Blockchain Fundamentals
Intro to Web3

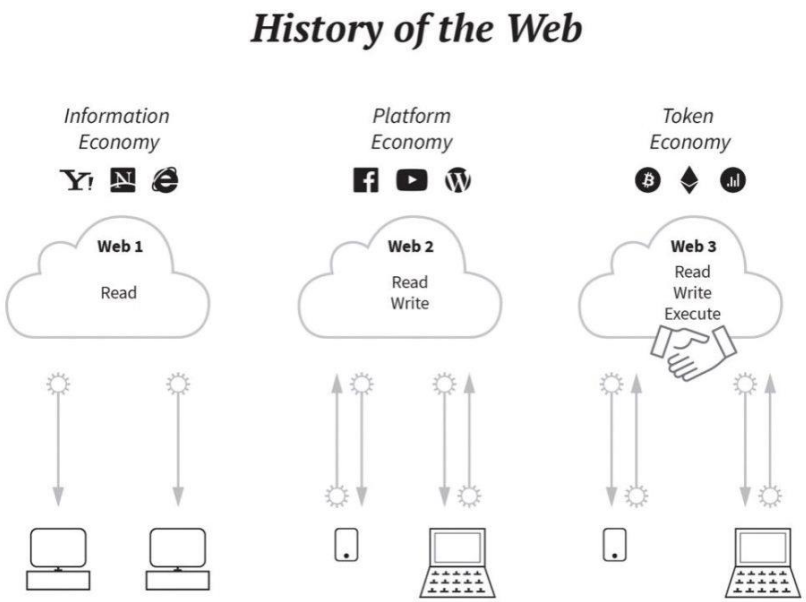
Hamza Salem

What is?

- Web
- Database
- P2P(Torrent)

Web3

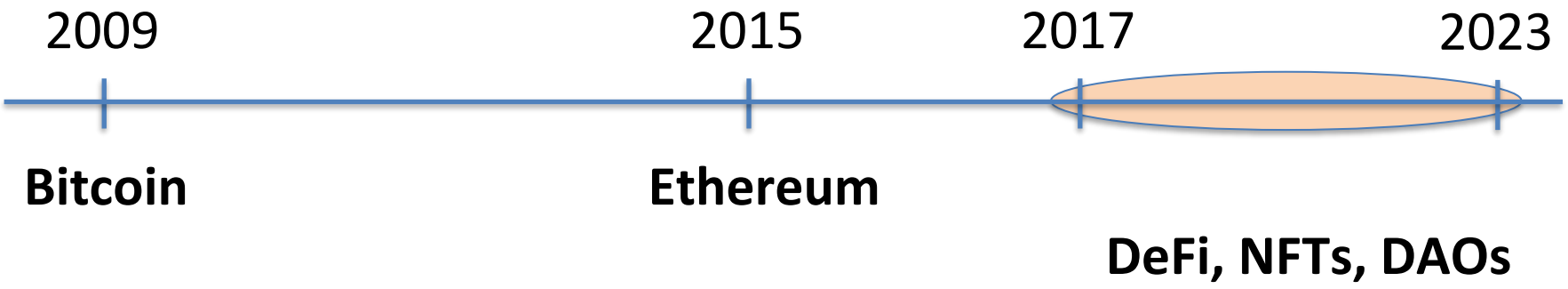
Web3 is an idea for a new iteration of the **World Wide Web** which incorporates concepts such as **decentralization, blockchain technologies, and token-based economics.**



Meme Review

The history of blockchain and Bitcoin

First computing/hosting platform to deploy Apps



Database allows Append only ...
Protected by **Replication** and **consensus** algorithm.

Blockchain Story

In 2008, a groundbreaking paper entitled Bitcoin: A Peer-to-Peer Electronic Cash System was written on the topic of peer-to-peer electronic cash under the pseudonym Satoshi Nakamoto. It introduced the term chain of blocks. No one knows the actual identity of Satoshi Nakamoto. After introducing Bitcoin in 2009, he remained active in the Bitcoin developer community until 2011. He then handed over Bitcoin development to its core developers and simply disappeared. Since then, there has been no communication from him whatsoever, and his existence and identity are shrouded in mystery. The term chain of blocks evolved over the years into the word blockchain.

Bank Problems (Ledgers)

Bank Problems (Ledgers)

1) Name of the Business:						
2) Name of the document - General Ledger						
Date	**Particulars**	**DR or CR**	**Account No.**	**Post ref**	**Debit $**	**Credit $**
1/1/2014	Owner contributes $100					
	Bank	Dr			100	
	Capital	Cr				100
31/3/14	The Ship buys Pall Mall for $200					
	Bank	Decreasing - Cr				200
	Property	Increasing - Dr			200	
1/4/2014	The boot lands on Pall Mall and pays $10 rent to the ship (we are the ship)					
	Bank	increasing - Dr			10	
	Revenue - rent	increasing - Cr				10

What happen when you open your Spearbank App?

