

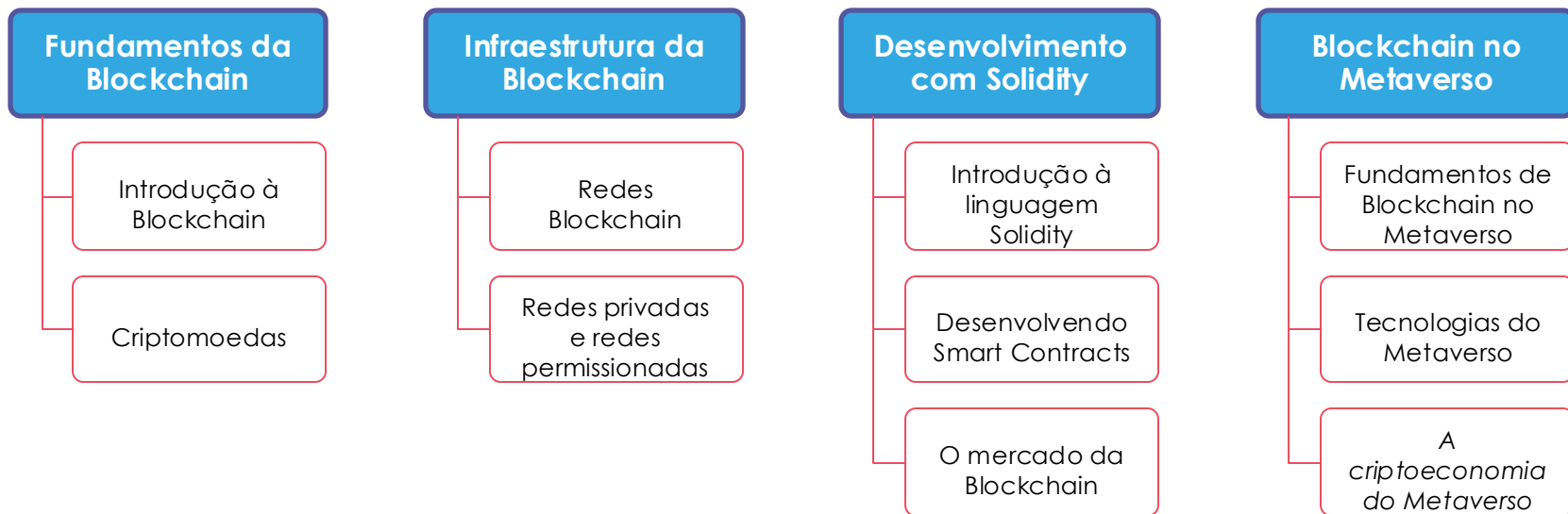
# Espaço da webcam

**Em todos os slides, evite escrever ou usar imagens que possam ocupar a área mostrada ao lado, pois ela representa o espaço reservado para a webcam.**

