



# Intro

## Ethereum Blockchain

16 FEBBRAIO 2021

01

Corso Token ERC20



# IN BREVE

## Panoramica

### Blockchain

*Tecnologia facente parte della famiglia delle "Distributed Ledger", punta ad avere una struttura di dati condivisa, immutabile e indipendente da un'autorità centrale.*



### Crittografia a doppia chiave

*Ogni user è in possesso di due chiavi (una pubblica e una privata). Utilizzandola all'interno della blockchain, permette all'utente A di cifrare una qualsiasi informazione da inviare all'utente B tramite la chiave pubblica di B.*



### Ethereum

*Un ambiente virtuale decentralizzato del Web3.0 basato sullo scambio di informazioni peer-to-peer (ogni utente può essere sia user che client).*



### Smart contract

*Codice contenente istruzioni e dati necessari a regolare le interazioni tra utenti sulla blockchain.*



### Solidity

*Codice contenente istruzioni e dati necessari a regolare le interazioni tra utenti sulla blockchain.*



# BLOCKCHAIN

Due utenti hanno uno scambio di informazioni

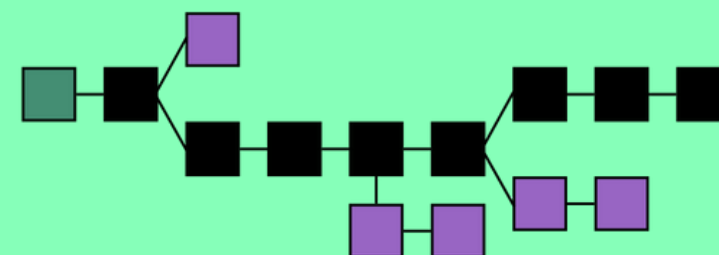
- Devo dare 10€ a un amico entro giorno x
- Devo pagare le rate del mutuo ogni x del mese
- Devo portare a mia madre la farina per la pizza
- Devo vaccinare il mio cane ogni anno a febbraio
- ...



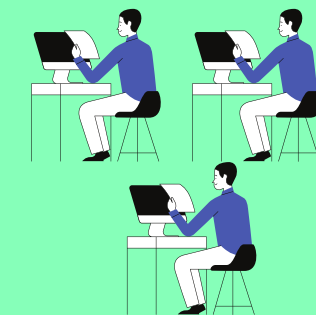
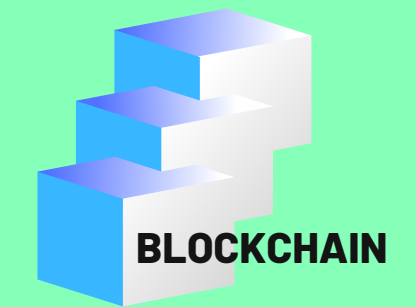
La lista di "interazioni" tra utenti viene collezionata su un "blocco" e salvata su di esso in modo permanente (mining).

Una volta raggiunto il limite di un blocco (circa 500tx - 1MB per la blockchain BTC), si passa al successivo.

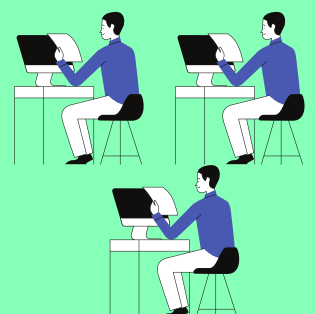
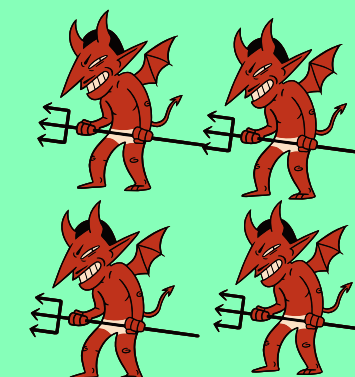
Ogni blocco contiene informazioni sul precedente (univocità temporale), una lista di transazioni, un nonce e un timestamp.



Solo uno di tutti i possibili rami "sopravvive" grazie a diversi meccanismi. Bitcoin e Ethereum ad esempio utilizzano il PoW (Proof of Work). Ethereum 2.0 utilizzerà il PoS (Proof of Stake).



Per far in modo che le informazioni siano scritte su ogni blocco in modo permanente, una serie di calcoli matematici è svolto dai "nodi" che mettono il proprio hardware a disposizione per "minare" un blocco in cambio di ricompense (BTC ad esempio).