"I should go out for lunch today"

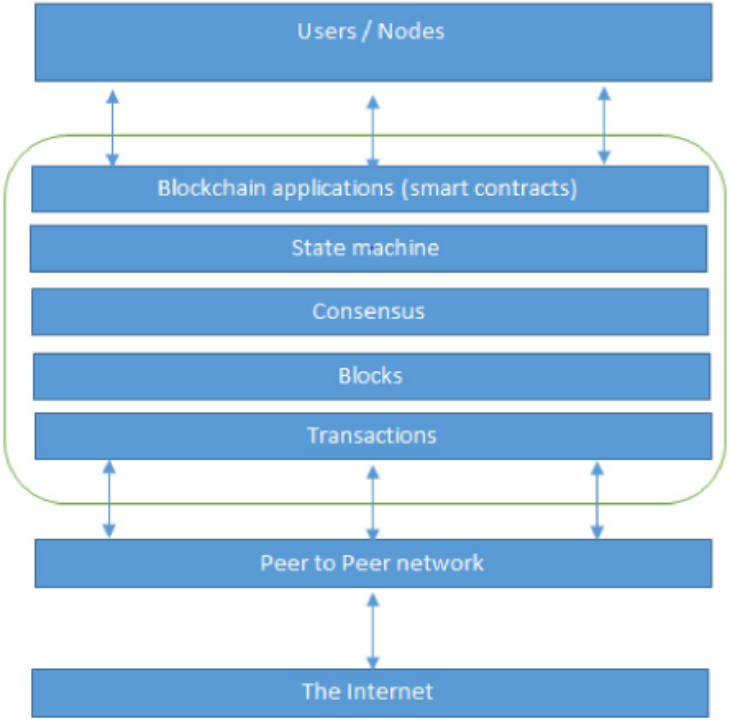
opens the mobile banking app

Account balance:

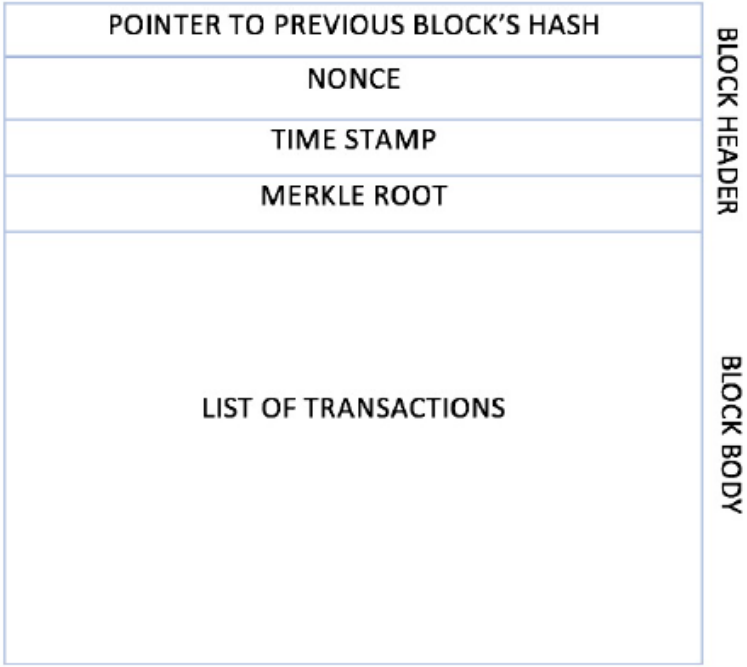
😩

Terminology

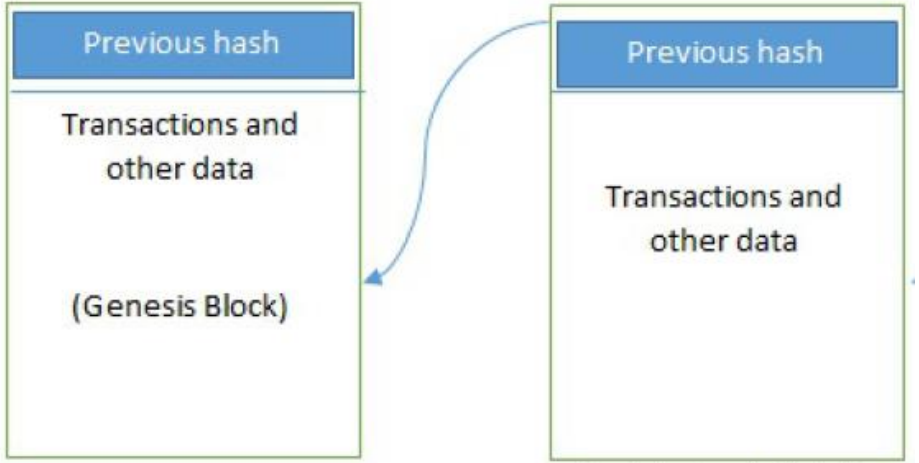
- Peer-to-peer
- Distributed ledger
- Cryptographically-secure
- Append-only/time-ordered sequential order
- Updateable via consensus



Blocks Architecture



The generic structure of a block.

