

Blockchain aplicado a contratos inteligentes

An abstract graphic consisting of several overlapping, elongated, and pointed shapes in shades of teal and lime green, positioned behind the text.

INTRODUCCIÓN

¿Qué son. . .



Blockchain



Contrato Inteligente

Contexto Histórico.

SLIDE 3

1991

Introducción.

Stuart Haber y W. Scott Stornetta introdujeron una solución computacionalmente práctica.

2004

Vence la Patente.

Hal Finney (Harold Thomas Finney II) introdujo un sistema llamado RPoW.

2008

Efectivo Electrónico.

Satoshi Nakamoto publicó un libro blanco que introdujo un sistema de efectivo electrónico descentralizado entre pares (llamado Bitcoin).

2009

Bitcoin.

Bitcoin nació cuando el primer bloque de bitcoin fue minado por Satoshi Nakamoto.

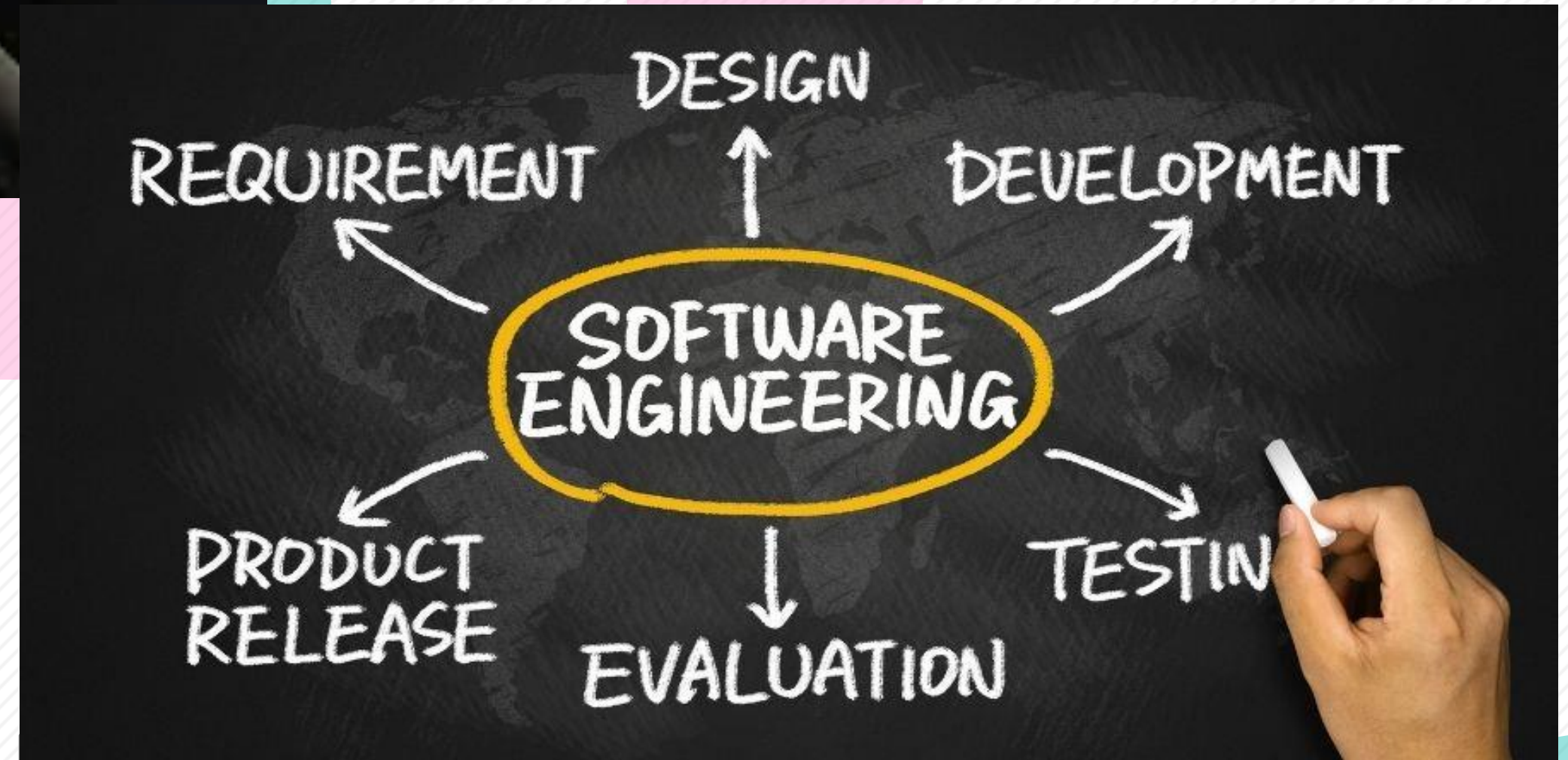
2013

Ethereum

Vitalik comenzó el desarrollo de una nueva plataforma de computación distribuida basada en blockchain, Ethereum

