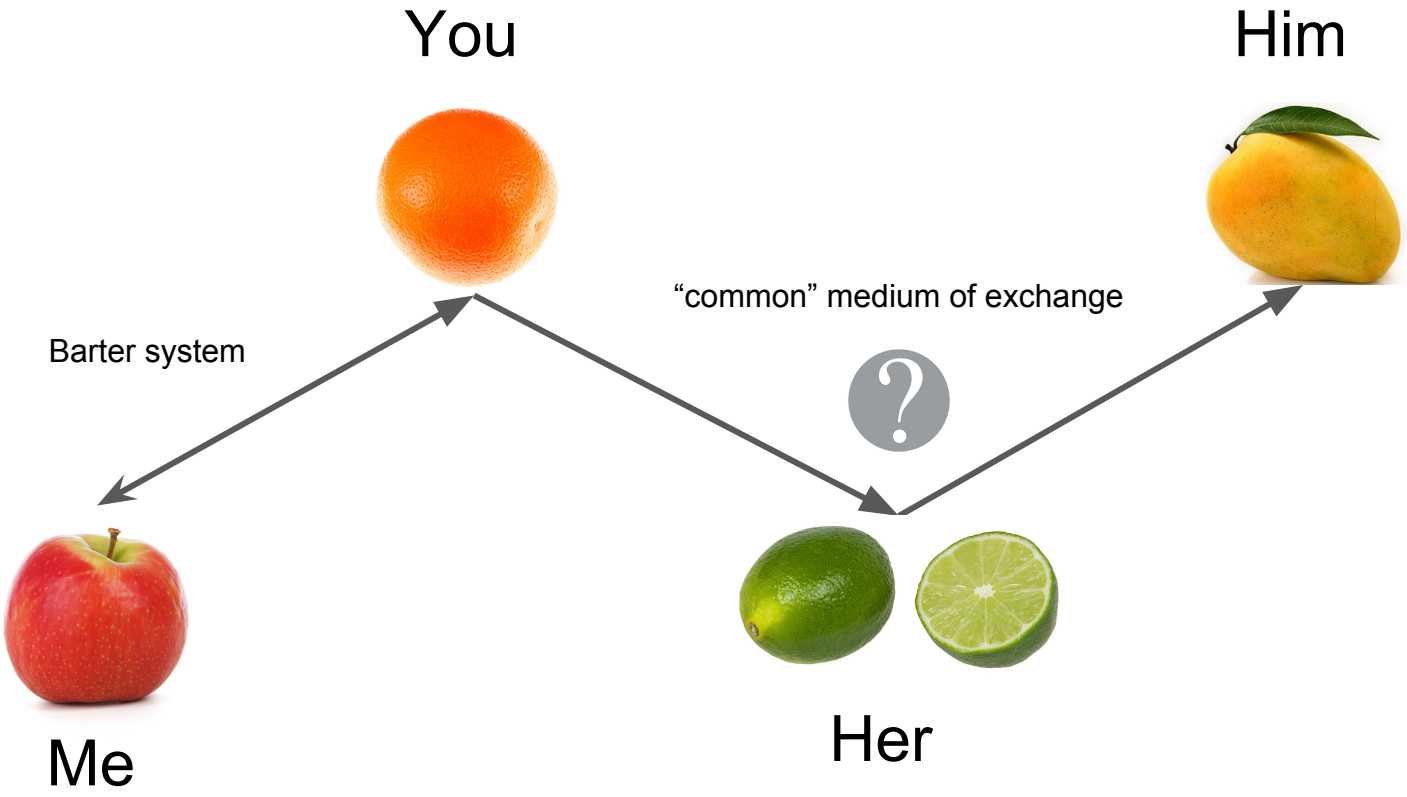
@babudahal
Cryptocurrency Enthusiast
26 Dec 2017

Agenda: To provide basic information on cryptocurrencies to help make informed decisions.

(Not financial advice)

You have been warned!

Why Money?



Types of Money?

Cattle

Salt

Shells

Government moves into money

Gold

Paper Money / Cash

Government moves out of money

Bitcoin

Money vs Currency

- **Money is ‘store of value’.**
- **Money is not tied to any government.**
- **Eg. Gold**

- **Currency is representation of money.**
- **Currency is tied to a government.**
- **Commodity Currency (CAD, AUD)**
- **Fiat Currency (USD, Euro)**

Bitcoin Is Global Currency.

Why Bitcoin?

Just like Internet provides access to information to anyone in the world,
Bitcoin provides access to global financial system to anyone in the world.

- Whitepaper: Bitcoin: A Peer-to-Peer Electronic Cash System - Satoshi Nakamoto
- Who is Satoshi Nakamoto?
  - Still unknown and doesn't matter. Just like it doesn't matter who invented internet.

Other Benefits?

- Privacy
- Security
- Speed
- Transaction Fees
- No Third-Party Interruptions
- Global Accessibility
- Push vs Pull
  - Merchants can pull money from credit cards anytime. And if hackers have credit card number, they can pull too.
  - In Bitcoin, money goes out only when you send (push) the transaction, no one can pull it.

Why Ethereum?

Ethereum adds programming layer on top of blockchain for building smart contracts and decentralized applications.

- Blockchain
  - Global Public Ledger (list of all transactions, like database but publicly accessible)
- Smart Contracts
  - Contract is legally binding agreement
  - Smart Contract is code (for that agreement) stored in blockchain that can self-execute
  - Eg. In 8 June 2018 send ‘1 ETH’ to <Son’s ETH Address>
- Decentralized Applications (DAPPS)
  - If Ethereum is Internet then DAPPS are like websites, but decentralised. (like decentralized Google, Facebook, Youtube)

- Whitepaper: A Next-Generation Smart Contract and Decentralized Application Platform - Vitalik Buterin

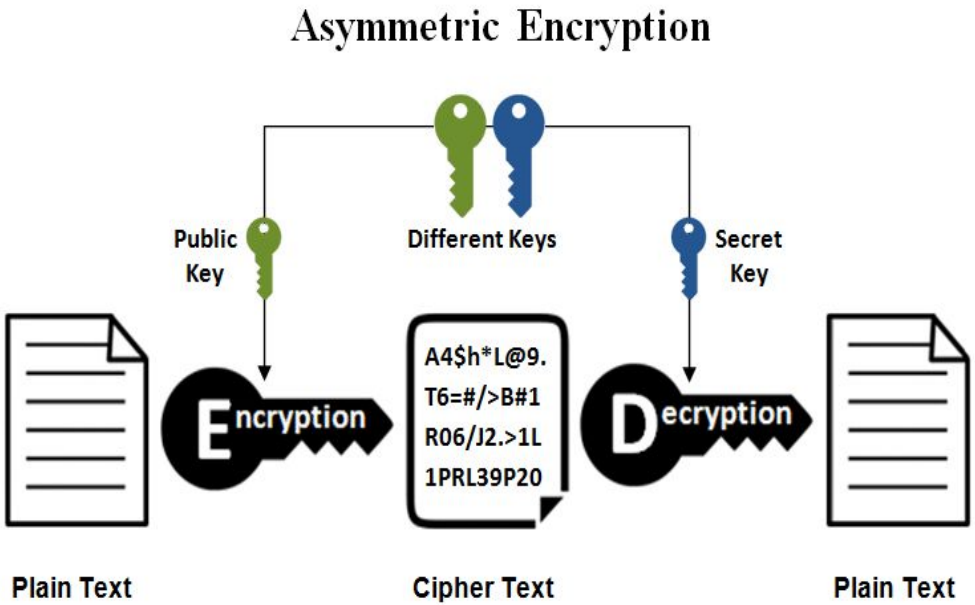
Why Token?

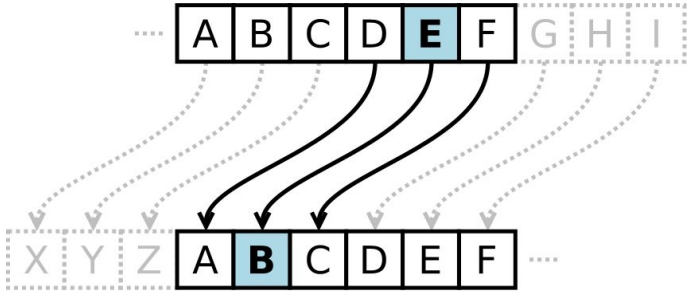
- If Ether (currency on Ethereum) is gold, then Tokens are gold pegged currencies like CAD, AUD etc.
- DAPPS need tokens same like different countries need their own currencies to meet with own economic requirements.

Types of Tokens

- Utility Tokens
  - Tokens required to use the service of DAPPS. Eg. Golem Network
- Work Tokens
  - Tokens that act as shares and gives you voting rights. Eg. DAO (Decentralized Autonomous Organization) Tokens

Cryptocurrency = Cryptography + Currency





Alphabet shift ciphers are believed to have been used by Julius Caesar over 2,000 years ago. (Wikipedia)