**WEBCAM**

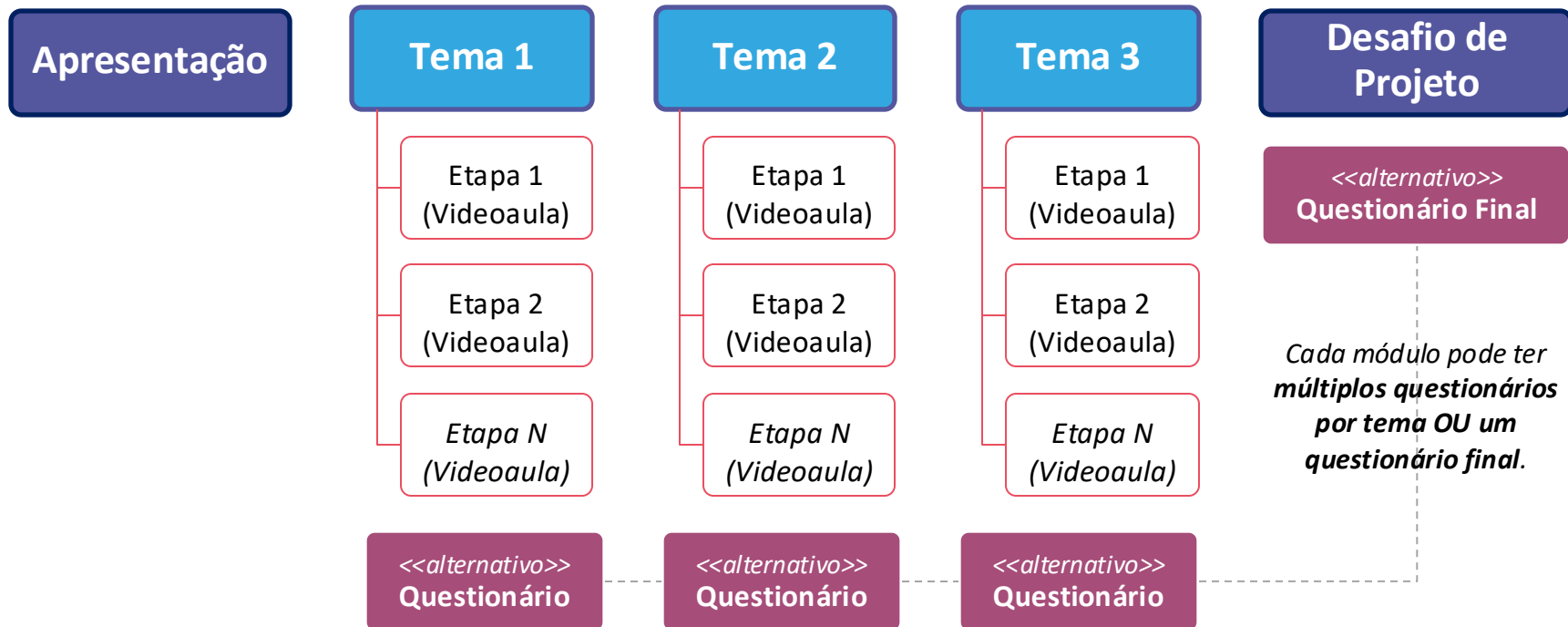
# Estrutura das Trilhas

## COMPETÊNCIA



Nesse contexto, "**Módulos**" são "**Subcompetências**".

# Estrutura dos Módulos



Cada **Etapa** deve ter, idealmente, em torno de **15 minutos** (isso pode variar, principalmente em videoaulas práticas). Analogamente, recomendamos que cada **Tema** tenha entre **2 e 4 horas**.

# O mercado de blockchain e criptomoedas

**Cassiano Peres**

DIO Tech Education Analyst

# Sobre Mim

- Analista e desenvolvedor de sistemas
- Empreendedor
- Apaixonado pela liberdade
- Fã de criptomoedas e da economia descentralizada

 cassiano-dio

 peres-cassiano

# Objetivo Geral

Neste curso vamos abordar o desenvolvimento de Smart Contracts.

# Pré-requisitos

- Conhecimento básico em JavaScript, C++ ou Python;
- Noções de redes de computadores;
- Conhecimento fundamental de criptografia e algoritmos.

# Percurso

## Etapa 1

Padrões de contratos inteligentes

## Etapa 2

O Padrão ERC-20

## Etapa 3

Tokens ERC-20



# Percurso

## Etapa 4

Criando o seu primeiro Token ERC-20

## Etapa 5

O padrão ERC721

## Etapa 6

Tokens ERC-721

# Percurso

## Etapa 7

O protocolo IPFS

## Etapa 1

# Padrões de contratos inteligentes

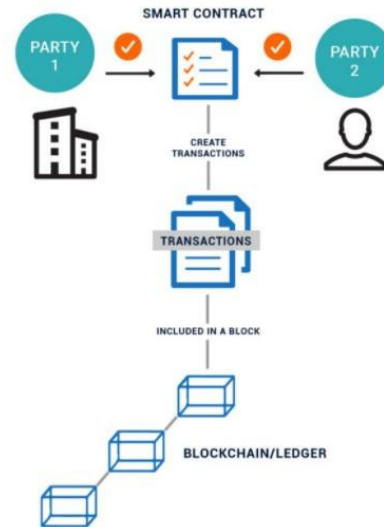


# Smart contracts

Smart contracts (contratos inteligentes) são acordos essencialmente **automatizados** entre o criador do contrato e o destinatário, sendo registrados na blockchain, tornando-os imutáveis e irreversíveis.