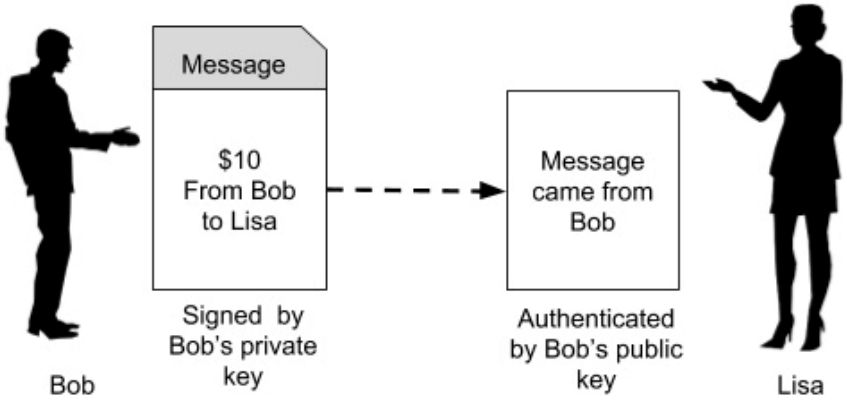


Generic structure of a blockchain

Public-key cryptography in Bitcoin

Bitcoin uses a type of public-key cryptography called **Elliptic Curve Cryptography (ECC)** to generate and manage its digital signatures. Public-key cryptography uses a pair of mathematically related keys - a public key and a private key - to encrypt and decrypt data.

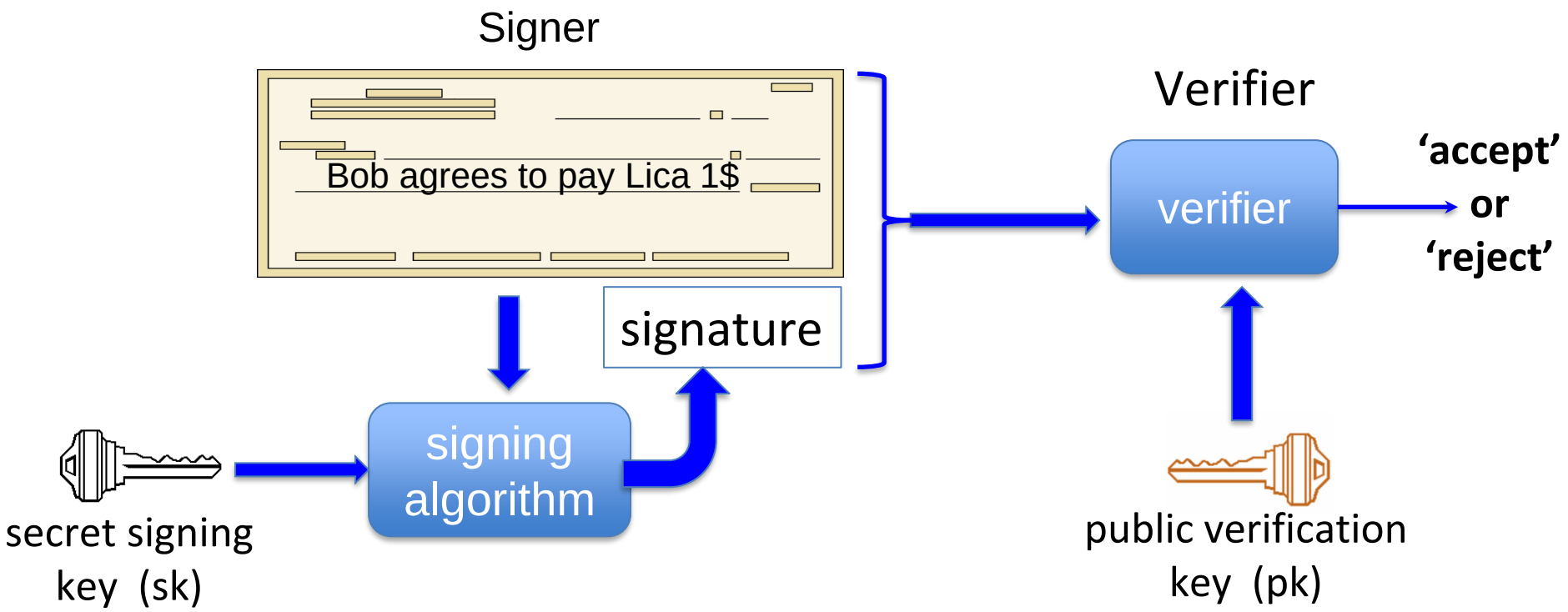
In Bitcoin, the private key is kept secret by the owner, while the public key is shared with the network.



Digital signature

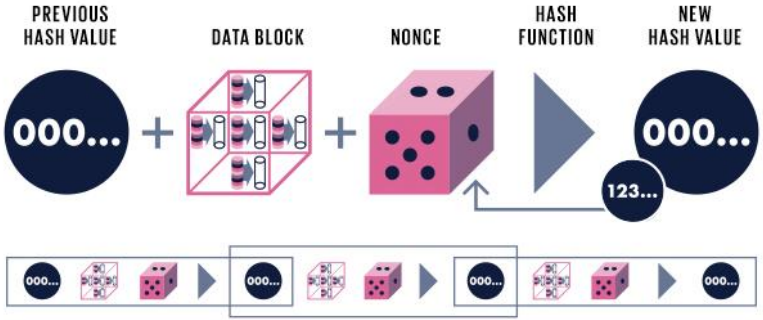
<u>Def</u>: a signature scheme is a triple of algorithms:

- **Gen**(): outputs a key pair (pk, sk)
- **Sign**(sk, msg) outputs sig. σ
- **Verify**(pk, msg, σ) outputs ‘accept’ or ‘reject’


Signer
Bob agrees to pay Lica 1$
signature
signing algorithm
secret signing key (sk)
Verifier
verifier
'accept' or 'reject'
public verification key (pk)

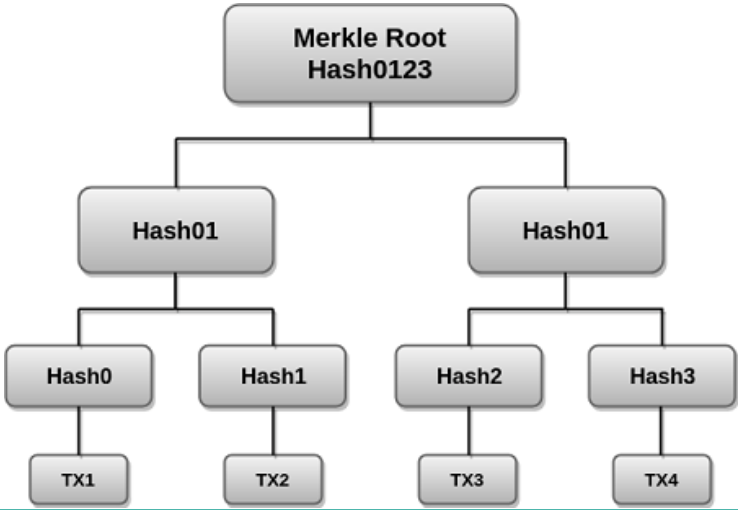
Hash functions in Bitcoin

Bitcoin uses a type of hash function called SHA-256 (Secure Hash Algorithm 256-bit) to secure its transactions and to create new blocks in the blockchain. A hash function takes input data of any size and produces a fixed-size output (called a hash) that is unique to the input data. In Bitcoin, each block in the blockchain contains a hash of the previous block, which creates a chain of blocks that are linked and secured by cryptography.



Merkle trees in Bitcoin

Bitcoin also uses a type of data structure called a Merkle tree to efficiently store and verify the integrity of large amounts of transaction data. A Merkle tree is a hierarchical structure of hashes that allows nodes in the network to verify that a particular transaction is included in a block without having to download and verify the entire block.



Tiers of blockchain technology

- **Blockchain 1.0:** This tier was introduced with the invention of Bitcoin.
- **Blockchain 2.0:** This second blockchain generation is used by financial services and smart contracts.
- **Blockchain 3.0:** This third blockchain generation is used to implement applications beyond the financial services industry and is used in government, health, media, the arts, and justice.
- **Blockchain X.0:** This generation represents a vision of blockchain singularity where one day there will be a public blockchain service available that anyone can use just like the Google search engine

Features of a blockchain

- Distributed consensus
- Transaction verification
- Platform for smart contracts
- Transferring value between peers
- Generation of cryptocurrency
- Smart property
- Provider of security
- Immutability
- Uniqueness

Terminology

- Distributed Ledger Technology (DLT)
- Public blockchains
- Private blockchains
- Semi Private blockchains
- Permissioned ledger
- Shared ledger
- Tokenized blockchains
- Tokenless blockchains
- Consensus mechanism

Consensus Algorithms in blockchain

- Proof of Work (PoW)
- Proof of Stake (PoS)
- Delegated Proof of Stake (DPoS)
- Proof of Elapsed Time (PoET)
- Proof of Deposit (PoD)
- Proof of Importance (PoI)