RELEVANCIA CON LA INGENIERÍA EN COMPUTACIÓN





RELACIÓN CON EL CÓMPUTO MÓVIL

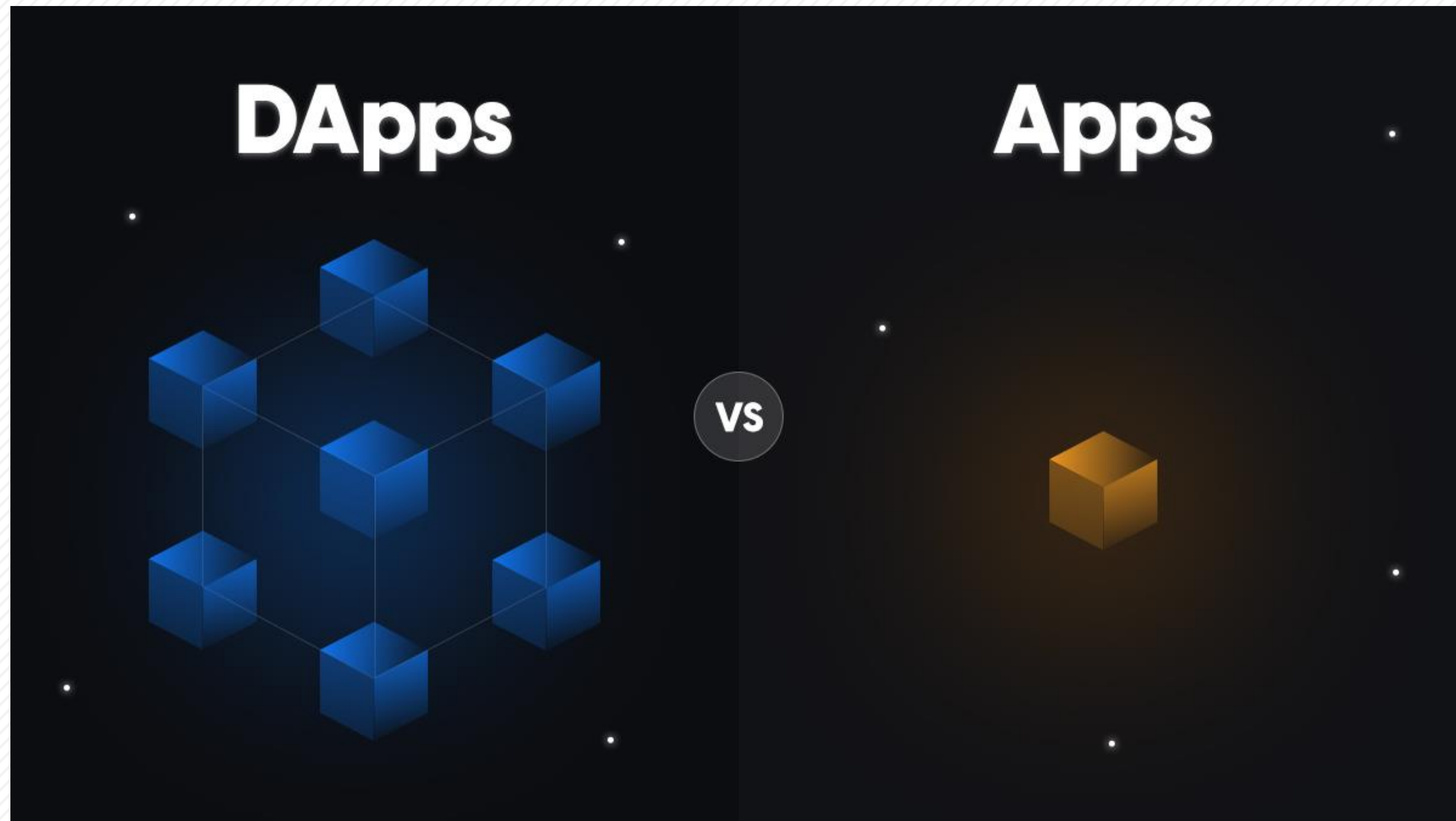


Cómputo Móvil

+



Contrato Inteligente



DApps

Programas informáticos que se ejecutan en una red descentralizada

Ethereum

Contratos inteligentes




```
// Indica la versión de Solidity que se utilizará para compilar el contrato
pragma solidity ^0.8.0;
```

```
// Declaración del contrato
contract Token {
```

```
    // Variables públicas del contrato
    string public name; // Nombre del token
    string public symbol; // Símbolo del token
    uint8 public decimals; // Número de decimales del token
    uint256 public totalSupply; // Suministro total del token
```

```
    // Mappings (tablas hash) para almacenar balances y aprobaciones
    mapping(address => uint256) public balanceOf; // Balances de los usuarios
    mapping(address => mapping(address => uint256)) public allowance; // Aprobaciones de transferencia de terceros
```

```
    // Eventos que se emiten al realizar transferencias o aprobaciones
    event Transfer(address indexed from, address indexed to, uint256 value); // Evento de transferencia
    event Approval(address indexed owner, address indexed spender, uint256 value); // Evento de aprobación
```

```
    // Constructor del contrato
    constructor(string memory _name, string memory _symbol, uint8 _decimals, uint256 _totalSupply) {
        name = _name; // Asigna el nombre del token
        symbol = _symbol; // Asigna el símbolo del token
        decimals = _decimals; // Asigna el número de decimales del token
        totalSupply = _totalSupply; // Asigna el suministro total del token