CAT	ZXQ	CAT	CAT = ZXQ

Before Understanding Blockchain

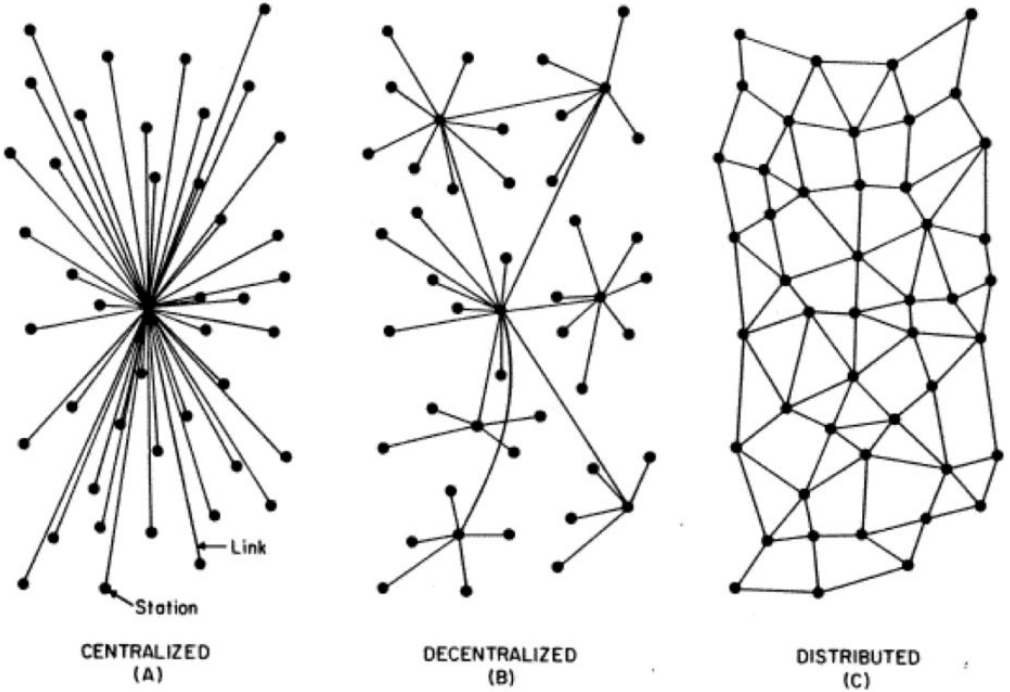
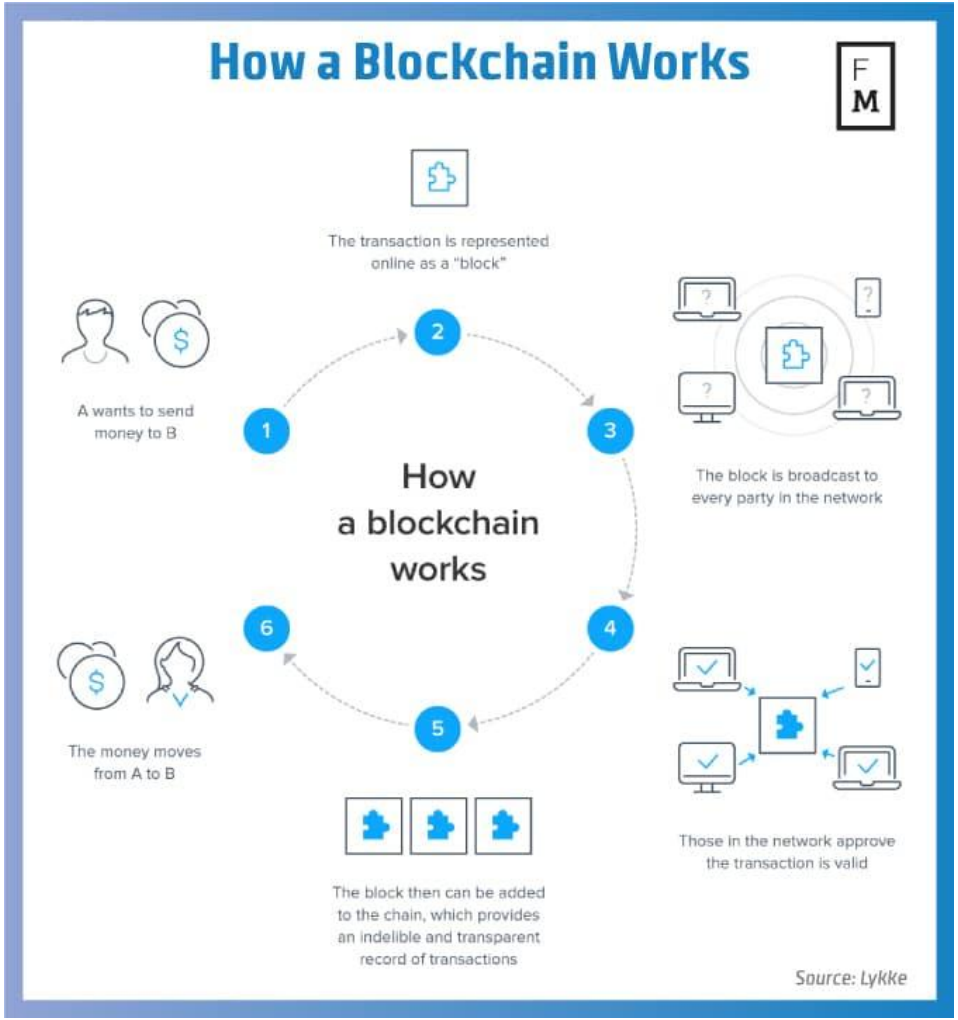
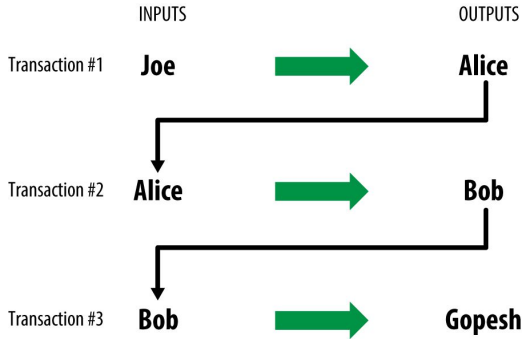


FIG. 1 — Centralized, Decentralized and Distributed Networks

(Types of networks. Paul Baran, 1964)


How a Blockchain Works
F M
The transaction is represented online as a "block"
A wants to send money to B
The block is broadcast to every party in the network
How a blockchain works
The money moves from A to B
Those in the network approve the transaction is valid
The block then can be added to the chain, which provides an indelible and transparent record of transactions
Source: Lykke

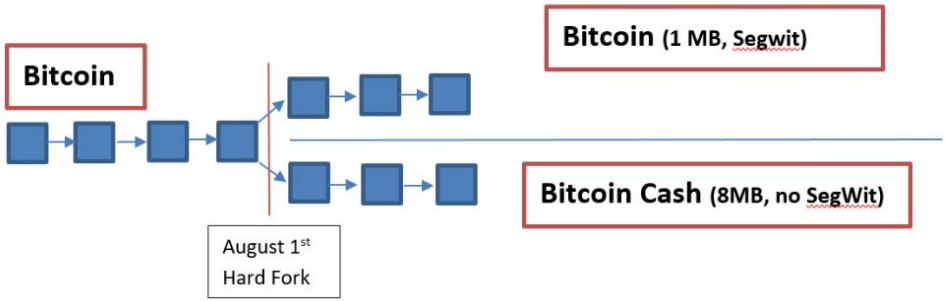

INPUTS
OUTPUTS
Transaction #1
Joe
Alice
Transaction #2
Alice
Bob
Transaction #3
Bob
Gopesh


block n
block 1
block 0

Why Fork?

Because of network changes, or due to community disagreement.

- Hard Fork
  - No backward compatibility
- Soft Fork
  - Backward compatibility



Replacing old with new

	Bank	Bitcoin
Identity	Account Username Password	Wallet Public Address Private Key
Network	Centralized	Decentralized (Nodes)
Transaction Stored	Database	Blockchain (Public Ledger)
Transaction Settlement	Clearing houses	Miners (Verify Transactions)

Blockchain use cases list by industry

Financial
Trading
Deal origination
POs for new securities
Equities
Fixed income
Derivatives trading
Total Return Swaps (TRS)
2nd generation derivatives
The race to a zero middle office
Collateral management
Settlements
Payments
Transferring of value
Know your client (KYC)
Anti money laundering
Client and product reference data.
Crowd Funding
Peer-to-peer lending
Compliance reporting
Trade reporting & risk visualizations
Betting & prediction markets

Insurance
Claim filings
MBS/Property payments
Claims processing & admin
Fraud prediction
Telematics & ratings

Media
Digital rights mgmt
Game monetization
Art authentication
Purchase & usage monitoring
Ticket purchases
Fan tracking
Ad click fraud reduction
Resell of authentic assets
Real time auction & ad placements

Computer Science
Micronization of work (pay for algorithms, tweets, ad clicks, etc.)
Expanse of marketplace
Disbursement of work
Direct to developer payments
API platform plays
Notarization & certification
P2P storage & compute sharing
DNS

Medical
Records sharing
Prescription sharing
Compliance
Personalized medicine
DNA sequencing

Asset Titles
Diamonds
Designer brands
Car leasing & sales
Home Mortgages & payments
Land title ownership
Digital asset records

Government
Voting
Vehicle registration
WIC, Vet, SS, benefits, distribution
Licensing & identification
Copyrights

Identity
Personal
Objects
Families of objects
Digital assets
Multifactor Auth
Refugee tracking
Education & badging
Purchase & review tracking
Employer & Employee reviews

IoT
Device to Device payments
Device directories
Operations (e.g. water flow)
Grid monitoring
Smart home & office management
Cross-company maintenance markets

Payments
Micropayments (apps, 402)
B2B international remittance
Tax filing & collection
Rethinking wallets & banks

Consumer
Digital rewards
Uber, AirBNB, Apple Pay
P2P selling, craigslist
Cross company, brand, loyalty tracking

Supply Chain
Dynamic ag commodities pricing
Real time auction for supply delivery
Pharmaceutical tracking & purity
Agricultural food authentication
Shipping & logistics management

How Cryptocurrencies are being used, today?

- Store of value
  - People in countries like Venezuela are using cryptocurrency to protect themselves against inflation cause ‘value’ of their own currency is going down significantly.
  - People of other countries are using it as digital gold.

How Cryptocurrencies are being used, today? cont..

Crowdfunding	People who created protocols like TCP/IP, SMTP, HTTP didn't gain financial benefit. But, now they have more incentive to create better protocols that will ultimately benefit society.
Remittance	Yearly remittance from international financial transfers out of the US is up to $50 billion. BitPesa (Africa), Bitspark (Philippines, Indonesia and Vietnam)
Identity	Microsoft and Accenture created Global ID system to track refugees. Same can be used to prevent human trafficking.
Supply Chain Tracking	WaBi is tracking infant baby milk to ensure fake goods never get to the hands of customers. Same can be used for fake land ownership.

How to buy and store?

- Fiat currency to Crypto
  - https://www.coinbase.com/
  - https://gemini.com/
- Wallet
  - Web Wallets
    - https://www.myetherwallet.com
    - https://metamask.io/
  - Hardware Wallets
    - Ledger Nano S
    - Trezor
- Search Transaction
  - https://blockchain.info/
  - https://etherscan.io/

TIPS:

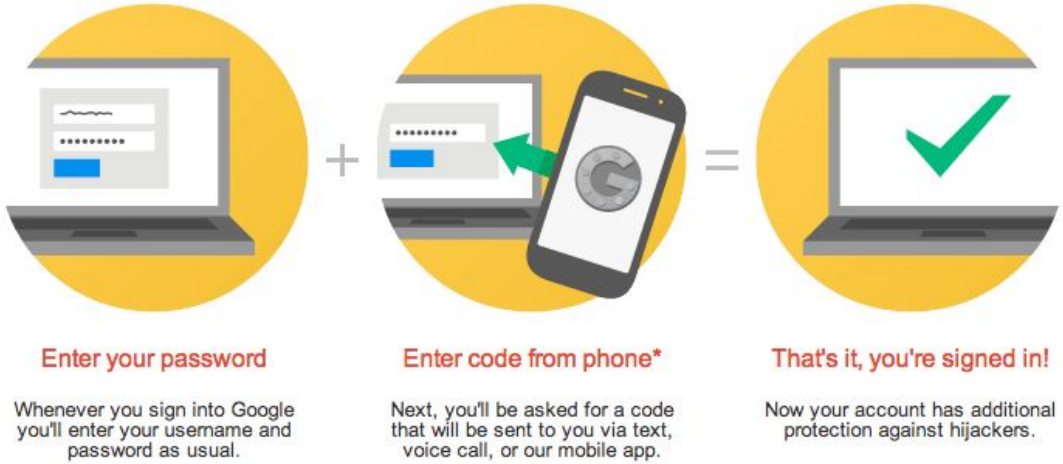
- Don't transfer in bitcoin cause currently transactions are slow and expensive (high transaction fees). Instead, convert to Ethereum and send Ethereum between exchanges.
- Check address multiple times and make sure they are of right coin. Ethereum can only be transferred to Ethereum address and not Bitcoin.
- Ethereum tokens CAN be transferred to Ethereum address.
- To avoid using long addresses, Ethereum Name Services (ENS) can be used to send Ethers, where applicable.