....etc

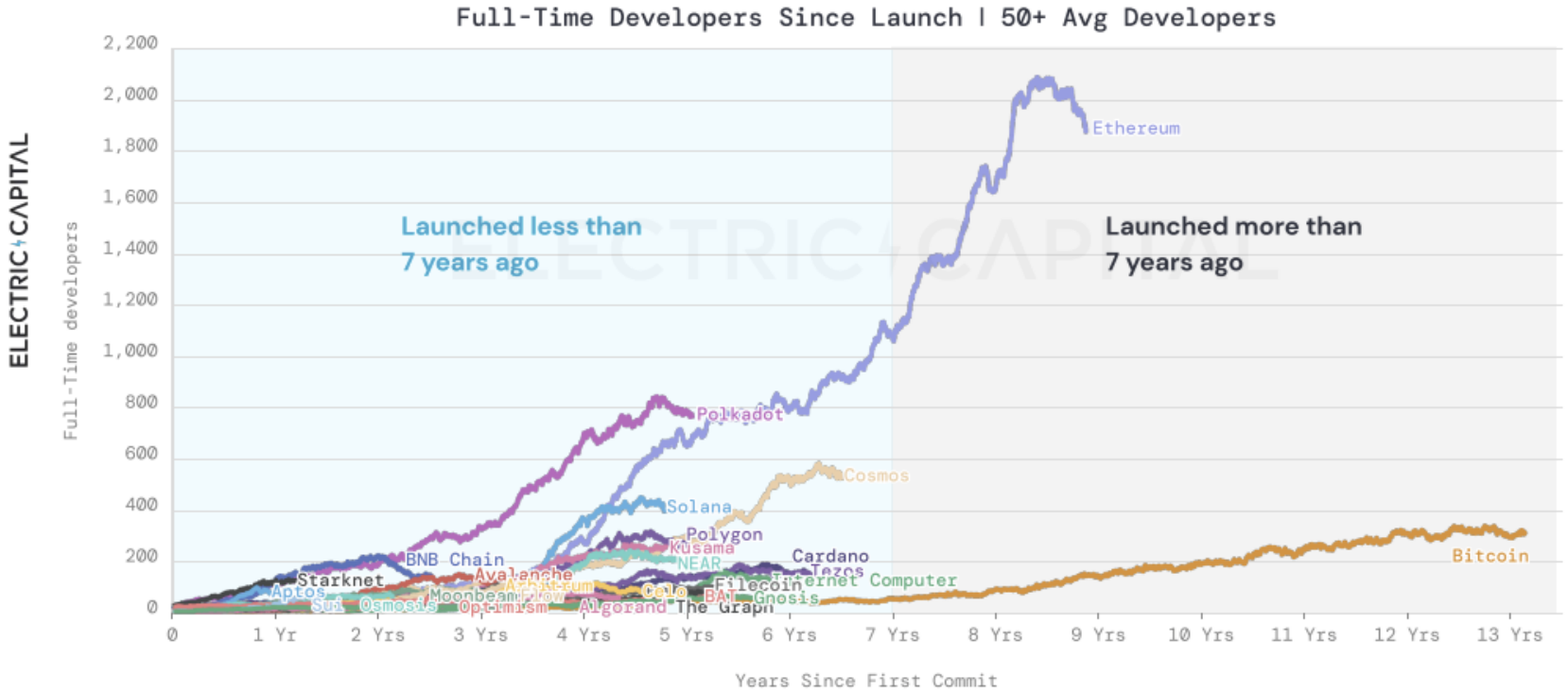
cryptofunny

Trump: Bitcoin is not money and is based on thin air.
Federal Reserve:

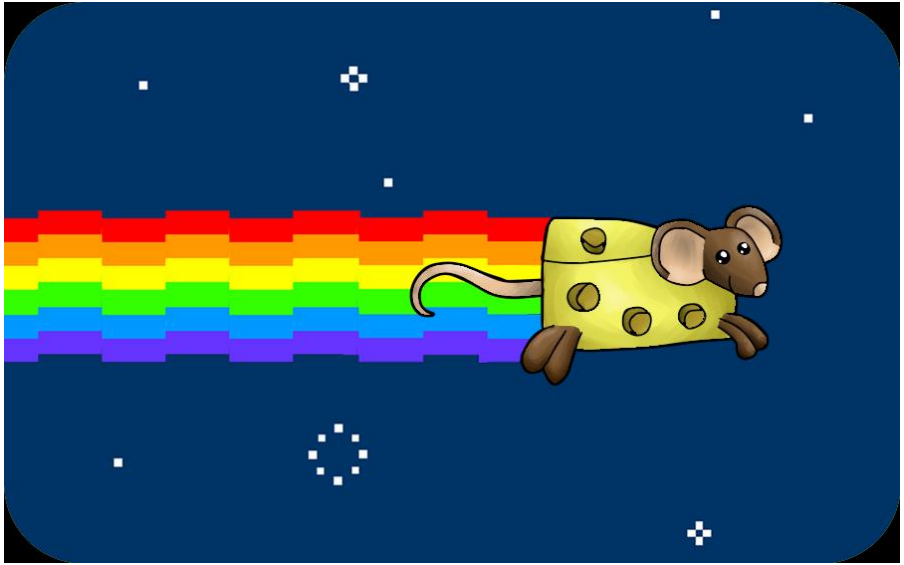
Transaction Volume

#	Name	Price	1h %	24h %	7d %	Market Cap	Volume(24h)	Circulating Supply	Last 7 Days
1	Bitcoin BTC	$27,852.54	▼0.36%	▼0.23%	▲0.41%	$538,651,839,892	$13,113,464,195 471,087 BTC	19,339,418 BTC	
2	Ethereum ETH	$1,851.51	▼0.78%	▼1.48%	▲3.15%	$223,029,057,188	$9,603,253,730 5,179,912 ETH	120,457,776 ETH	
3	Tether USDT	$1.00	▲0.05%	▲0.05%	▲0.04%	$80,245,820,240	$22,927,859,085 22,922,892,673 USDT	80,195,385,809 USDT	
4	BNB BNB	$310.29	▼0.36%	▼0.63%	▼1.48%	$48,991,524,406	$505,533,074 1,628,155 BNB	157,887,127 BNB	
5	USD Coin USDC	$1.00	▼0.04%	▲0.03%	▲0.01%	$32,763,001,936	$3,661,672,569 3,663,321,498 USDC	32,762,839,295 USDC	