# Smart contracts

BLOCKCHAIN AND SMART CONTRACTS - FLOW DIAGRAM



# Smart Contracts

Pode ser utilizada para desenvolver contratos como votações, crowdfunding, rastreabilidade de ativos, NFT's, entre outros

# Smart contracts

Para ser considerado um smart contract, algumas regras devem ser seguidas:



# Padrões

Os padrões de tokens são definidos pela ERC – *Ethereum Request for Comment* que define a convenção para os smart contracts, com regras para interação com os contratos.

# Padrões

Standard Name	Created Date	Use Cases
ERC-20	2015-11-19	Fungible token standard that provides basic functionality to transfer tokens, as well as allow tokens to be approved.
ERC-721	2018-01-24	Non-Fungible Token standard.
ERC-777	2017-11-20	Standard that defines all the functions required to send tokens on behalf of another address, contract or regular account.
ERC-1155	2018-06-17	A standard for contracts that manage multiple token types.

# Padrões

- Padronização da programação;
- Simplificação do desenvolvimento;
- Suporte para múltiplas linguagens;

# Padrões

- Tokens menos complexos;
- Segurança;
- Menores riscos de incompatibilidade.

# Conclusão

Vimos nesta etapa uma introdução sobre os padrões de contratos para o Solidity.

## Etapa 2

# O padrão ERC-20

# Introdução

Nesta etapa vamos abordar o padrão ERC-20 para tokens baseados em Ethereum.



# Introdução

- **ERC** (Ethereum Request for Comments) é um protocolo oficial para fazer sugestões para melhorar a rede Ethereum
- **20** é o número de identificação único da proposta



# O padrão ERC-20

O padrão ERC-20 define um conjunto de regras que devem ser atendidas para que um token seja aceito e capaz de interagir com outros tokens na rede.

# O padrão ERC-20

Um token ERC-20 deve ser obrigatoriamente:

- Fungível;
- Transferível;
- Base monetária fixada.

# O padrão ERC-20

O padrão ERC-20 possui ***Getters***, **Funções** e **Eventos** que definem o comportamento do token.

# Getters

```
function totalSupply() external view returns (uint256)
```

```
//Retorna a quantidade de tokens existentes.
```

```
function balanceOf(address account) external view returns (uint256);
```

```
//Retorna a quantidade de tokens pertencentes a um endereço
```

# Getters

```
function allowance(address owner, address spender) external view returns (uint256);
```

// O padrão ERC-20 permite que um endereço autorize outro endereço a recuperar tokens dele.

# Funções

```
function transfer(address recipient, uint256 amount) external returns (bool);
```

```
// Transferência de tokens entre endereços
```

```
function approve(address spender, uint256 amount) external returns (bool);
```

```
//Emite o evento de aprovação de uma transferência, retornando se foi ou não aprovada.
```

# Funções

```
function transferFrom(address sender, address recipient, uint256 amount) external returns  
(bool);
```

```
//Move uma quantidade de tokens entre endereços e deduz do saldo do emissor. Retorna um  
evento Transfer
```

# Eventos

```
event Transfer(address indexed from, address indexed to, uint256 value);
```

//Evento emitido quando a quantidade de tokens é enviada de um endereço para outro

```
event Approval(address indexed owner, address indexed spender, uint256 value);
```

//Evento emitido quando uma quantidade de tokens é aprovada pelo dono do contrato para ser enviado por um *spender*.



# Campos do token ERC-20

Um token ERC-20 possui alguns campos opcionais:

- **Token Symbol:** símbolo do token (ETH);
- **Decimals:** casas decimais para fracionamento do token;
- **Token Name:** nome do token.