        balanceOf[msg.sender] = totalSupply; // Asigna todo el suministro inicial al creador del contrato
        emit Transfer(address(0), msg.sender, totalSupply); // Emite un evento de transferencia que indica que se ha
        transferido todo el suministro inicial al creador del contrato
    }
```

```
    // Función para transferir tokens a otro usuario
    function transfer(address _to, uint256 _value) public returns (bool success) {
        require(balanceOf[msg.sender] >= _value); // Verifica que el remitente tenga suficientes tokens para la transferencia
        balanceOf[msg.sender] -= _value; // Reduce el balance del remitente
        balanceOf[_to] += _value; // Aumenta el balance del destinatario
        emit Transfer(msg.sender, _to, _value); // Emite evento de transferencia que indica que se ha transferido un cierto
        número de tokens de un remitente a un destinatario
        return true; // Indica que la transferencia se ha realizado con éxito
    }
```

```
    // Función para aprobar la transferencia de tokens a un tercero
    function approve(address _spender, uint256 _value) public returns (bool success) {
        allowance[msg.sender][_spender] = _value; // Asigna una cantidad de tokens que el tercero está autorizado
        a transferir desde la cuenta del remitente
        emit Approval(msg.sender, _spender, _value); // Emite un evento de aprobación que indica que se ha aprobado una
        cierta cantidad de tokens para transferirse desde una cuenta del remitente a una cuenta del tercero
        return true; // Indica que la aprobación se ha realizado con éxito
    }
```