# CRITTOGRAFIA A DOPPIA CHIAVE



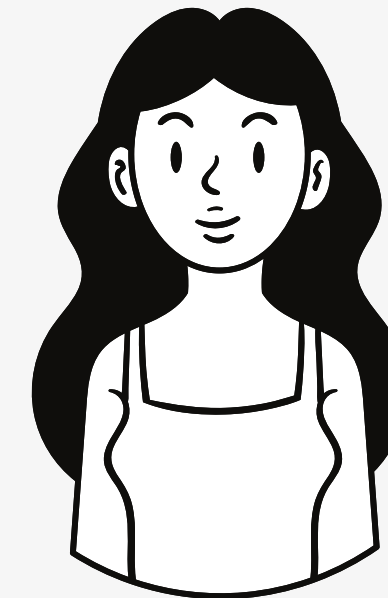
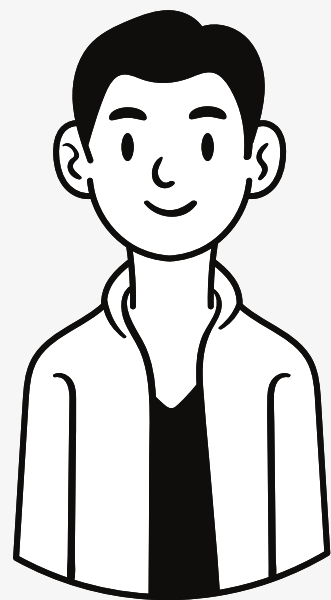
A ha una chiave **privata**  
"AAAA"



Da essa genera una chiave **pubblica** tramite ECC  
"A-PUBLIC"



A scrive un messaggio e lo cripta con la chiave pubblica di B "**B-PUBLIC**"



B riceve il messaggio di A grazie alla sua chiave pubblica  
"**B-PUBLIC**"



B decripta il messaggio grazie alla sua chiave privata "BBBB"



Una funzione **HASH** deve essere:

Univoca (non è possibile avere output uguali per input diversi)

Deterministica (lo stesso input produrrà sempre lo stesso output)

Semplice da calcolare (in un senso)

Impossibile ricavarne l'inversa (se non trovando tutti i possibili hash e confrontando con l'originale)

# ETHEREUM



EVM



Ethereum è un universo virtuale creato su un singolo computer (Ethereum Virtual Machine) sul cui stato devono essere d'accordo tutti i "nodi" connessi a esso. Può intendersi quindi come la più ampia applicazione del concetto di blockchain.



Ogni utente di questo network, può richiedere a questa macchina virtuale l'esecuzione di particolari operazioni. Questa procedura ha luogo tramite "transazioni".



Ogni richiesta deve essere validata, approvata ed eseguita da tutti i partecipanti al network (nodi). Quando il consenso comune è raggiunto, lo stato della macchina virtuale viene modificato e una copia del nuovo stato è accessibile a tutti gli utenti.



## Come si richiede una transazione?



Tramite Smart Contracts

Codice salvato direttamente sulla macchina virtuale con il quale chiunque può interagire.



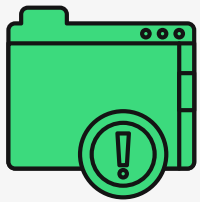
## Cosa ci guadagnano i nodi nello spendere risorse e tempo per la validazione?



I nodi vengono ricompensati in ETH, la moneta nativa del mondo Ethereum.

CURIOSITA': ETH è stata creata prima dell'arrivo dello standard ERC20. Per questo motivo, per essere utilizzata su exchanges, deve essere "trasformata" in WETH (Wrapped Ethereum). Lo stesso succede per i BTC (Wrapped Bitcoin) e così via, tramite il processo che viene detto Wrapping (incartare).

# SMART CONTRACTS



## Distributore virtuale

Possono essere immaginati come dei distributori automatici virtuali. Al loro interno ci sono una serie di istruzioni interagendo con le quali si generano degli output.



## Nessun permesso richiesto

Chiunque, appresa la sintassi di Solidity (linguaggio in cui sono scritti) può creare degli smart contract sulla macchina virtuale Ethereum.



## Limiti

Essendo creati sulla macchina virtuale, non possono di per se interfacciarsi col mondo virtuale esterno (il web come lo intendiamo). Non possono infatti lanciare richieste http verso l'esterno. Per ovviare al problema, sono stati introdotti gli oracoli che permettono di ascoltare determinate informazioni esterne tramite APIs.

```
pragma solidity 0.6.11;
contract VendingMachine {

    address public owner;
    mapping (address => uint) public cupcakeBalances;

    constructor() public {
        owner = msg.sender;
        cupcakeBalances[address(this)] = 100;
    }

    function refill(uint amount) public {
        require(msg.sender == owner, "Solo il proprietario può ricaricare.");
        cupcakeBalances[address(this)] += amount;
    }

    function purchase(uint amount) public payable {
        require(msg.value >= amount * 1 ether, "Devi pagare 1 ETH per cupcake.");
        require(cupcakeBalances[address(this)] >= amount, "Prodotto terminato.");
        cupcakeBalances[address(this)] -= amount;
        cupcakeBalances[msg.sender] += amount;
    }
}
```