# Documentação

A documentação sobre padrões na rede ethereum pode ser encontrada no site <https://ethereum.org>

## Etapa 3

# Tokens ERC-20

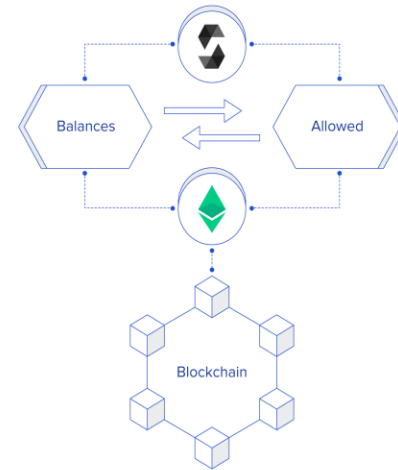
# Introdução

Os tokens ERC-20 são contratos inteligentes (smart contracts) executados na blockchain da rede Ethereum.



# Introdução

Esses contratos seguem um conjunto de regras a ele especificadas, com o intuito de executar uma determinada tarefa.



# Tokens ERC-20

Diferentemente do Ether (ETH), criptomoeda nativa da Ethereum, esses tokens existem apenas dentro de um contrato inteligente, que define as regras para seu funcionamento

# Token ERC-20

Para enviar e receber tokens ERC-20 na rede Ethereum, mesmo que um usuário não esteja enviando Ether, ele precisa possuir uma quantia de ETH para realizar a transação.

# Token ERC-20

Isso porque é preciso pagar uma taxa de transação necessária para incluir sua transferência em um bloco da rede.

Essa taxa na Ethereum é chamada de "*gas*".





# Tokens ERC-20

Embora a criptomoeda Ether possa ser minerada, os tokens do baseados em Ethereum **não podem**.

# Tokens ERC-20

Ao criar um contrato, é determinado o fornecimento total de unidades do token (*total supply*) e o período de distribuição.

Quando um novo token é criado, ele é *cunhado*, do inglês *minted*.

# Tokens ERC-20 no mercado

- Theter (USDT)
- Chainlink (LINK)
- ApeCoin (APE)
- Chiliz (CHZ)
- Axie Infinity Shards (AXS)



# Conclusão

Nesta etapa conhecemos um pouco mais a respeito dos tokens ERC-20, além de exemplos de aplicação no mercado.

## Etapa 4

# Criando um token ERC-20

# Introdução

Nesta etapa vamos desenvolver nosso projeto de um token no padrão ERC-20.

# O projeto

Vamos implementar as funções e definir os campos de informações que caracterizam um token ERC-20.

# O projeto

Recursos utilizados:

- Remix IDE
- Ganache
- Solidity



# Código

O código deste projeto estará disponível no GitHub.

## Etapa 5

# O padrão ERC-721

# Introdução

Nesta etapa vamos abordar o padrão de token ERC-721, utilizado para o desenvolvimento de NFT's, *Non-fungible tokens*.



# NFT

É um padrão utilizado para representar a posse de NFT's, representando a **unicidade** de um token.



# NFT

Um token não fungível é a representação de um ativo físico **único, insubstituível**, como um quadro de Da Vinci, Picasso ou um violino Stradivarius.

# NFT

Um token fungível **pode ser substituído** em termo de valores, como por exemplo o Bitcoin, onde podemos substituir 1 btc por dez depósitos de 0.1 btc, ou dinheiro convencional onde cinco notas de R\$ 10 valem o mesmo que uma nota de R\$ 50.

# NFT

Um token fungível **pode ser substituído** em termo de valores, como por exemplo o Bitcoin, onde podemos substituir 1 btc por dez depósitos de 0.1 btc, ou dinheiro convencional onde cinco notas de R\$ 10 valem o mesmo que uma nota de R\$ 50.

# ERC-721

Um token baseado no ERC-721 tem o seu valor baseado na sua **raridade**, sendo muito utilizado para representar itens **coleccionáveis**.