How to Invest?

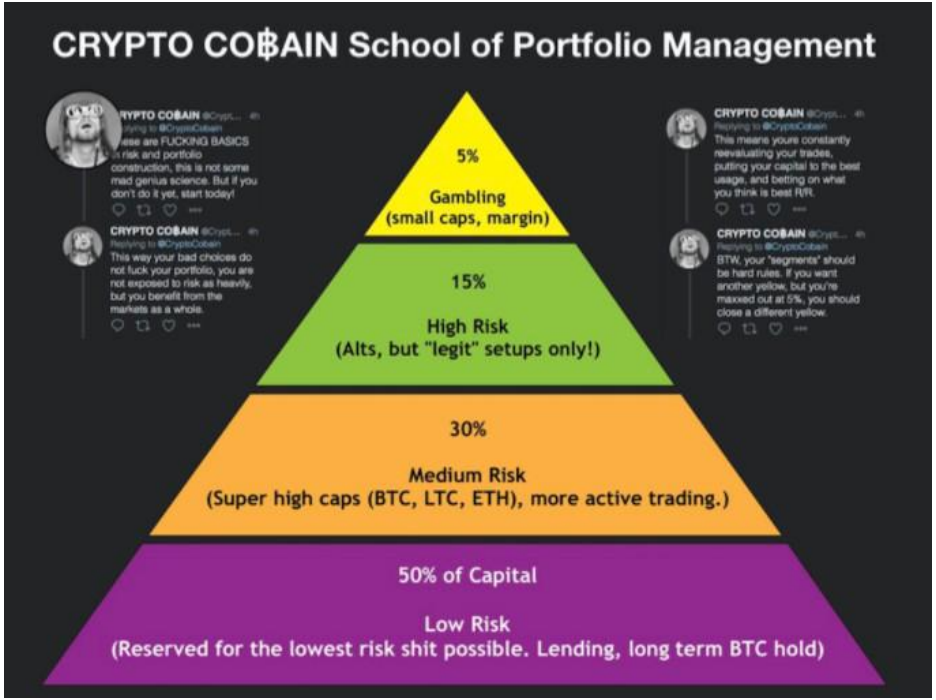
- List of Cryptocurrencies
  - https://coinmarketcap.com/
  - https://www.tokendata.io/
- ICOs (Initial Coin Offerings)
  - https://icodrops.com
- Research
  - Company **(avoid scams)**
    - https://www.crunchbase.com/
    - https://crushcrypto.com/
  - Team
    - Interviews, Podcasts (youtube.com)
    - Presentations (youtube.com)
    - Blogs (medium.com)
  - White Paper, Product and Market
  - Community, VCs, Advisors
  - Twitter.com, tweetdeck.com
- Exchanges
  - Centralized
    - https://bittrex.com/
    - https://www.binance.com/
  - Decentralized
    - https://radarrelay.com/
    - https://etherdelta.com/
- Portfolio Management
  - https://www.blockfolio.com/
- News and Entertainment
  - https://www.coindesk.com/
  - https://bitcointalk.org/
  - https://www.reddit.com/r/CryptoCurrency/
  - https://www.reddit.com/r/Bitcoin/
  - https://www.reddit.com/r/ethereum/
  - https://www.reddit.com/r/ethtrader/

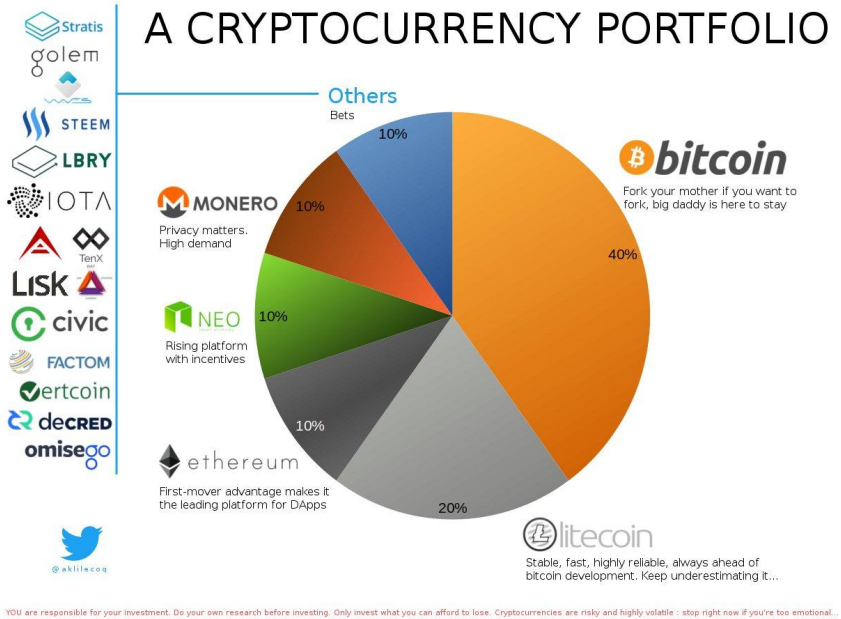
WARNING

- ALWAYS active 2FA (Two Factor Authentication) on ALL online wallets and exchanges. Else, your account could get hacked.
- Use Google Authenticator phone app for 2FA.
- NEVER use phone SMS as recovery option.



Portfolio Management?





Investing Tips

- Following rules is more important than having them.
- Always look for new technologies and invest in them even if you have to cut from current ones.
- Follow the wind.
- Don't trade in individual currency, manage a portfolio.
- Follow great people until you become one.
- You won't get rich investing in high cap companies.
- Opportunity is just getting started.
- Doesn't matter how early you invest, you will still think you should have invested earlier, unless you are Satoshi Nakamoto.
- Track each transaction in excel for tax purposes.
- Best strategy is to regularly invest some portion of paycheck.
- ~~Don't invest more than you can afford to lose.~~ Only invest what you are comfortable with.
- Never take loan to invest.
- Always look at Market Capital, not price per currency.

	Circulating Supply	Price	Market Cap
Bitcoin	16,763,025	$14,077.00	$235,973,102,925
Ripple	38,739,144,847	$1.12 (Current) $14,077.00 (If)	$43,330,120,903 545 Trillion

Trading Tips

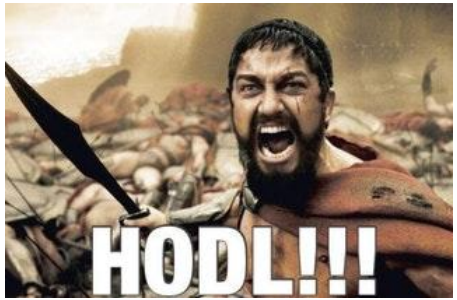
- Never go all-in, never go all-out.
- Don't buy at rock bottom, cause you don't know it's rock bottom.

Critics

- When in doubt try replacing word ‘Bitcoin’ with ‘Internet’ and that should make things clear.
  - Big corporations are powerful and will try to stop ~~Bitcoin~~ Internet?
  - What if governments try to stop ~~Bitcoin~~ Internet?
  - Why are there so many ~~cryptocurrencies~~ websites in ~~crypto~~ Internet?
  - ~~Bitcoin~~ Internet is a ponzi scheme.
- Is it bubble? Doesn't matter. Like the internet bubble of 2000 ultimately didn't matter. What matter is how can you make best of it.
- Will it succeed? Should it succeed? As long as Bitcoin provides value to society it will succeed.

Crypto Memes

- To The Moon
- FOMO (Fear of Missing Out)
- HODL
- Small fish, Sharks, Whales
- This is gentlemen
- It’s happening



References

- https://youtu.be/qtOIh93Hvuw - Muneeb Ali
- https://youtu.be/JP9-IAYngi4 (Highly Recommended)
- Mastering Bitcoin - Andreas M. Antonopoulos
- https://mycrypto.guide/
- Bitcoin, Ethereum and other whitepapers

Q & A


What are you trying to tell me, that I can trade my bitcoin for millions someday?
No Neo, I'm trying to tell you that when you're ready...
you won't have to.

Chapter 2

Ethereum Developer Bootcamp

Decentralized App Development

@babudahal
Blockchain Developer
21 Jan 2018

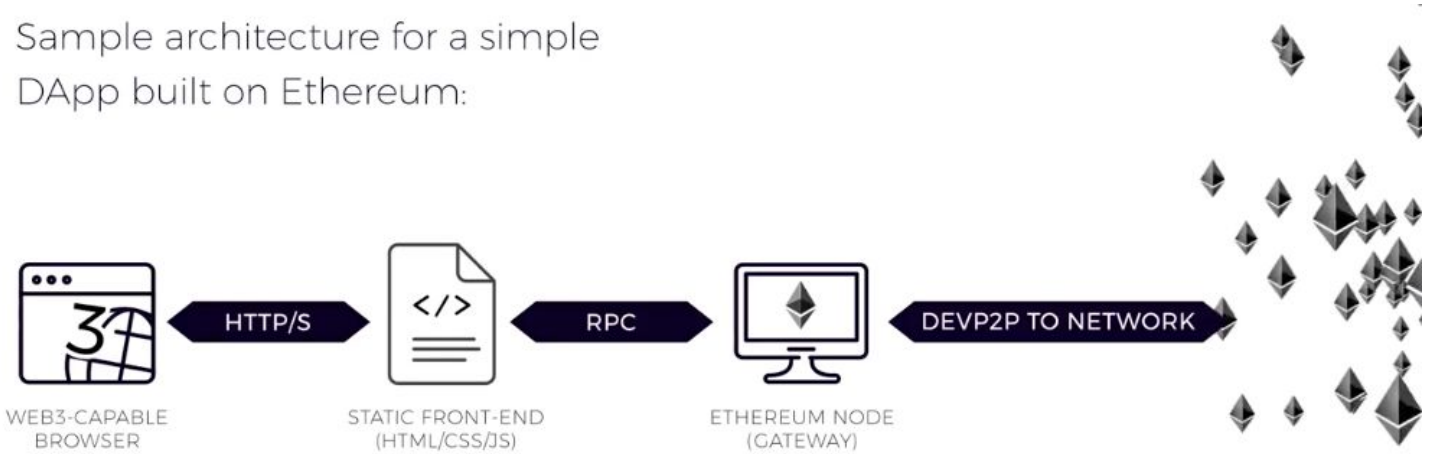
Agenda: To explain Blockchain technology in simplest form and help develop decentralized applications (dapps).