EXCLUDING BTC & ETH, ALL TOP 200 ECOSYSTEMS WITH 50+ FULL-TIME DEVS LAUNCHED UNDER 7 YEARS AGO

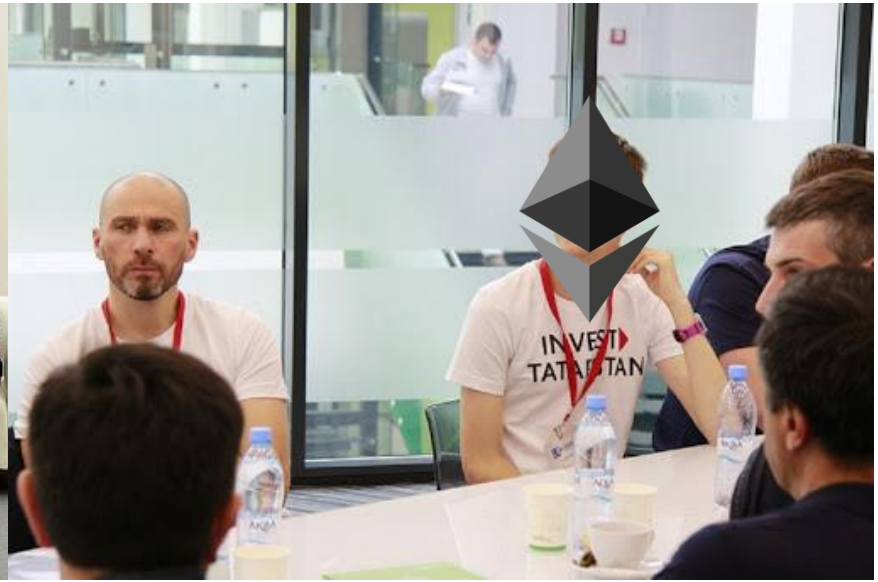


https://github.com/electric-capital/developer-reports/blob/master/dev_report_2022.pdf



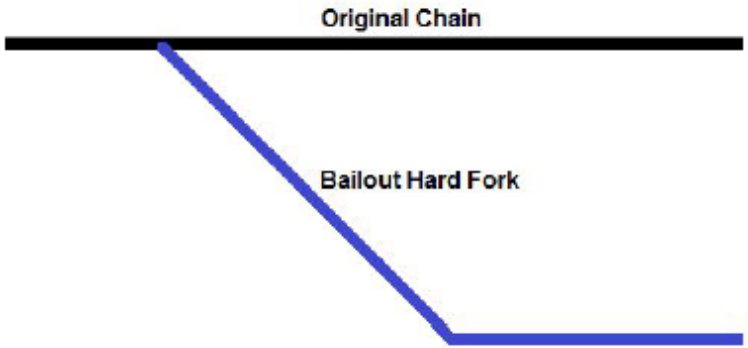

ПРЯМОЙ ЭФИР
ИЗ БЛОКЧЕЙН-КОММУНЫ
ЦЕНТР
БЛОКЧЕЙН
КОМПЕТЕНЦИЙ
ВЭБ
2.0


ПРЯМОЙ ЭФИР
ИЗ БЛОКЧЕЙН-КОММУНЫ
ЦЕНТР
БЛОКЧЕЙН
КОМПЕТЕНЦИЙ
ВЭБ
2.0



Ethereum Classic

The story of ethereum



" Theft of $50 million worth of Ether the Ethereum network split into two separate blockchains – the altered history was named Ethereum and the unaltered history was named Ethereum Classic "

Blockchain Demo

https://andersbrownworth.com/blockchain/block

Web3 Dapp Demo

https://Cpptokens.com

The 2nd-generation blockchain (Ethereum)

Ethereum

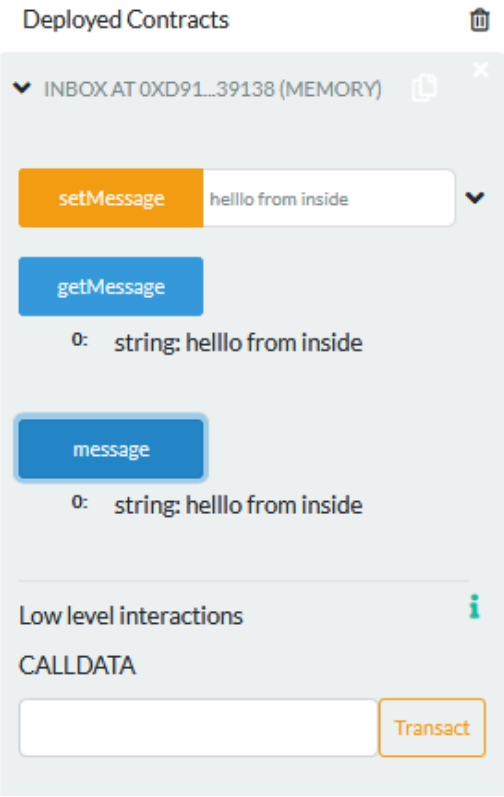
- **Ethereum** is an open source, globally decentralized **computing infrastructure** that **executes programs called smart contracts**. It uses a blockchain to synchronize and store the system's state changes, along with a cryptocurrency called Ether (ETH) to meter and constrain execution resource costs.
- The Ethereum platform enables developers to **build powerful decentralized applications** with built-in economic functions. While providing high availability, auditability, transparency, and neutrality, it also reduces or eliminates censorship and reduces certain counterparty risks. Ethereum's language is Turing complete, meaning that Ethereum can straightforwardly function as a general-purpose computer