# ERC-721

Da mesma forma que um token ERC-20, os tokens ERC-721 deve seguir um padrão de métodos e atributos que o definem.

# ERC-20 x ERC-721

- Tokens ERC-721 são tokens NFT (token não fungível, ou seja, único e insubstituível);
- Tokens ERC-20 são **divisíveis** enquanto os ERC-721 não o são.

# OpenZeppelin

O OpenZeppelin é um framework que disponibiliza um conjunto de bibliotecas com padrões de contratos para o desenvolvimento seguro de smart contracts.



# OpenZeppelin

A documentação do OpenZeppelin pode ser acessado pelo link <https://www.openzeppelin.com/>



# Conclusão

Nesta etapa conhecemos os conceitos de NFT, características e funcionalidades.

## Etapa 6

# Tokens ERC-721

# Introdução

Nesta etapa vamos falar de exemplos de tokens ERC-721 e as regras que devem seguir.

# Características

- Cada token ERC-721 possui um campo de nome para identificar o token para aplicações ou contratos externos;
- Cada token possui funções específicas para definir qual o dono e como transferir a propriedade do token;
- O padrão ERC-721 possui uma função de *tokenOwnerByIndex* para rastrear um token por um ID único.



# Características

Um token ERC-721 compartilha alguns métodos com o padrão ERC-20, como definição de nome, base monetária, e saldos.

Além destas, possui funções relacionadas à transferência e propriedade dos tokens.

# Características

Um token ERC-721 compartilha alguns métodos com o padrão ERC-20, como definição de nome, base monetária, e saldos.

Além destas, possui funções relacionadas à transferência e propriedade dos tokens.

# Benefícios

- Fácil movimentação entre contas e troca de NFT's por outras criptomoedas;
- Definição do suprimento total de um grupo de NFT's disponíveis;
- Rastreabilidade da propriedade do token.

# Exemplos

- Axie
- Avastar
- VNFT
- Swap
- Sorare



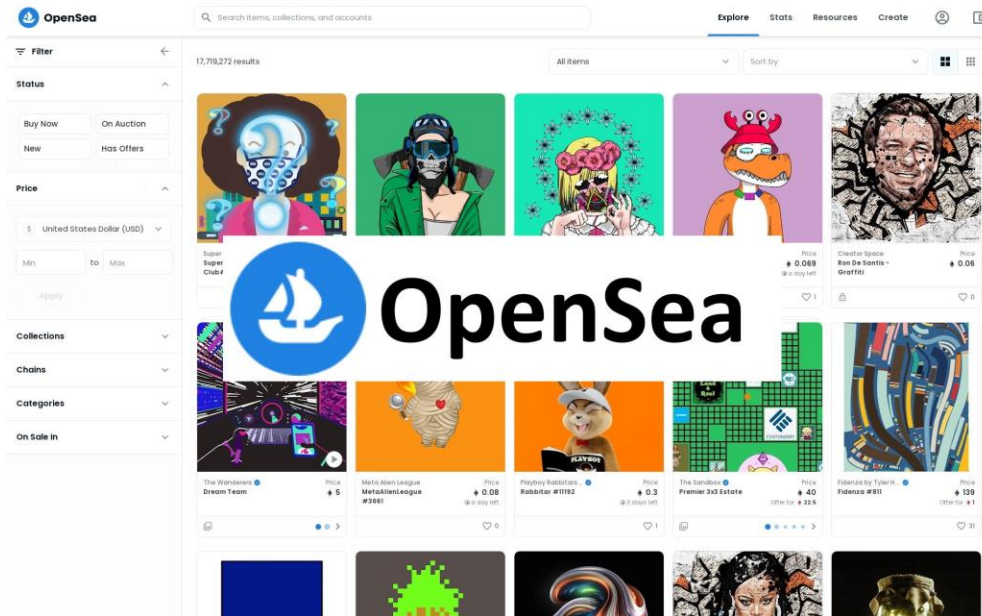
# OpenSea

O OpenSea é o maior marketplace de NFT's do mundo atualmente.