(Reference: Course from Consensys)

Why Decentralized Application?

Because it eliminates central authority and gives power back to users.

- Should data be managed by a central authority?
- Should data be dynamic and auditable?



Why Web3?

To make web decentralized.

- Web1: The Read-Only Web
- Web2: The Read-Write Web
- Web3: The ‘Unmediated’ Read-Write Web
  - Censorship resistance
  - Data owned by users
  - Transparent, open networks
  - Global interactivity
  - Self-sovereign identity
  - Push, not pull
    - Users pushes their information to third parties, not other way around.

Why Distributed Ledgers?

Because they are fault tolerant (system operates even when some of its component fails).

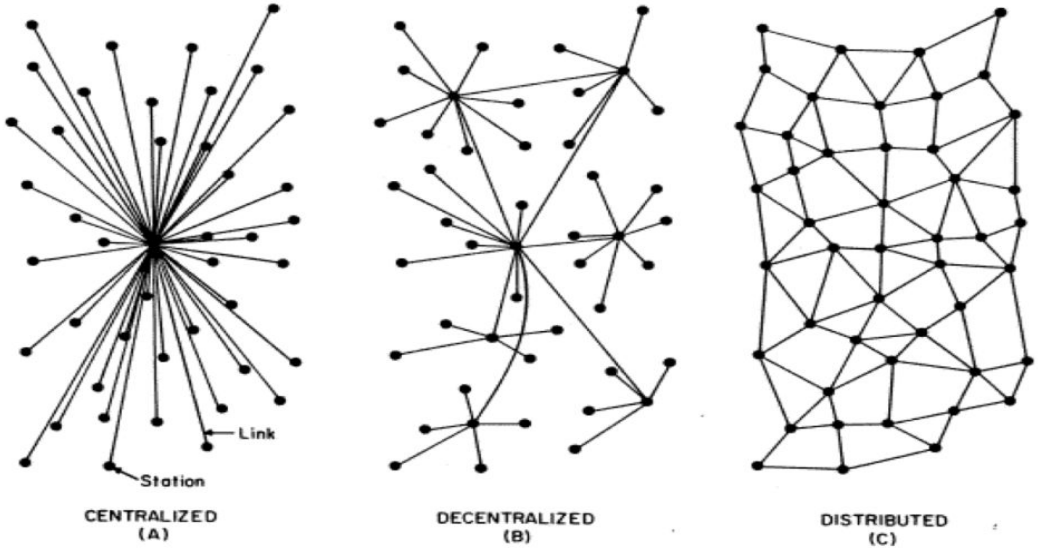


FIG. 1 — Centralized, Decentralized and Distributed Networks

(Types of networks. Paul Baran. 1964)

Why Consensus Mechanisms?

To make sure every node has a copy of same ledger; and to avoid invalid transactions.

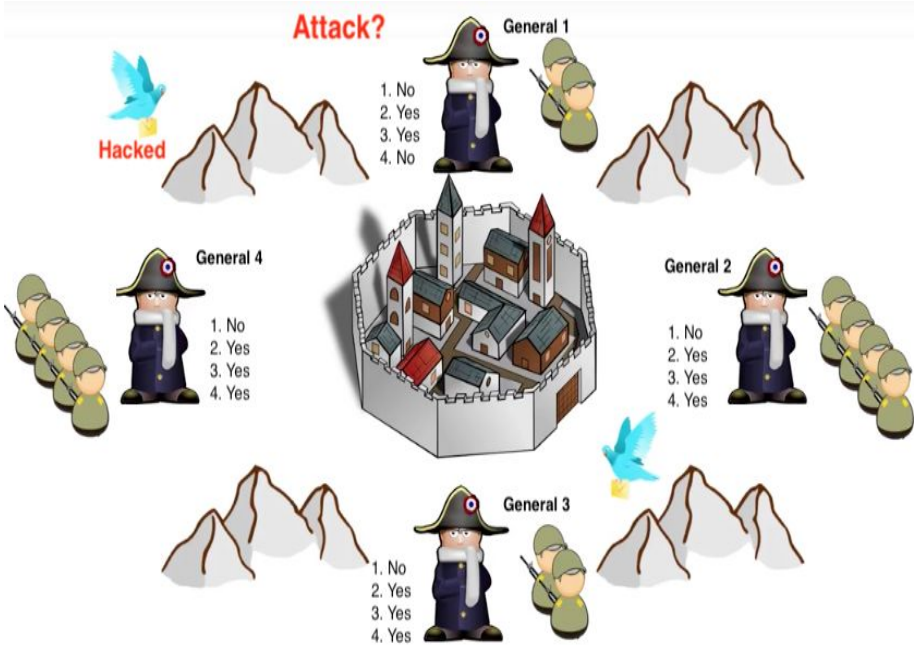
Types:

- Practical Byzantine Fault Tolerance (PBFT)
- Proof of Work
- Proof of Stake

Byzantine Generals Problem

Because Practical Byzantine Fault Tolerant algorithm can tolerate ~⅓ dishonest or absent participants and still reach consensus.

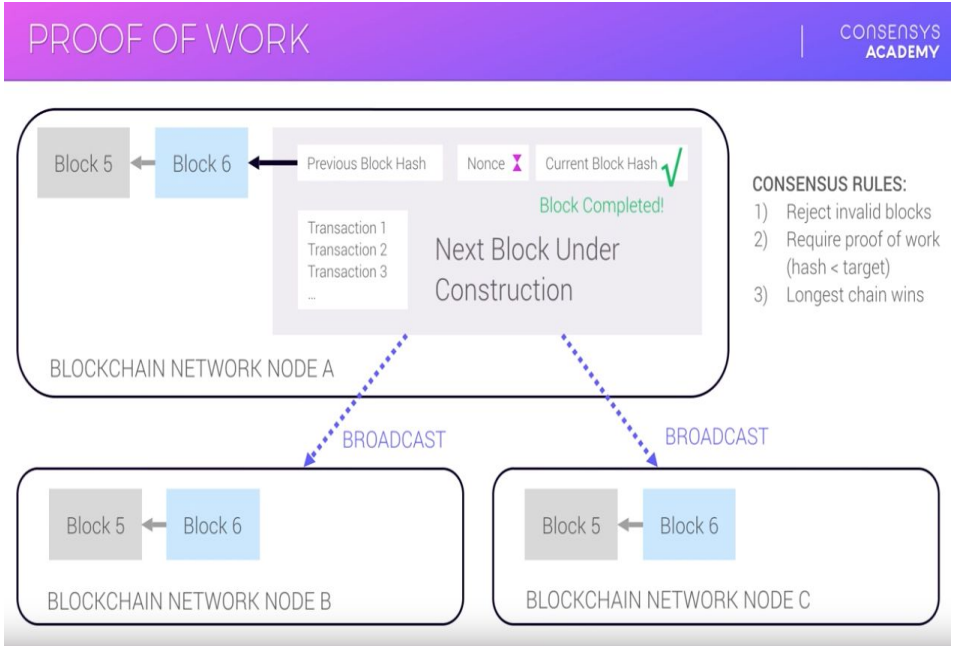
- Attack or Retreat?
- Attack by few general is worse that either full retreat or full attack.
- Byzantine Fault Tolerant is achieved if generals have majority agreement on their strategy.
- A system is called Byzantine Fault Tolerant even if 1 out of 4 nodes are faulty.



Why Proof of Work (PoW)?

Because in PoW consensus mechanism, participants will only accept valid block when block hash is less than a target number (difficulty).

- Mining secures the network with PoW.
- Miners repeatedly hash the contents of a block and check against the difficulty. If it is below the difficulty, they broadcast the valid block to network and get rewarded for it.
- Difficulty increases or decreases to maintain the block time. For, Ethereum, time to validate a block is ~15 secs.



Why Proof of Stake (PoS)?

Because PoS uses less energy (as opposed to PoW) and can support high volume of transactions (unlike PoW, which has low transaction volume due to hardware limitations).

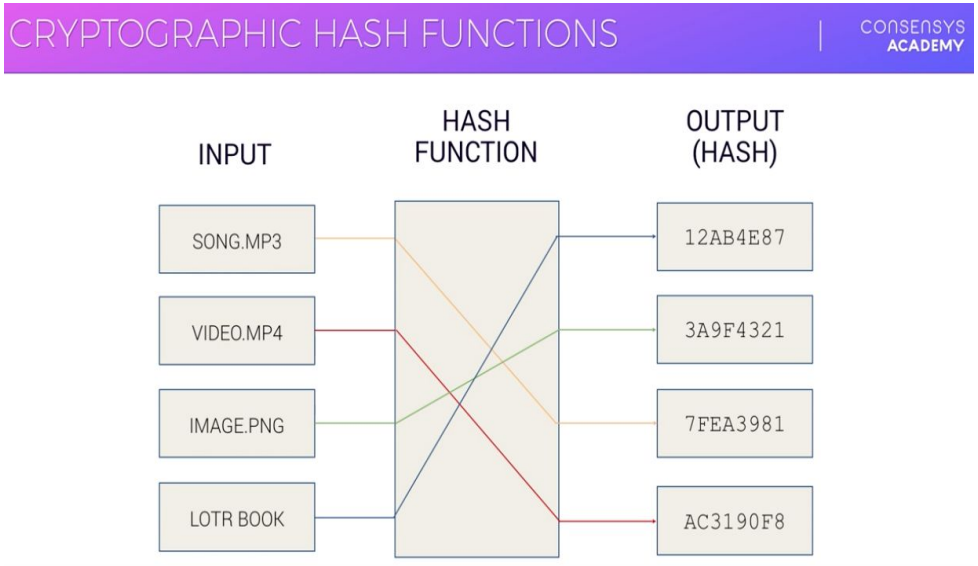
Types:

- Leased PoS
  - Token holders can lease balances to others
  - Rewards are split among leasers
- Delegated PoS
  - Token holders elect block validators
  - No sharing of block rewards

Why Cryptographic Hash Functions?

Because Hash function secures the data.

- Maps input data of any size to output data of fixed size.
- It is a one-way function.