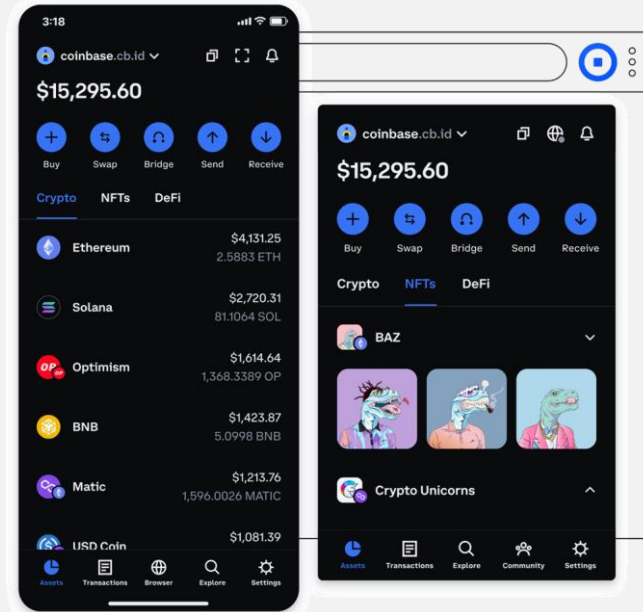
Contrato inteligente para un token ERC- 20 básico

Apps y contratos inteligentes



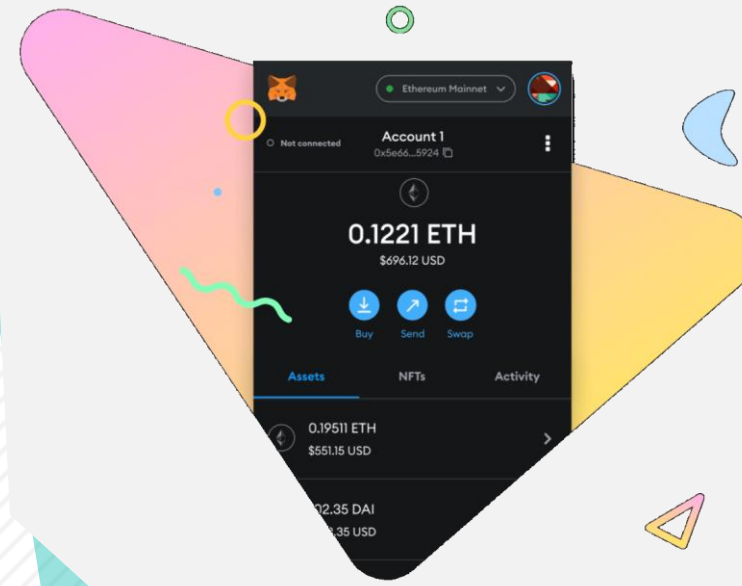
Trust Wallet

Es una billetera de criptomonedas que permite a los usuarios almacenar, enviar y recibir criptomonedas, incluyendo tokens ERC-20



Coinbase Wallet

Billetera de criptomonedas que también permite a los usuarios almacenar, enviar y recibir criptomonedas y tokens ERC-20



MetaMask

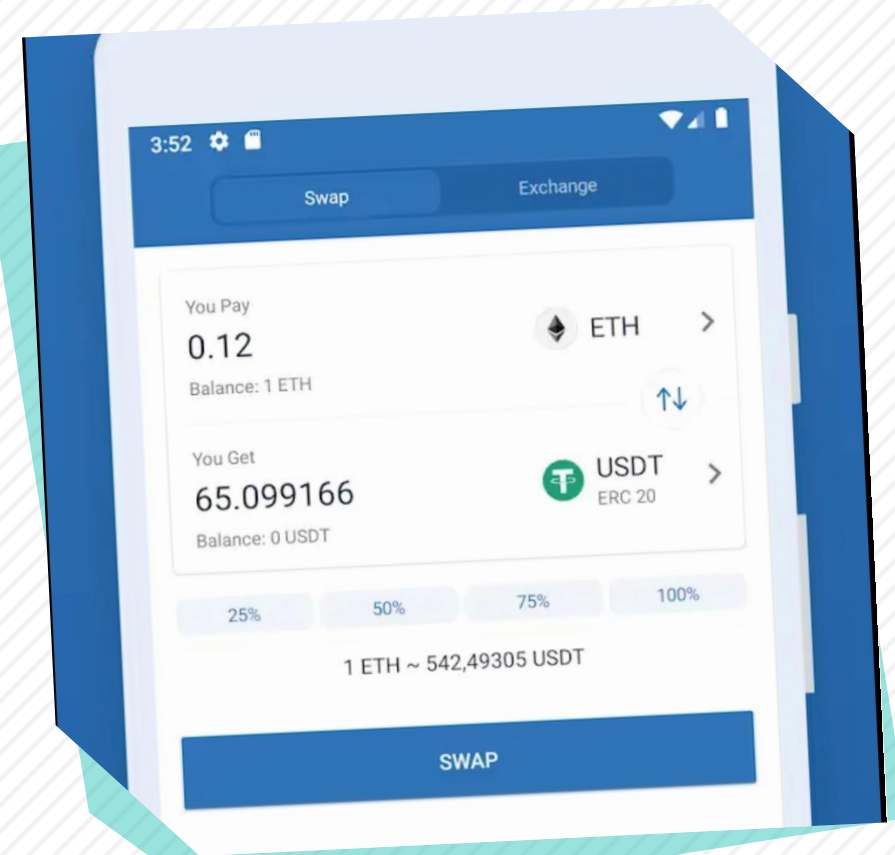
Es una extensión de navegador web que permite a los usuarios interactuar con la blockchain de Ethereum y sus contratos inteligentes.



