ethereum

BLOCKCHAIN APP PLATFORM

TEMAS

História do Ethereum
O fundador: Vitalik Buterin
Publicação na Bitcoin Magazine
Personagens fundamentais
ICOs: Initial Coin Offerings
O que é o Ethereum
Completude de Turing
Nick Szabo e os Smart Contracts
EVM: Ethereum Virtual Machine
Linguagens de programação para Smart Contracts
Solidity


VITALIK BUTERIN
Kolomna,Rússia
1994

Vitalik Buterin

Criador do protocolo
Criança superdotada
2011 primeiro contato com o Bitcoin
Co-fundou a Bitcoin Magazine
Propôs mudanças no protocolo => não foi aceito
Começou então seu próprio projeto
Rumor de sua morte: bloco # 3.930.000

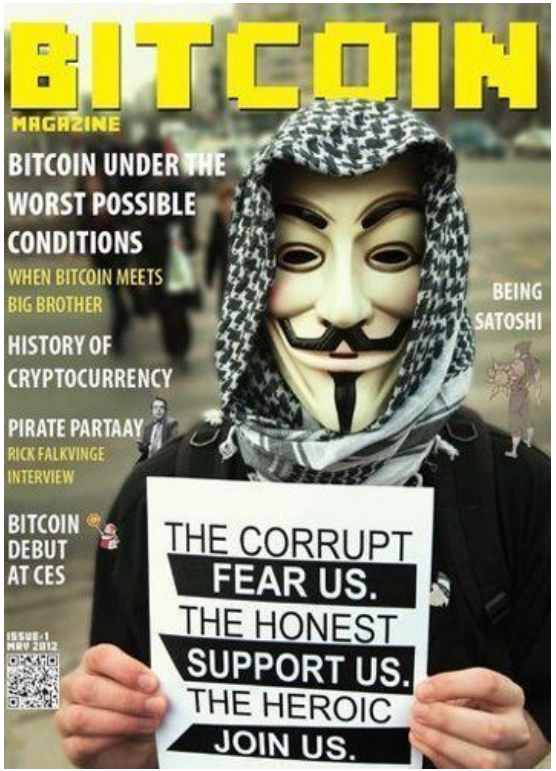
PUBLICAÇÃO NA BITCOIN MAGAZINE

BITCOIN MAGAZINE

https://bitcoinmagazine.com/

Artigos de Vitalik na revista

Publicação introduzindo o Ethereum



PERSONAGENS FUNDAMENTAIS: BITCOIN

Satoshi Nakamoto?

Adam Back

Nick Szabo

Hal Finney

PERSONAGENS FUNDAMENTAIS - Ethereum

Vitalik Buterin

Mihai Alisie

Gavin Wood

Jeffrey Wilcke

Joseph Lubin

Anthony Di Iorio

O QUE É O ETHEREUM?

ethereum

ETHEREUM

Blockchain geral: mesmas características

Plataforma para inovação:

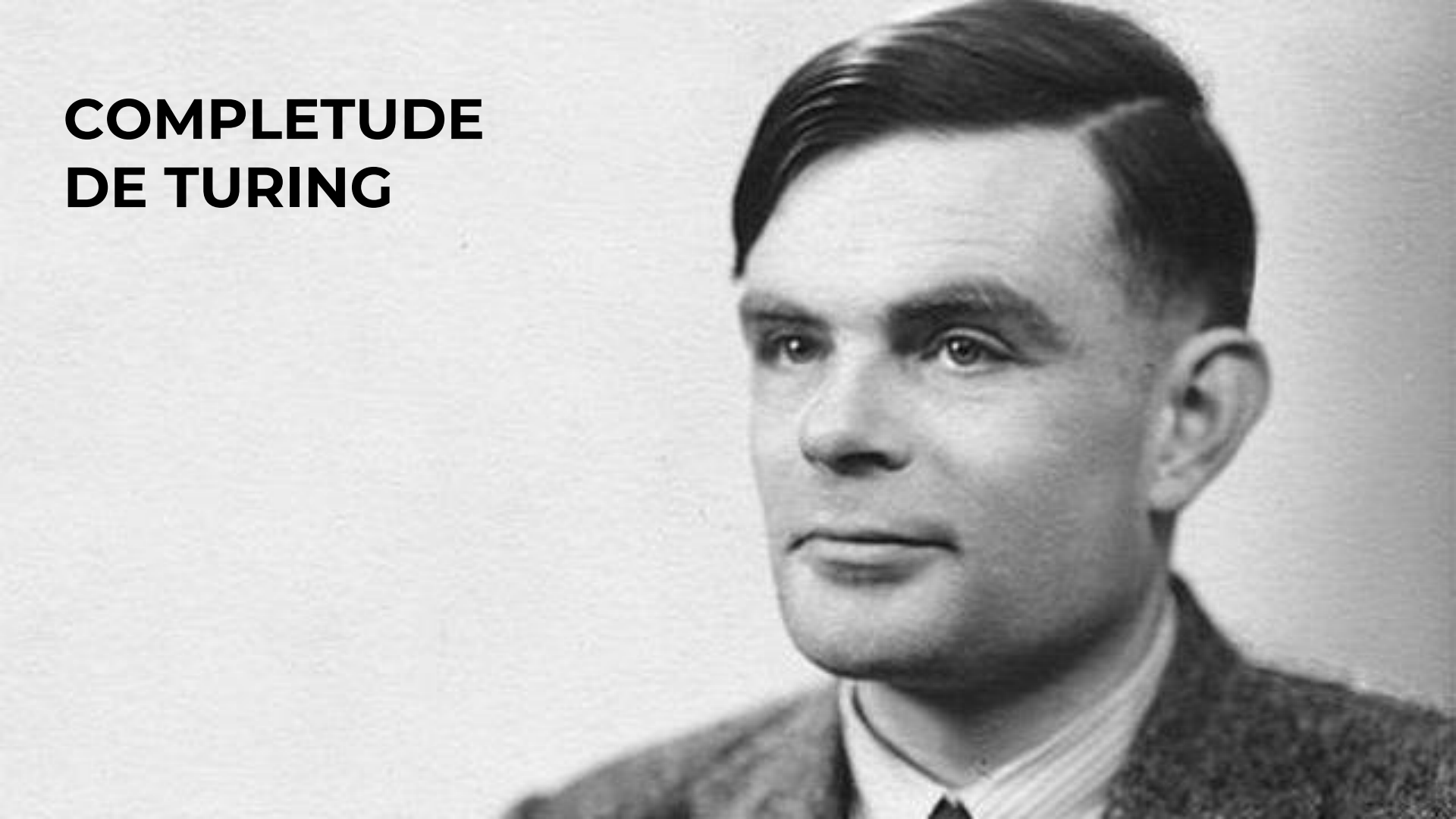
=> Programar qualquer coisa

=> Qualquer um pode usar

Particularidades:

- Endereços
- Transações
- Criptomoeda própria
- Taxas em “Gas”
- Computação / EVM