Two eras of cryptography

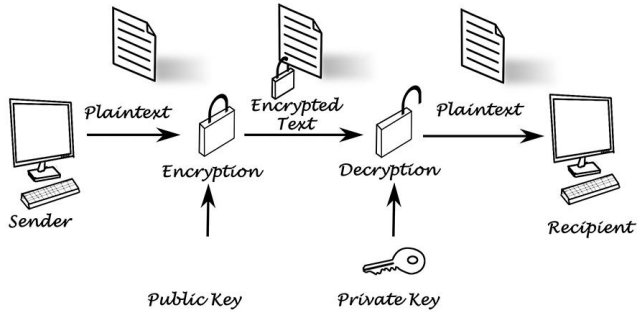
- Classical
  - Transporting secret codebooks around the world.
- Modern
  - Being able to have secure communication between two parties without worrying about someone listening in.
  - RSA (Rivest, Shamir, and Adelman)
  - Elliptic Curves

Why Public Key (Asymmetric) Cryptography?

Because it uses different keys (public & private) for encryption and decryption. Hence, it can be used for not just encrypting data, but also verifying them via digital signatures.

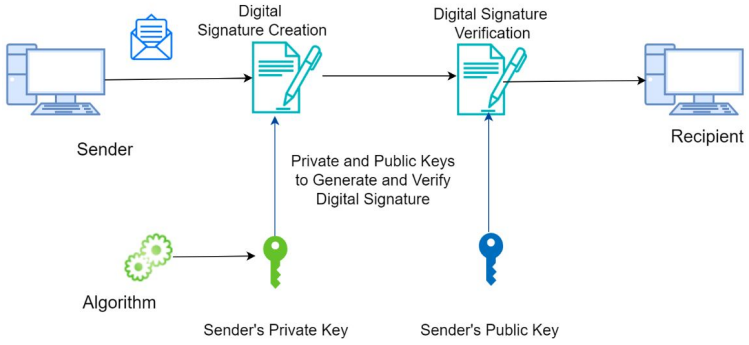
Encryption

- From private key, you can derive your public key.
- People can encrypt things to your public key, and only your private key can decrypt them.



Authentication (Digital Signatures)

- A signature is something that can only be created by your private key.
- Then public key can be used to verify that the signature indeed must have been created with your private key.



Elliptic Curves Discrete Logarithm Function (ECDLF)

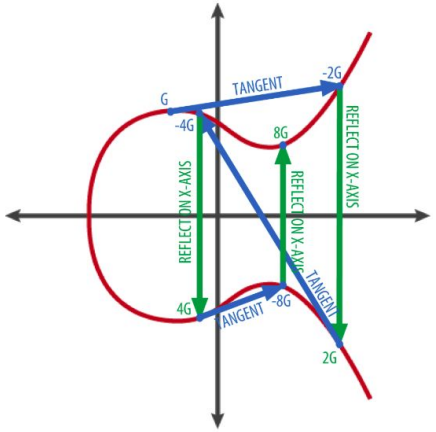
Formula: y^2 = x^3 + ax + b

Properties

- Any non-vertical line will intersect the curve in at most three places.
- It has horizontal symmetry.

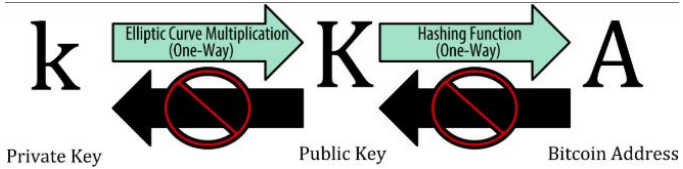
Rule

- Take tangent of the point.
- Take its reflection.
- Repeat.



*Formula: K = k * G*

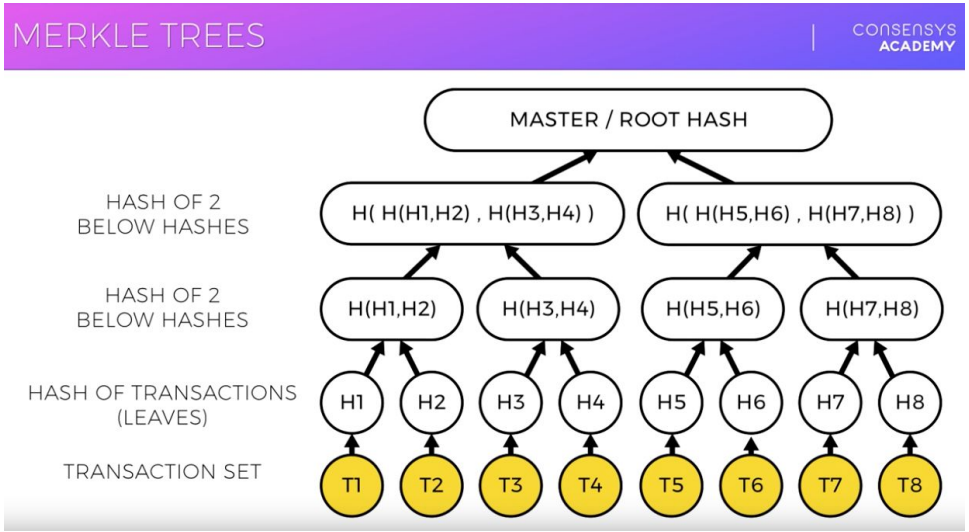
Cryptography	Billiard Analogy
G - Generator Point (constant)	Starting position of the ball. (center of the table)
k - Private Key	No. of times you hit the ball.
K - Public Key	Final position of the ball.



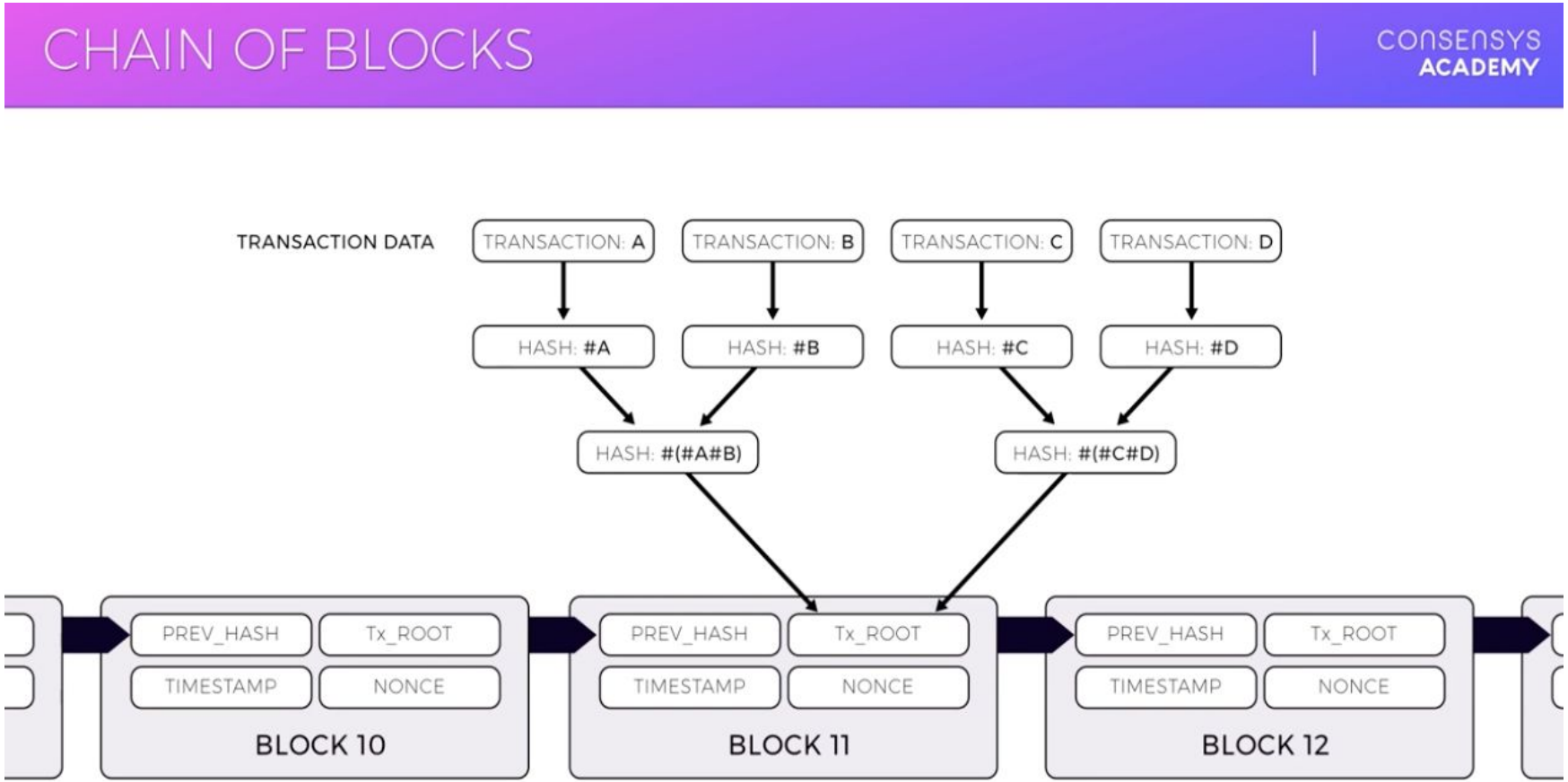
Why Merkle Trees?

Because Merkel Trees allows efficient and secure verification of contents of large data structures.

- At the bottom is transaction set.
- Hash of each transactions corresponds to leaves
- Hash of leaves corresponds to nodes, and the process continues.
- At the top is the root hash.