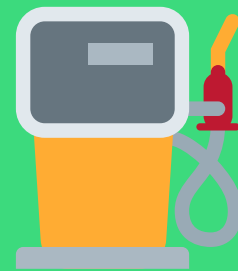
# CONCETTI IMPORTANTI PER LO SVILUPPO DI SMART CONTRACTS



## Transazione

Istruzione crittografata da parte di un account che viene inviata sulla blockchain.

- Contiene delle informazioni e dei dati
- E' firmata tramite la chiave privata del mittente
- Ha un limite di gas (massima quantità di gas che la transazione può spendere)
- Ha delle tasse da pagare (dipendenti dal prezzo per unità di gas)

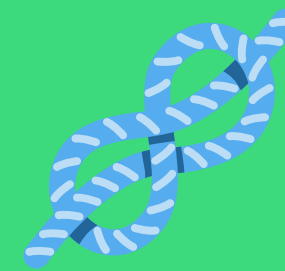


## Gas

E' il "carburante" che deve essere bruciato per completare delle transizioni sulla EVM.

Come nel mondo reale, il gas ha un prezzo (in ETH). In particolare l'unità di misura del prezzo del gas è comunemente il gwei ( $10^{-9}$ ETH).

- Più gas di brucia, più veloce sarà la transazione
- Il prezzo del gas varia in base a diversi fattori e può essere controllato su <https://ethgasstation.info/>

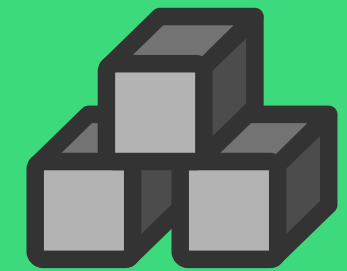


## Nodo

Sono dei pezzi di codice, salvati su un hardware, che contribuiscono alla sicurezza e all'accuratezza della blockchain validando le diverse transazione su ogni blocco.

Posso essere:

- Full (conservano una copia dell'intera blockchain)
- Light (conservano una parte della blockchain e richiedono informazione sul resto)
- Archive (archiviano dati sui blocchi passati)



## Blocco

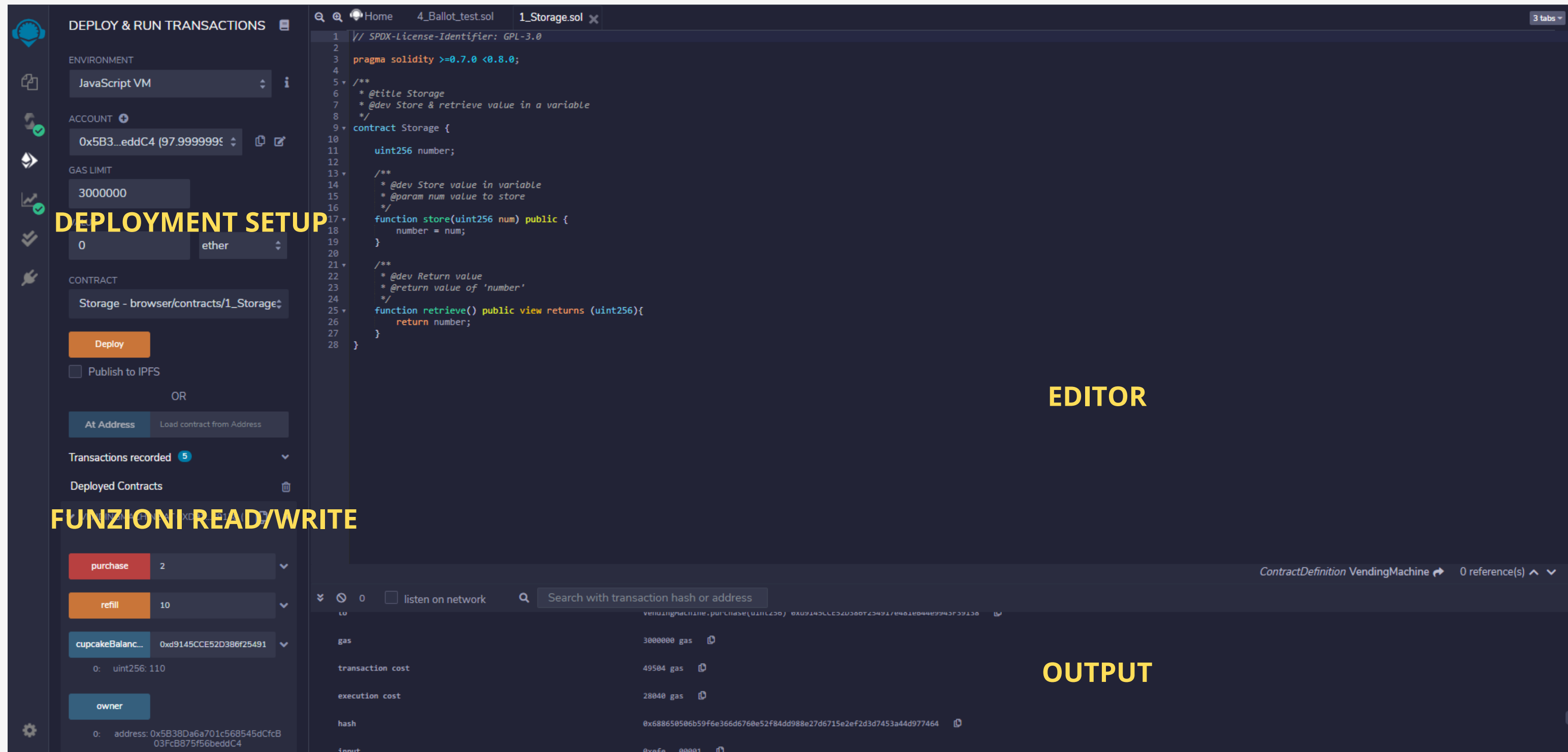
Un insieme di transazioni validate da un determinato nodo. Una volta completo, viene agganciato in modo univoco al blocco successivo.