COMPLETUDE DE TURING



SMART CONTRACTS

Nick Szabo, 1996

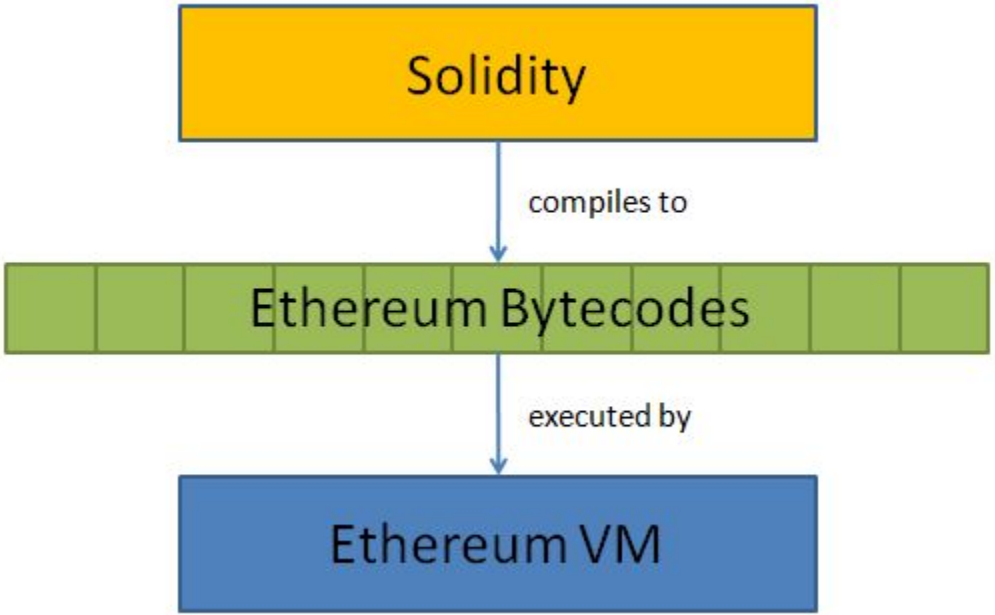
bit.ly/Nick-Szabo-Smart-Contracts

“set of promises, specified in digital form,

including protocols within which

the parties perform on these promises”

EVM: ETHEREUM VIRTUAL MACHINE



LINGUAGENS DE PROGRAMAÇÃO PARA SMART CONTRACTS

Solidity, Vyper (Ethereum)

C/C++ (EOS)

Plutus (Cardano)

RIDE (Waves)

...

Artigos relacionados:

Awesome Smart Contracts

Comparison of Smart Contract Platforms

A deeper look at different Smart Contract Platforms


I C O
Initial
Coin
Offering

Solidity

https://solidity.readthedocs.io/

```
// single line comment


/*
 * multi
 * line
 * comment
 */
```

comments

```
pragma solidity ^0.4.24;
```

pragma

bool

int

uint

bytes

string

address

types

```
struct Candidate {
    uint number;
    string politicalParty;
    string name;
    uint votes;
}
```

structs

```
enum ActionChoices {
    GoLeft,
    GoRight,
    GoStraight,
    SitStill
}
```

enum

```
uint[ ] arr = new uint[](7);

uint[ ] arr = new uint(7);

x = arr[5];

value = arr[i];
```

arrays

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

mapping (uint => uint) index;

mapping (uint => Candidate) aux;
```

mapping

```
DEFAULT:

constructor() public {}
```

```
constructor() {

    totalTotes = 0;

}
```

constructor

```
candidates.push(
    Candidate(
        passportID,
        name,
        Msg.sender,
        0
    )
);
```

creating contracts

```
contract Animal { … }

contract Human is Animal { … }
```

inheritance

```
require( i < array.length );
```

```
require(
    now <= auctionEnd,
    "Auction already ended."
);
```

require

```
address owner;

constructor() public {
    owner = msg.sender;
}

modifier onlyowner(){
    if (msg.sender == owner) {
        _;
    }
}
```

modifiers

```
function endElection() public onlyowner returns (bool) {
    ended = true;
    return true;
}
```

modifiers

(1) block.timestamp

(2) now

alias

```
auctionEnd = block.timestamp + timeFrame;

auctionEnd = now + timeFrame;
```

ok:

```
uint totalVotes;
string name = "Joao";
a = b;
totalVotes += 1;
numberOfCandidates++;
(x, y) = (2, 7);
(x, b, y) = f();
```

not ok:

```
uint a, b;
```

assignments

```
if

else

while

do

for

break

continue

return

? :
```

control structures

Vitalik doesn't want:

switch

goto

control
structures

internal

external

public

private

pure

view

function
special
words

```
function taker (uint a, uint b){

    ...

}
```

function arguments

```
function add(uint a, uint b)
public returns (uint) {
    return a + b;
}
```

function
return

```
function add(uint a, uint b)
public returns (uint c) {
    c = a + b;
}
```

function
return

```
function double(uint a, uint b)
public returns (uint, uint) {
    return (a + a, b + b);
}
```

function
return

```
balance

address(this).balance;


transfer()

_to.transfer();


send()

_to.send();
```

address

```
function sendETHToContract()
public payable returns (bool) {
    return true;
}
```

payable

```
event HighestBidIncreased(address a, uint v);

emit HighestBidIncreased(msg.sender, msg.value);
```

events

msg.sender

msg.value

etc.

msg

Ether units

Wei	1000000000000000000
Szabo	1000000
Finney	1000
Ether	1

```
block.number

block.timestamp (alias: now)

block.blockhash(uint blockNumber)

msg.sender

msg.value

tx.gasprice
```

properties

```
this

selfdestruct(address)

suicide(address) : DEPRECATED
```

contract related


– СПАСИБО!