Smart Contract

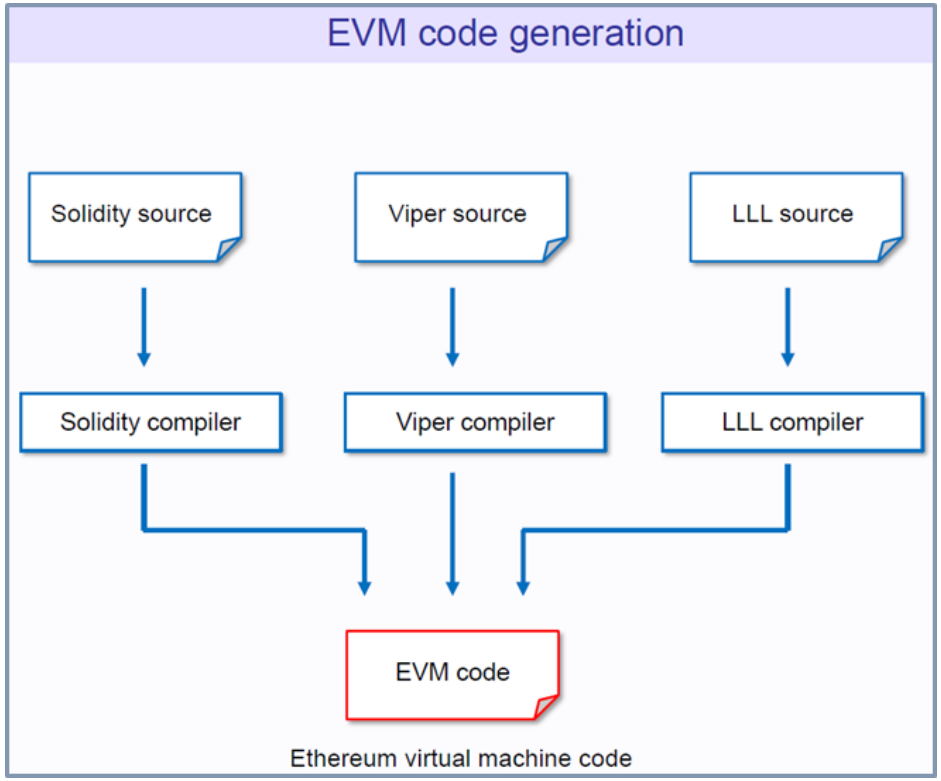
- **Smart contract :** refer to **immutable computer programs that run deterministically in the context of an Ethereum Virtual Machine** (EVM) as part of the Ethereum network protocol — i.e., on the decentralized Ethereum world computer.
- Life cycle for smart contract ：
  - **Write** smart contract in some high-level language.
  - **Compile** to the low-level EVM bytecode.
  - **Deploy** on the Ethereum platform using a special contract creation transaction.
  - **Execution**..

Solidity Smart Contract example

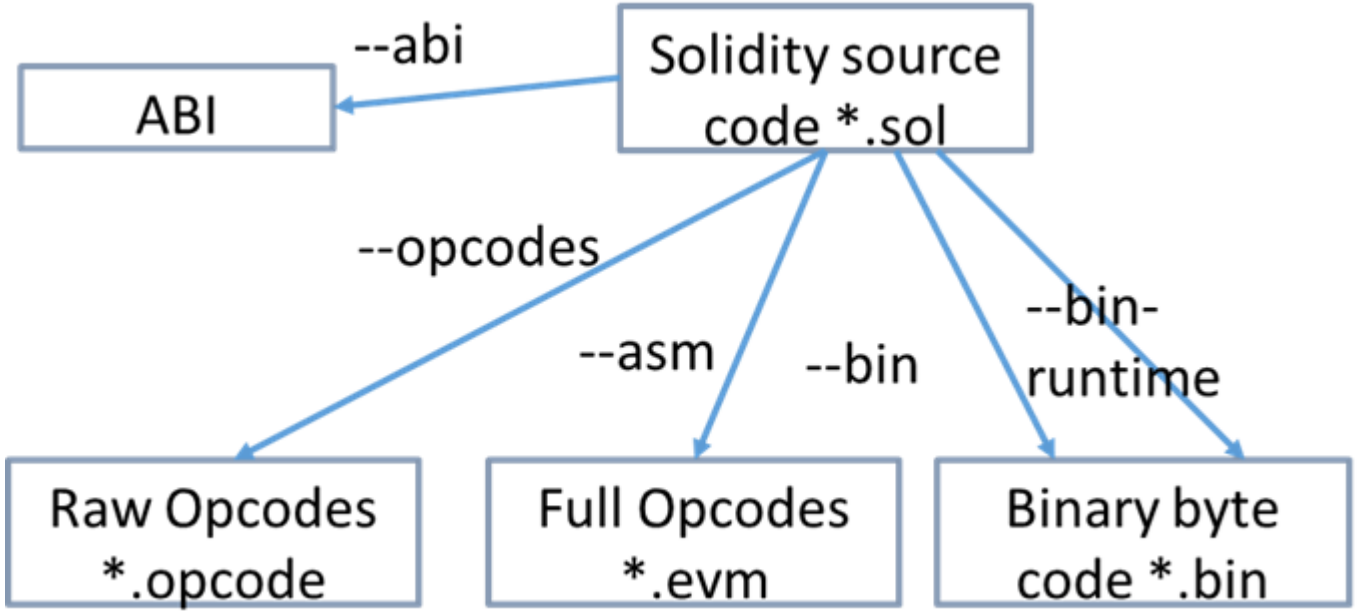
```
pragma solidity  ^0.5.1;
contract Inbox{
string public message;
 constructor (string memory initialMessage) public {
message= initialMessage;
}
function setMessage(string memory newMessage)public{
    message=newMessage;
}
function getMessage()public view  returns(string memory){
    return message;
}
}
```