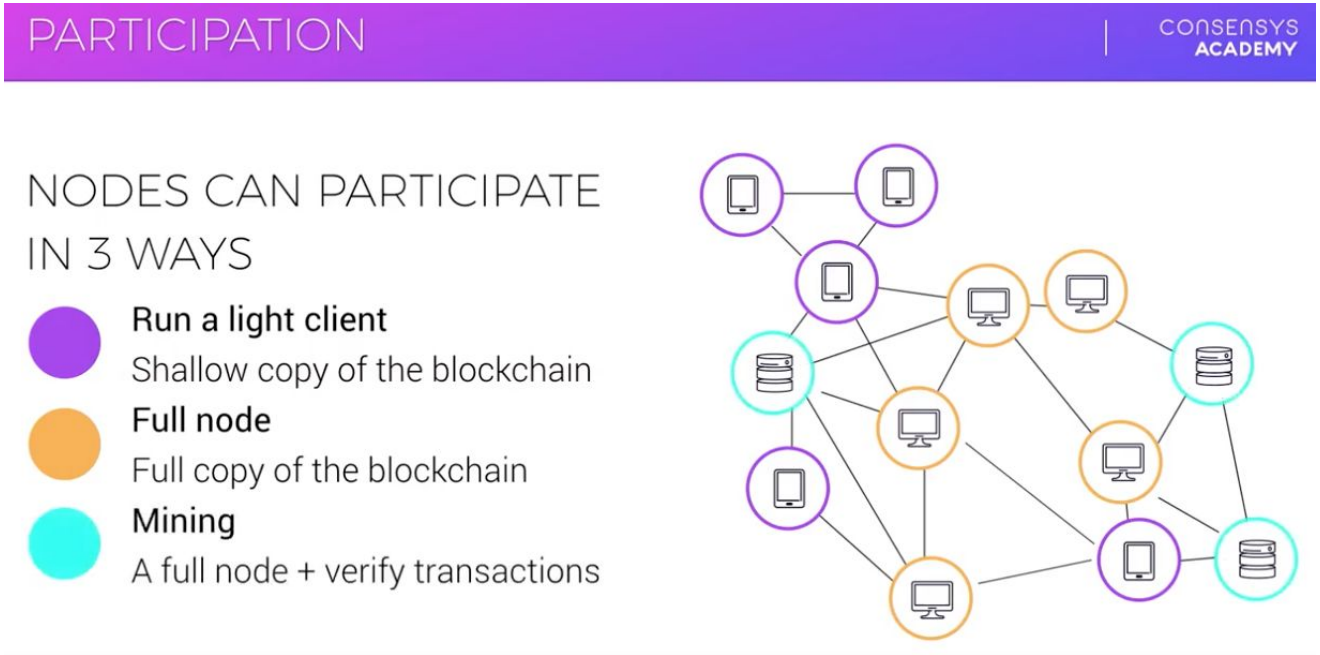


Blockchain Structure



Why Nodes?

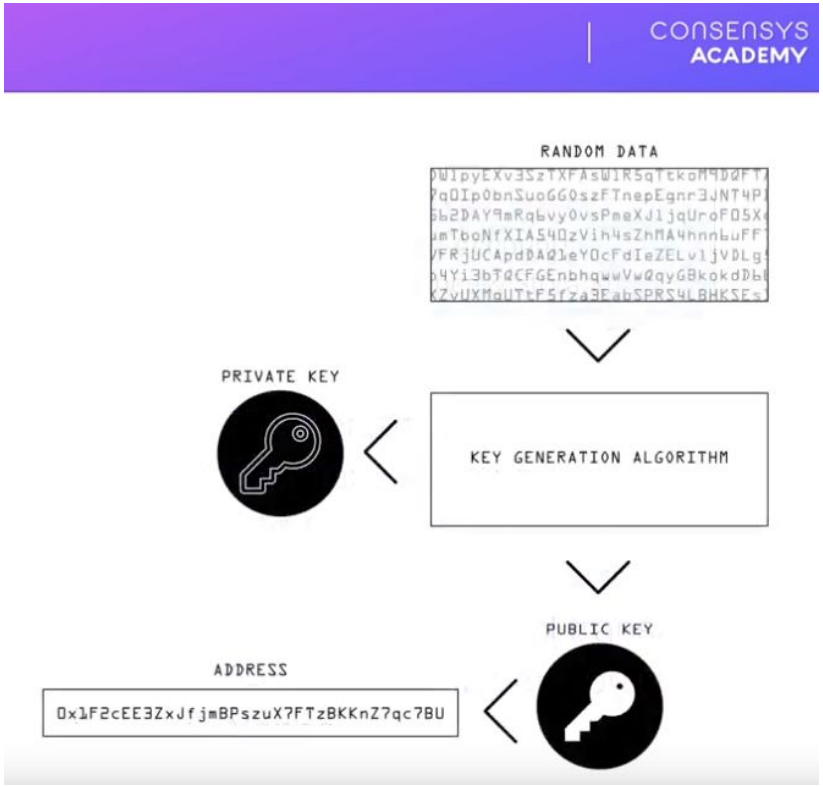
Because nodes are gateway to the network to read/write data to the blockchain.



Account

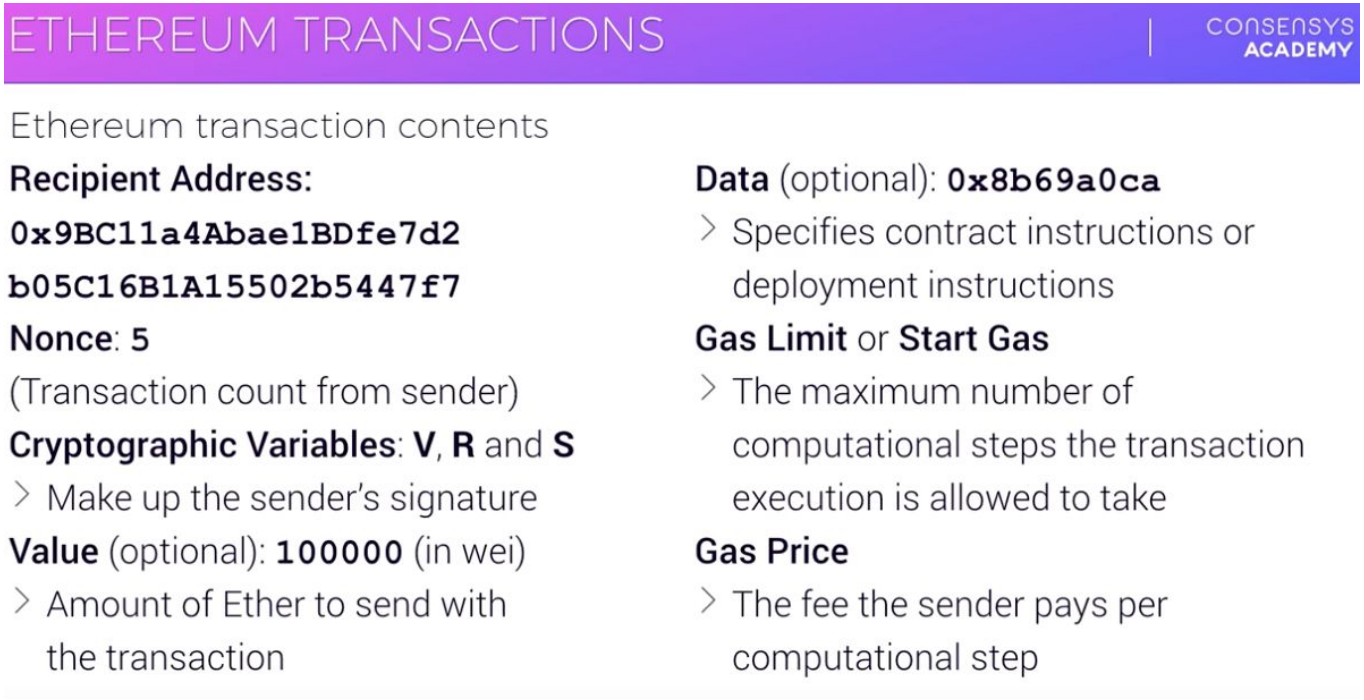
20-byte address gets created when a user generates a private-public keypair.

- Externally Owned Accounts (EOA)
  - State: Balance
- Contract Accounts
  - State: Balance & Storage



Transaction

Message sent from one account to another.



Block

Valid block contains:

- Transaction list
- Uncles list (discovered block not included in main chain)
- Block header:
  - Previous block hash
  - State root (from State Merkle Tree)
  - Transactions root (from Transaction Merkle Tree)
  - Receipts root (from Receipts Merkle Tree)
  - Block number
  - Gas used
  - Timestamp
  - Nonce (value used during mining to demonstrate proof of work for a block)

Why Fees?

- Because of fees, spammers can't flood the network with infinite transactions.
- Infinite loops will run out of funds.
- Fees are incentive for miners to process transactions.

Why Gas?

- Gas is the metering unit of Ethereum Virtual Machine.
- Each operation on the EVM consumes gas.
  - Addition consumes 3 gas.
  - Multiplications consumes 5 gas.
- Gas in paid for with Ether.
- Gas Limit
  - Max gas allowed for the transaction.
- Gas Price
  - How much Ether the sender pays per gas.
- Both Gas Limit and Gas Price are specified per transaction.

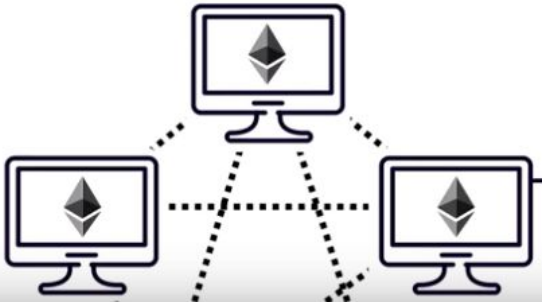
Transaction Fee = Gas Price * Gas Limit

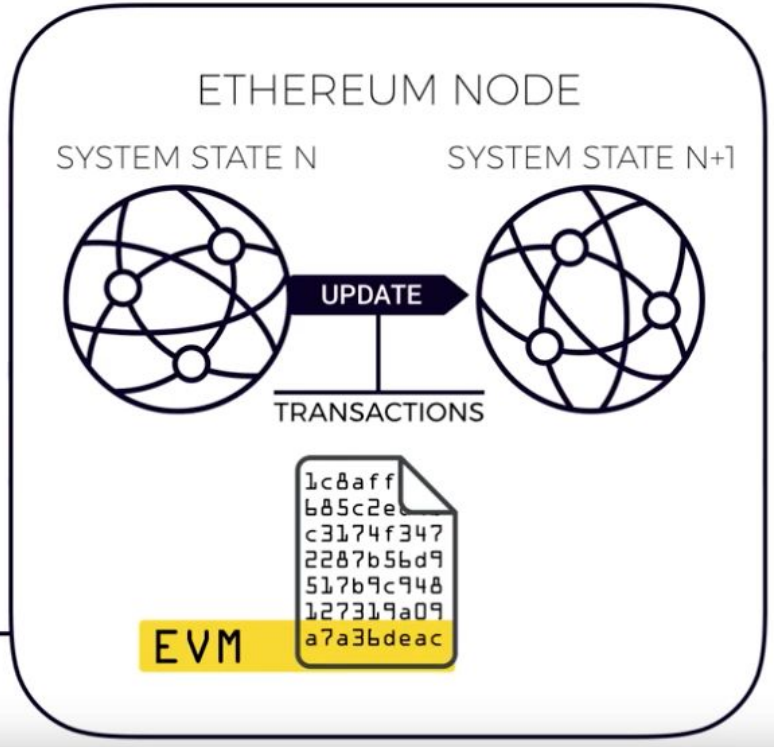
Ethereum Structure

DISTRIBUTED COMPUTING | CONSENSYS ACADEMY

Ethereum Virtual Machine (EVM)