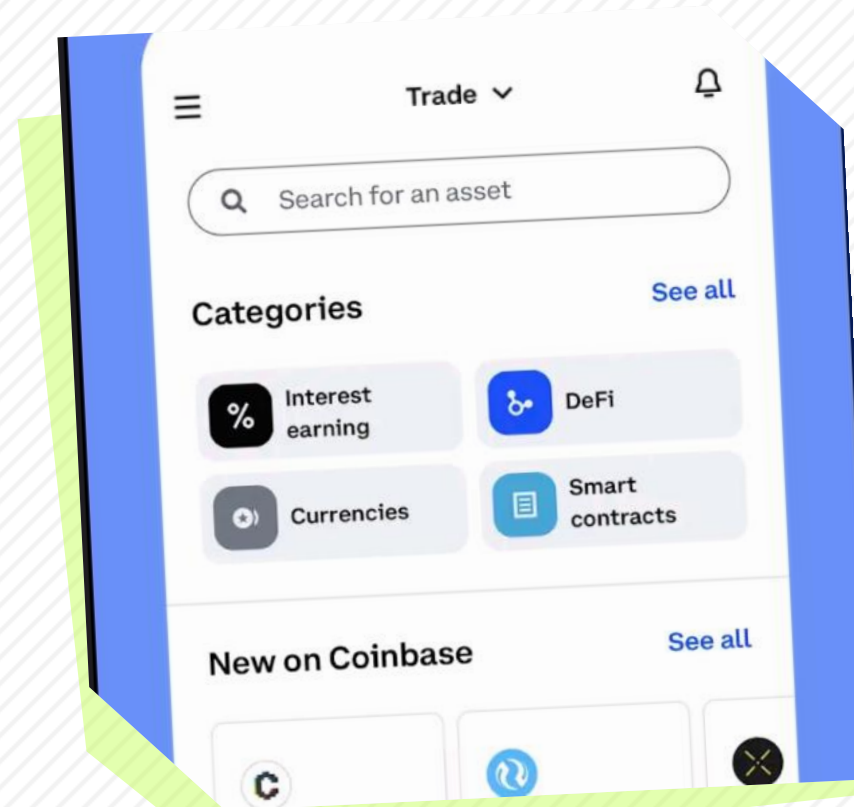
MyEtherWallet

Es una billetera de criptomonedas que permite manipular y gestionar criptomonedas, al igual que tokens ERC-20.