EVM


EVM code generation
Solidity source
Viper source
LLL source
Solidity compiler
Viper compiler
LLL compiler
EVM code
Ethereum virtual machine code

Compile


--abi
ABI
Solidity source code *.sol
--opcodes
--asm
--bin
--bin-runtime
Raw Opcodes *.opcode
Full Opcodes *.evm
Binary byte code *.bin

```
// Our first contract is a faucet!
contract Faucet {

    // Give out ether to anyone who asks
    function withdraw(uint withdraw_amount) public {

        // Limit withdrawal amount
        require(withdraw_amount <= 100000000000000000);

         // Send the amount to the address that requested it
         msg.sender.transfer(withdraw_amount);
    }

    // Accept any incoming amount
    function () public payable {}

}
```

```
$ solc --optimize --bin Faucet.sol
======= Faucet.sol:Faucet =======
Binary:
6060604052341561000f57600080fd5b60cf8061001d6000396000f3006060604052600436106 03e5
763ffffffff7c0100000000000000000000000000000000000000000000000000000000006000350416
632e1a7d4d81146040575b005b3415604a57600080fd5b603e60043567016345785d8a00008111156
06357600080fd5b73ffffffffffffffffffffffffffffffffffffffff331681156108fc0282604051
6000604051808303818588 8f19350505050151560a057600080fd5b505600a165627a7a723058203
556d79355f2da19e773a9551e95f1ca7457f2b5fbbf4eacf7748ab59d2532130029
```

```
$ solc --abi Faucet.sol
======= Faucet.sol:Faucet =======
Contract JSON ABI
[{"constant":false,"inputs":[{"name":"withdraw_amount","type":"uint256"}], \
"name":"withdraw","outputs":[],"payable":false,"stateMutability":"nonpayable", \
"type":"function"},{"payable":true,"stateMutability":"payable", \
"type":"fallback"}]
```

```
/* "Example.sol":26:132  contract example {... */
  mstore(0x40, 0x60)
    /* "Example.sol":74:130  function example() {... */
  jumpi(tag_1, iszero(callvalue))
  0x0
  dup1
  revert
tag_1:
    /* "Example.sol":115:125  msg.sender */
  caller
    /* "Example.sol":99:112  contractOwner */
  0x0
  dup1
    /* "Example.sol":99:125  contractOwner = msg.sender */
  0x100
  exp
  dup2
  sload
  dup2
  0xffffffffffffffffffffffffffffffffffffffff
  mul
  not
  and
  swap1
  dup4
  0xffffffffffffffffffffffffffffffffffffffff
  and
  mul
  or
  swap1
  sstore
  pop
    /* "Example.sol":26:132  contract example {... */
  dataSize(sub_0)
  dup1
  dataOffset(sub_0)
  0x0
  codecopy
  0x0
  return
stop

sub_0: assembly {
        /* "Example.sol":26:132  contract example {... */
      mstore(0x40, 0x60)
      0x0
      dup1
      revert

    auxdata: 0xa165627a7a7230582056b99dcb1edd3eece01f27c9649c5abcc14a435efe3b...
}
```

```
PUSH1 0x60 PUSH1 0x40 MSTORE CALLVALUE ISZERO PUSH1 0xE JUMPI PUSH1 0x0 DUP1
REVERT JUMPDEST CALLER PUSH1 0x0 DUP1 PUSH2 0x100 EXP DUP2 SLOAD DUP2 PUSH20
0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF MUL NOT AND SWAP1 DUP4 PUSH20
0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF AND MUL OR SWAP1 SSTORE POP PUSH1
0x35 DUP1 PUSH1 0x5B PUSH1 0x0 CODECOPY PUSH1 0x0 RETURN STOP PUSH1 0x60 PUSH1
0x40 MSTORE PUSH1 0x0 DUP1 REVERT STOP LOG1 PUSH6 0x627A7A723058 KECCAK256 JUMP
0xb9 SWAP14 0xcb 0x1e 0xdd RETURNDATACOPY 0xec 0xe0 0x1f 0x27 0xc9 PUSH5
0x9C5ABCC14A NUMBER 0x5e INVALID EXTCODESIZE 0xdb 0xcf EXTCODESIZE 0x27
EXTCODESIZE 0xe2 0xb8 SWAP10 0xed 0x
```

https://remix.ethereum.org/

Demo