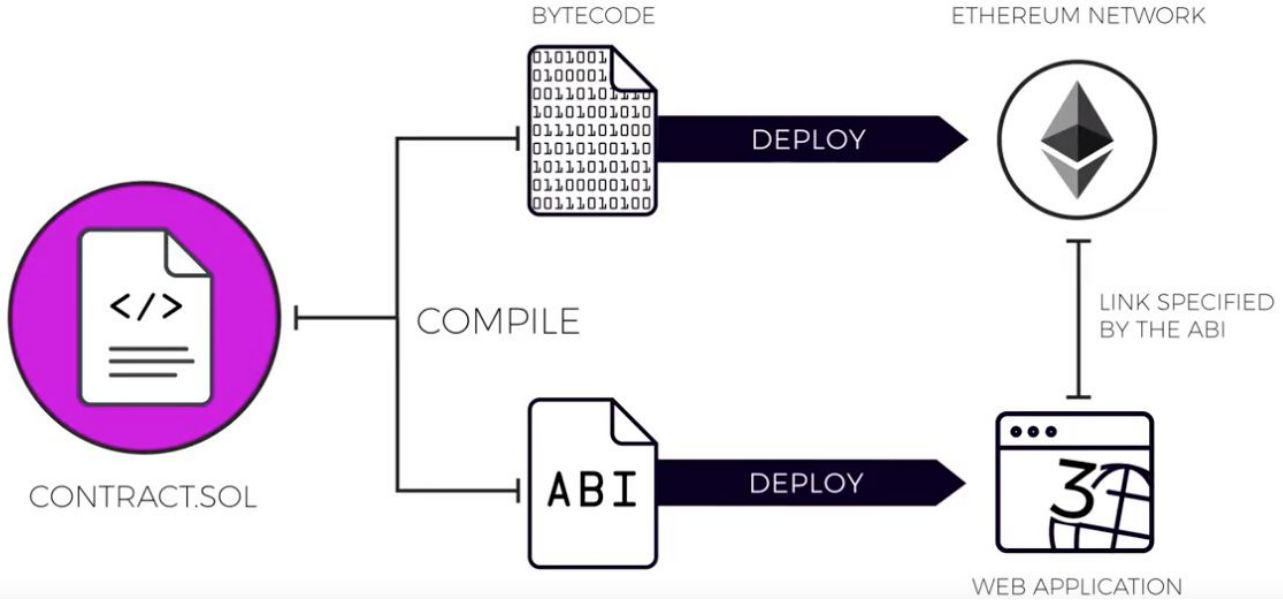
- Runs on every node
- Handles all transaction processing
- Turing complete
- Operates on bytecode



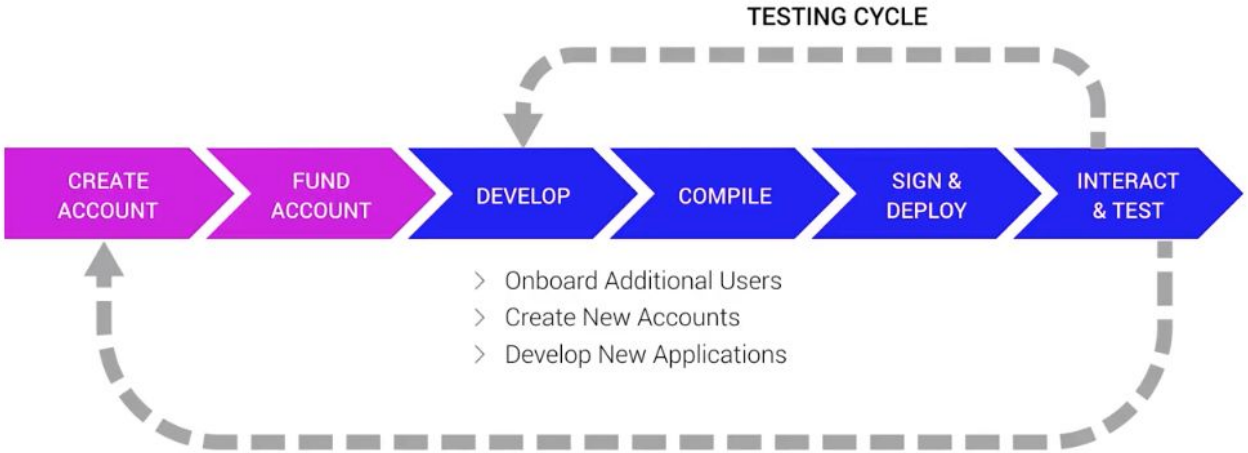


EVM Bytecode & Application Binary Interface (ABI)

- When solidity contract is compiled, it generates Bytecode and ABI.
- Bytecode - Because EVM runs on Bytecode.
- ABI - Because it is a way to interact with contracts from outside the blockchain and from other contracts.
    - ABI is list of contract functions and their arguments in JSON format.
    - ABI translates the Bytecode, so web application (web3) can understand it.



Development Process



Mainnet	chainID 1
Testnet	Ropsten - chainID 3 Rinkeby - chainID 4 Kovan - chainID 42
Private Network	Private Blockchain

Geth (Go-Ethereum, Node)	Public / Private Blockchain
Metamask	Public / Private Blockchain
Truffle	Public / Private Development Framework
Ganache	Local Private Development Blockchain
Remix	Local Private Browser Based Smart Contract IDE

Setup Development Environment

What?	Why?
Install Homebrew	Because Homebrew is package manager for macOS. It helps install packages that Apple didn't provide.
Brew install node	Because NodeJS is JavaScript runtime built on Chrome's V8 JavaScript engine.
Install npm	Because NPM is package manager for JavaScript. It helps install tools like truffle, solidity, ganache-cli. Npm gets installed with NodeJS.
Npm install	To install all dependencies.
Install git	Because Git helps with version controlling.
Sudo npm install -g truffle (-g = global)	Because Truffle is a world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier.
Sudo npm install solc	Because Solidity is contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine (EVM).
Install Ganache	Because Ganache is personal blockchain for Ethereum development you can use to deploy contracts, develop your applications, and run tests.
Install Metamask	Because Metamask is required to send transactions.
Install Mythril	Because Mythril helps to check security vulnerabilities.

Create Test Project

What?	Why?
mkdir test *cd test*	To make directory where truffle project will be set up.
Npm install	To install all dependencies.
Truffle init (optional)	To Initialize a default truffle project inside a folder. It creates all truffle files and folder required for a truffle project.
Truffle unbox react	To unbox React files from Truffle Boxes (which provides list of example projects).
Git init	To initialize git repository.
Touch .gitignore	To create new file to tell git which files it should ignore while doing commit.
Open .gitignore file and add: *.gitignore* *node_modules/*	To tell git to ignore '.gitignore' file and 'node_modules' folder.
Git status	To check status.
Git add .	To add everything to staging area.
Git commit -m "Unboxing Truffle React"	To commit with message.

Create Test Project (cont..)

What?	Why?
Configure network setting in file: ● truffle.js (for Mac) ● truffle-config.js (for Windows)	Because Truffle looks for ‘development’ network by default. Ports: ● Ganache-cli = 8545 ● Truffle develop = 7545
Run Ganache	To run local blockchain instance.
Set Metamask	To connect to local Ganache blockchain ● Copy MNEMONIC from ganache and import to metamask ● Go to settings and in ‘Net Network’ put ‘HTTP://127.0.0.1:7545’
Go to ‘client’ folder and run: *npm run start*	To start web server and make sure everything is working. Runs script in package.json file.
Truffle console (optional	If you want to migrate to testnet or mainnet.
Truffle develop (optional)	If you don’t need to migrate to mainnet and don't want to install and manage a separate blockchain client. Don’t need to run ganache.
Truffle compile (optional)	To create EVM Bytecode and ABI. It creates “build” directory that contains “contracts” directory where json file (corresponding to each contracts) are created. These are artifacts.
Truffle migrate --reset	To deploy contracts to blockchain.
Truffle test	To run tests. Tests can be written in javascript or solidity. Truffle uses mocha testing framework and chai for insertions

Solidity

Programming language to write smart contract in Ethereum.

- Statically typed, supports inheritance, libraries etc.
- Compiled language influenced by C++, Python and JavaScript.
- Elementary (value) types
  - Boolean
  - Integer (int and uint 256 bits)
  - Address
    - 20 byte value with member functions
    - Balance
    - Transfer
    - Send
    - Call
    - Callcode - use delegateCall() instead
    - Delegatecall
  - Byte arrays
    - Fixed size byte arrays
    - Dynamic size byte arrays
  - Enums
    - User defined types
- Complex types
  - Arrays
  - Structs
- Mappings
  - Key-value pair
  - Similar to hash tables
- Functions
  - Public (default)
    - No restriction
  - External
    - Only accessible from outside the contract (unless you use **this**)
  - Private
    - Only accessible from this contract
  - Internal
    - Accessible by this contract and all derived contracts
  - Constant / View
    - Just read from blockchain
    - Doesn't modify any state of contract
    - No gas cost, free to execute
  - Pure
    - Do not read or modify the state
  - Payable
    - To receive ether

Storage and Memory

Storage	Memory	Call Stack
Every contract has its own storage.	Stores only for the duration of function call.	Only hold limited number of values (Limited call depth: 1024).
Stores state variables.	Stores function arguments.	Stores local variables.
Expensive ● because storage variables are written to blockchain and stored in every node	Cheap ● because once function terminates, variables in memory no longer exists	Cheapest (almost free)

CryptoZombies Tutorial

What?	Why?
pragma solidity ^0.4.19;	Because pragma declares which version of Solidity compiler this code should use to prevent issues with future compiler versions potentially introducing changes that could break your code.
contract ZombieFactory {	Because all variables and functions need to be inside a contract.
event NewZombie(uint zombieId, string name, uint dna);	Because events lets app's front-end know that something happened on the blockchain, like 'listening' for certain events and take action when they happen. Like, in this case, zombie was created, so the app can display it.
uint dnaDigits = 16; uint dnaModulus = 10 ** dnaDigits;	Because state variables (variable declared outside of functions) are permanently stored in contract storage (written to the Ethereum blockchain). ● uint - unsigned (non-negative) integer, 256bit (default) ● 10 to the power dnaDigits Variables declared inside functions are stored in memory that disappear when function call ends.

CryptoZombies Tutorial (Cont..)

What?	Why?

struct Zombie {
string name;
uint dna;
}
	Because structs allow you to create more complicated data types that have multiple properties.
	● Struct - Custom data type with group of variables
	● Enum - Custom data type with group of constants

Zombie[] public zombies;	Because array allows you to create a collection of something. ● Fixed - uint[2] ● Dynamic - uint[] Because if array is declared public, Solidity will automatically create a getter method for it.
mapping (uint => address) public zombieToOwner; mapping (address => uint) ownerZombieCount;	Because mapping is needed to create key-value pairs like in hash tables. Mappings are another way of storing organized data in Solidity like arrays.
function _createZombie(string _name, uint _dna) private {	Because naming functions with underscore (_), identifies them as private functions. Because naming parameter variable with underscore (_) differentiate them from global variables. Because In Solidity, functions are public by default. That means anyone (or any other contract) can call your contract's function and execute its code. Therefore they need to be made private.