Ogni blocco contiene:

- Timestamp
- Numero blocco
- Difficoltà
- Hash
- Hash padre
- Lista transazioni
- Stato
- Nonce

# SOLIDITY & REMIX

Linguaggio di programmazione orientato alla definizione di Smart Contracts su Ethereum (influenzato fortemente da Python e JavaScript).



The screenshot displays the Remix IDE interface, which is used for developing and deploying smart contracts on the Ethereum blockchain. The interface is divided into three main sections:

- DEPLOYMENT SETUP:** Located on the left side, this panel contains settings for the deployment environment. It includes a dropdown for the environment (set to JavaScript VM), an account selection (0x5B3...eddC4), a gas limit (3000000), and a contract selection (Storage - browser/contracts/1\_Storage). A "Deploy" button is prominently displayed.
- EDITOR:** The central area of the IDE, which contains the Solidity code for the smart contract. The code defines a contract named "Storage" with a "store" function and a "retrieve" function.
- OUTPUT:** Located at the bottom right, this panel shows the results of the deployment. It includes a table with columns for "gas", "transaction cost", "execution cost", "hash", and "input". The "hash" column displays the contract's address: 0x688658506b59f6e366d6760e52f84dd988e27d6715e2ef2d3d7453a44d977464.

Additional labels are overlaid on the image:

- FUNZIONI READ/WRITE:** A yellow label pointing to the "retrieve" function in the code editor.
- EDITOR:** A yellow label pointing to the central code editor area.
- OUTPUT:** A yellow label pointing to the bottom right output panel.



# UNA TRANSAZIONE A CASO (AAVE CONTRACT)

https://etherscan.io/tx/0xea8b5f76e078c41870d944def8b4fce49de26a938893ce27f2336ba5e833582c

Transaction Details

Buy

Exchange

Earn

Gaming

OverviewInternal TxnsLogs (5)StateComments

Transaction Hash:

0xea8b5f76e078c41870d944def8b4fce49de26a938893ce27f2336ba5e833582c

Status:

Success

Block:

118572813 Block Confirmations

Timestamp:

32 secs ago (Feb-14-2021 09:32:14 PM +UTC) | Confirmed within 4 secs

From:

0x23c6fd897002a8c7d311a12857c2d5bf1f2d86de

Interacted With (To):

Contract 0x7fc66500c84a76ad7e9c93437bfc5ac33e2ddae9 (Aave: AAVE Token)

Tokens Transferred:

From 0x23c6fd897002a8c... To 0x738a1ea9df66af8... For 10 (\$4,822.63) Aave Token (AAVE)

Value:

0 Ether (\$0.00)

Transaction Fee:

0.029162475 Ether (\$53.21)

Gas Price:

0.000000155 Ether (155 Gwei)

Click to see More

Private Note:

To access the Private Note feature, you must be Logged In

Input Data:

#	Name	Type	Data
0	_to	address	0x738a1ea9DF66AF8761292d6f4346dC2b8f1881F1
1	_value	uint256	1000000000000000000

Switch Back