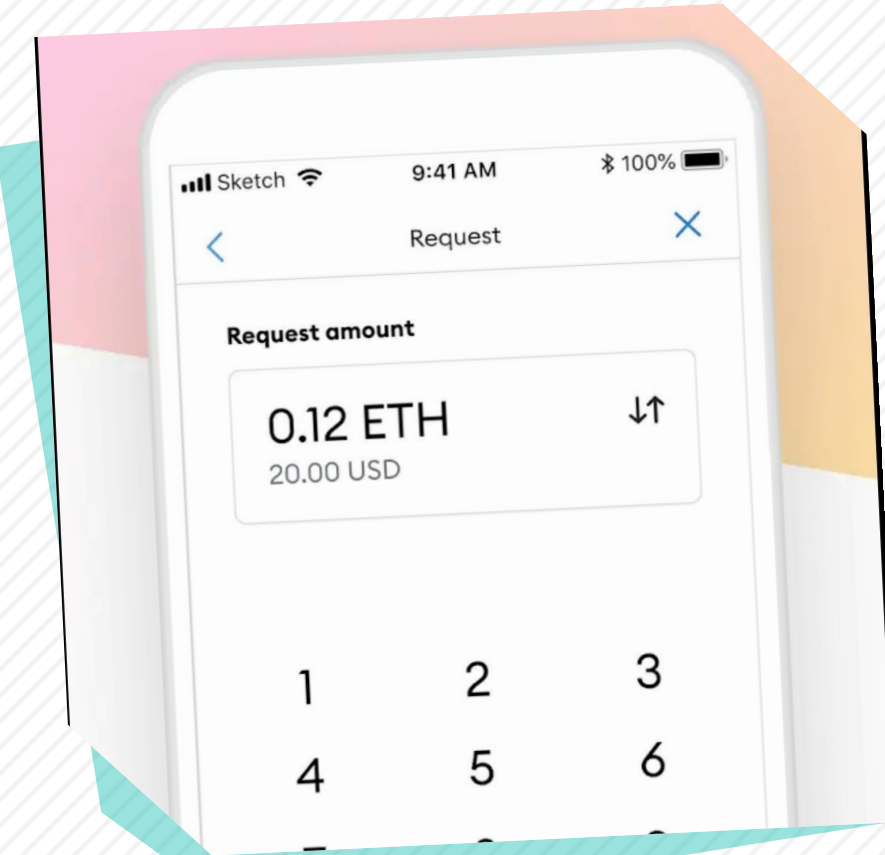
Apps y contratos inteligentes



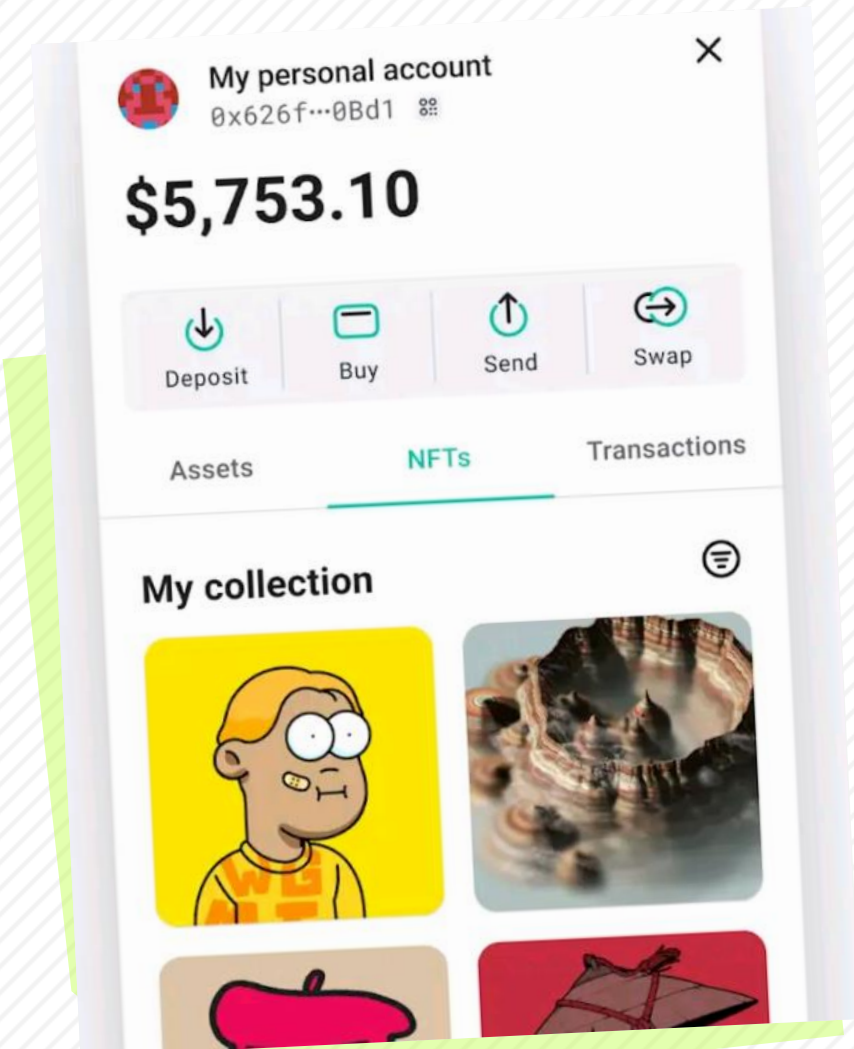
Trust Wallet



Coinbase Wallet



MetaMask

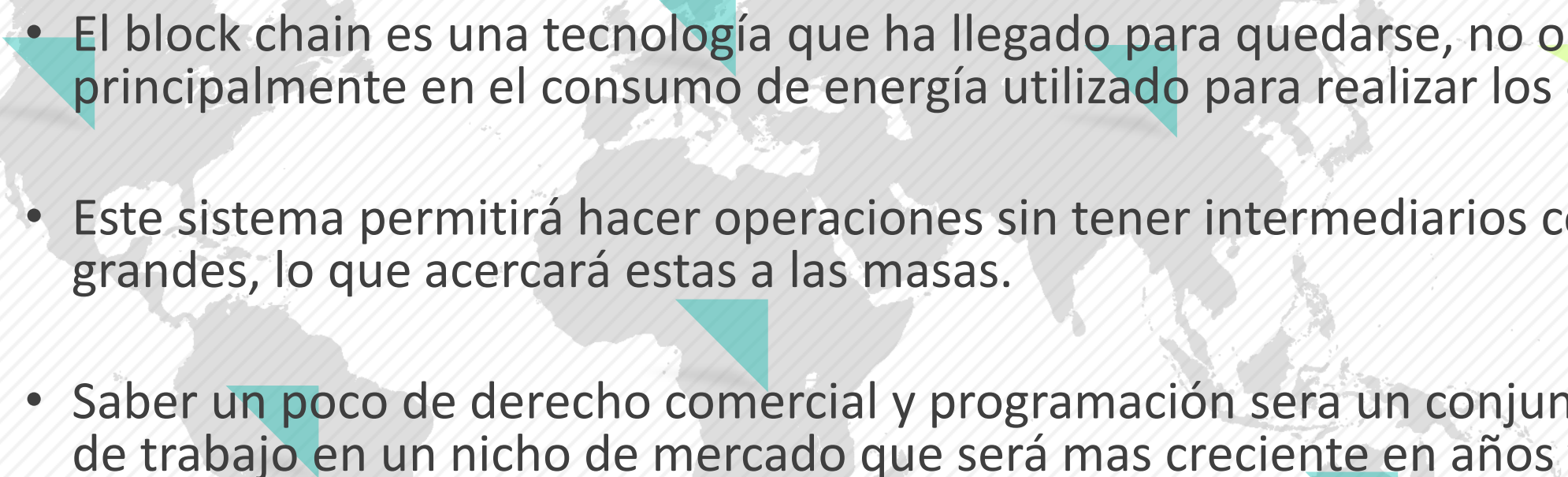


MyEtherWallet

¿Qué pasará a futuro?

- 
- Se tendrán sistemas descentralizados económicos en caso de seguir prosperando los sistemas de blockchain.
 - Operaciones bursátiles se acelerarán en caso de realizarse transacciones automáticas.
 - Mas personas podrán acceder a servicios de este giro debido a la ampliación de aplicaciones en dispositivos móviles.
 - La trazabilidad de las operaciones sera mas rastreable, por lo que se dificultará hacer fraudes.

Conclusiones

- 
- El block chain es una tecnología que ha llegado para quedarse, no obstante tiene puntos a mejorar principalmente en el consumo de energía utilizado para realizar los cálculos correspondientes.
 - Este sistema permitirá hacer operaciones sin tener intermediarios como el sistema Swift o bien bancos mas grandes, lo que acercará estas a las masas.
 - Saber un poco de derecho comercial y programación sera un conjunto de herramientas que permitirá hacerse de trabajo en un nicho de mercado que será mas creciente en años venideros.