CryptoZombies Tutorial (Cont..)

What?	Why?
uint id = zombies.push(Zombie(_name, _dna)) - 1;	Because array.push() adds something to the end of the array.

zombieToOwner[id] = msg.sender;
ownerZombieCount[msg.sender]++;
NewZombie(id, _name, _dna);
}
	Because msg.sender gives you the address of the person (or smart contract) who called the current function.

function _generateRandomDna(string _str) private view returns (uint) {
uint rand = uint(keccak256(_str));
return rand % dnaModulus;
}
	Because view (function modifier) only allows to view data. Doesn't change state in Solidity — e.g. it doesn't change any values or write anything.

function createRandomZombie(string _name) public {
require(ownerZombieCount[msg.sender] == 0);
uint randDna = _generateRandomDna(_name);
_createZombie(_name, randDna);
}
}
	Because require only allows you to run the function only once.

CryptoZombies Tutorial (Cont..)

What?	Why?
import "./zombiefactory.sol";	Because import helps split codes into multiple files to make it more manageable.

contract KittyInterface {
  function getKitty(uint256 _id) external view
returns (
  bool isGestating,
  uint256 cooldownIndex
  );
}
	Because interface helps talk to another contract on the blockchain that you don't own.
	
	It is defined same as contract except that we don't have to define function body and just end with semi-colon.

contract ZombieFeeding is ZombieFactory {	Because logical inheritance (such as with a subclass, a Cat is an Animal) can be used to inherit properties of a base contract.

function Ownable() public {
  owner = msg.sender;
  }
	Because constructor function get executed only one time, when the contract is first created. It has same name as the contract

modifier onlyOwner() {
  require(msg.sender == owner); _;
  }
	Because modifiers help modify other functions, usually to check some requirements prior to execution.

CryptoZombies Tutorial (Cont..)

What?	Why?

function levelUp(uint _zombieId) external payable {
  require(msg.value == levelUpFee);
  zombies[_zombieId].level++;
}
	Because payable makes a function to receive ether.

function withdraw() external onlyOwner {
  owner.transfer(this.balance);
}
	Because owner.transfer will send balance.

using SafeMath for uint256;	A library is a special type of contract in Solidity. One of the things it is useful for is to attach functions to native data types.
var myContract = new web3js.eth.Contract(myABI, myContractAddress);	Initiates contract. ABI stands for Application Binary Interface. Basically it's a representation of your contracts' methods in JSON format that tells Web3.js how to format function calls in a way your contract will understand.
event Transfer(address indexed _from, address indexed _to, uint256 _tokenId);	Because indexed keyword allows to filter events and only listen for changes related to the current user.
function() public { revert();	Because Fallback function is called if other functions don’t match call or send ether without data. Typically, called when invalid data is sent.

Events and Logs

- Because events provide return values from smart contracts to UI.
- They also provide notifications when transactions are complete. They are not emitted until the transaction has been successfully mined.
- They can act a cheaper form of storage.
- They can be used to debugging purposes during development.

- Logs are not part of blockchain but they are verified by the blockchain as the transaction receipt hashes are stored inside the blocks.
- Logs cost 8 gas per byte
- Events cost 625 gas per byte

Inheritance

To inherit states and functions so code can be reused.

- Inherits
  - State
  - Function
- Function overrides
  - Same name
  - Same input
  - Same output
- Function, modifier and event names must be unique
- Abstract contracts
  - Missing function implementations
  - Can be used as base contracts
- Interface contracts
  - Cannot implement any functions
  - Cannot inherit other contracts
  - No constructor, variables, structs, enums

Libraries & Ethereum Package Manager

For the purpose of code reuse.

Library

- Libraries are contracts that do not have storage.
- They cannot hold ether.
- They cannot inherit or be inherited by other contracts.

Ethereum Package Manager

- EthPM is npm for Ethereum contracts
- truffle install <package name>

Tests

- To implement Tests Driven Development (TDD).
  - Write tests first then implement functions to make the tests pass.
- Writing tests is not about finding bugs, but defining contract behavior.
- Can be written in Javascript or Solidity.
- Contract name must begin with ‘Test’ (capital ‘T’). Function tests must start with ‘test’ (small ‘t’).
- Truffle provides Assert.sol library.
- Truffle uses mocha testing framework and chai for insertions.

```
// Testing the adopt() function
function testUserCanAdoptPet() public {
  uint returnedId = adoption.adopt(8);

  uint expected = 8;

  Assert.equal(returnedId, expected, "Adoption of pet ID 8 should be recorded.");
}
```

Exploits and Dangers

Reentrancy	• One of the major dangers of calling external contracts is that they can take over the control flow, and make changes to your data. • Attacker can call functions repeatedly before first invocation of the function is finished. This is what caused the DAO attack.	Don't call external function until all internal work is done.
Cross function reentrancy	• Reentrancy that can occur across multiple functions.	Don't call external function until all internal work is done.
Transaction Ordering and Timestamp Dependence	• Transactions broadcasted to the network but not yet included in a block are in the mempool. • Since transactions are in the mempool before they make it into a block, anyone can know what transactions are about to occur on the network.	In exchanges, use batch auctions (also prevents across high frequency trading). Use pre-commit scheme.
Integer Overflow and Underflow	• Overflow - If a balance reaches the maximum uint value (2^256) it will circle back to zero. • Underflow - If a uint is made to be less than zero, it will cause an underflow and get set to its maximum value	Make sure they don't reach their maximum value.
Denial of Service	• Another danger of passing execution to another contract is a denial of service attack. • For example, iterating through array to pay multiple users can cause whole payout system to fail if one address is forcing an error.	Use pull over push payments.
Force Sending Ether	• Another danger is using logic that depends on the contract balance. • Attacker can use selfdestruct function, will force the destroyed contract's funds to be sent to the target.	Add contract logic to prevent this.

Smart Contract Best Practices

- Prepare for bugs, they are inevitable.
- Circuit Breaker design pattern
  - Pause contract if things go wrong.
- Roll out carefully.
  - Use automated Unit Testing to make sure logic is behaving as expected.
  - Use testnets.
  - Third party security audits.
- Bug bounties
- Greater complexity == more bugs
- Reuse audited code and libraries.
- Only use blockchain when necessary.
- Always verify input data.
- Use
  - Https
  - Two factor authentication
  - Encrypt
- Use msg.sender instead of tx.origin
- Gas limits
  - Don’t loop over arrays of undetermined length
  - Run tests for gas usage
  - Limit the length of user supplied data

Optimizing Gas

Because both deploying a contract and calling contract functions cost gas.

- Reduce number of loops. Zero, if possible.
- Modifying storage variables in a loop can be very expensive and should be avoided unless absolutely necessary.
- Dynamic-size byte arrays (byte[ ]) wastes space, so use fixed-size byte arrays (from bytes1 to bytes32) since they are cheaper.

Smart Contract Design Patterns

Fail early and fail loud	● Use ‘require’ than ‘if’ statement cause ‘require’ will throw exceptions, ‘if’ will not.
Restricting Access	● Restrict other contracts’ access to the state by making state variables private. ● Restrict function access so that only specific addresses are permitted to execute functions.
Auto Deprecation	● Close contracts that should expire after a certain time.
Mortal	● Add ability to destroy a contract and remove from blockchain.
Push over push payments	● Allow users to withdraw rather then using send transaction. ● This will prevent against re-entrancy and denial of service attacks.
Circuit Breakers	● Allow contract functionality to be stopped. ● This is useful when bugs are detected on live contracts. ● Freezing the contract would reducing harm before a fix is implemented.
Speed Bump	● Speed bumps slows down actions so that if malicious actions occur, there is time to recover.

Upgradeable Contracts

Because contracts in Ethereum are immutable, new contracts need to be published with new code.

- Two approaches to upgrading contracts
  - Registry - Stores addresses of latest version of contract.
  - Forward data and calls - Use relay contract to forward data and calls to correct contract.
- Keep components modular
  - Allows to upgrade certain components while keeping others same.
- Process to manage data migration is still exploratory.

Why Oracles?

Because Oracles brings information from outside blockchain into blockchain.

- Like API for the blockchain.
- Tricky part is trusting API providers.
- Oraclize queries uses “If this, then that” logic model.

Decentralized Oracle

- Bring information to blockchain without having to rely on 3rd parties.

Why Ethereum Name Service (ENS)?

Because ENS converts machine readable identifiers to human readable names.

- Ethereum addresses
  - 0x634f7f5af511636994d5060c5bcab728e35e9840 → ethereumaddress.eth
- IPFS hashes
  - QmWeSMxMWpsrsJdBU6Zqc6DXZEf4WXHkPzBAdmPjmmHUna → ipfshash.eth

Two main components

- Registry
  - Single contract that lists all domains and subdomains with associated data.
  - Stores:
    - Owner
      - External account
      - Contract
    - Address of Resolver contract
    - Time to live
- Resolver
  - Translate names to identifiers (Ethereum address, IPFS hash, Swarm hash etc).

Why Interplanetary File System (IPFS)?

Because IPFS makes web distributed.

What?	Why?
● Distributed P2P file system ● No servers ● Content addressed ● Hash of file ● A single network of peers sharing git objects	● Maintains Data Integrity ○ Changing data will change hash, therefore, changing address of the file. ● Decentralized ○ No central point of failure ○ Resilient ● Storing data is cheaper in IPFS than in blockchain. ○ Store data on IPFS. ○ Store IPFS hash in smart contracts.

IPFS - How to publish dapp in IPFS

What?	Why?
Brew install ipfs Ipfs init Ipfs daemon Ipfs swarm peers Ipfs add -r dist/ Ipfs name publish <hash of folder> https://gateway.ipfs.io/ipfs/<hash of folder>	To install IPFS To initialize new node To run the node To share contents with peers (nodes) To add files to ipfs To publish folder to ipfs so the folder can be accessed via ipfs Link to access dapp via browser

More…

Lisp Like Language (LLL)	"Lisp Like Language (LLL) is a low level language similar to Assembly. It is meant to be very simple and minimalistic; essentially just a tiny wrapper over coding in EVM directly."
Vyper	Vyper is an experimental, contract-oriented, pythonic programming language that targets the Ethereum Virtual Machine.
Ethereum Improvement Proposals (EIP)	Ethereum is an open source project with no one company or organization that controls the direction of the project. There is an open governance model where everyone is free to propose and discuss changes to the system. EIP is a way to propose new improvements to the network.