Link para o OpenSea: <https://opensea.io/>



# OpenSea



## Etapa 7

# O protocolo IPFS

# Introdução

Nesta aula vamos falar sobre o protocolo IPFS, responsável pelo armazenamento de arquivos.

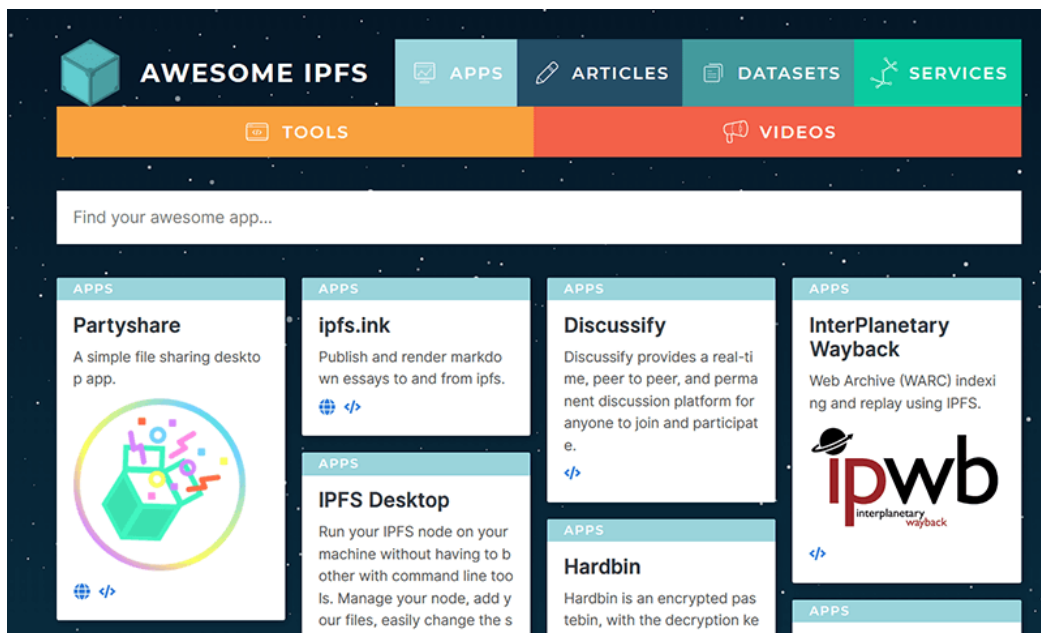


# Introdução

**IPFS** significa *Interplanetary File System*, um protocolo baseado em blockchain para transferência e armazenamento de arquivos.



# IPFS



# IPFS

## Install IPFS

Join the future of the web right now — just choose the option that's right for you.

### Store and provide files



#### IPFS Desktop

##### IPFS for everyone

The IPFS Desktop app offers menubar/tray shortcuts and an easy interface for adding, pinning, and sharing files — plus a full IPFS node ready for heavy-duty hosting and development. Great for developers and less experienced users alike.

Install IPFS Desktop



#### Command-line install

##### All IPFS, no frills

Just want to use IPFS from your terminal? Follow these step-by-step instructions for getting up and running on the command line using the Go implementation of IPFS. Includes directions for Windows, macOS, and Linux.

Install the IPFS CLI

# IPFS



## IPFS Companion

Add IPFS to your browser

Get `ipfs://` address support and more in your browser with this extension for Chromium and Firefox.

[Get IPFS Companion](#)



## IPFS Cluster

Orchestrate multiple IPFS nodes

Automatically allocate, replicate, and track your data as a global pinset distributed among a swarm of peers.

[Get IPFS Cluster](#)

## For developers



## Go implementation

The original IPFS implementation: IPFS core, daemon server, CLI tooling, and more.

[Get go-ipfs](#)



## JS implementation

IPFS implemented entirely in JavaScript for a world of possibilities in the browser and Node.js.

[Get js-ipfs](#)

# IPFS

Tem como objetivo tornar a transferência de arquivos na internet mais eficiente e barata, descentralizada e redundante e preservação dos arquivos.



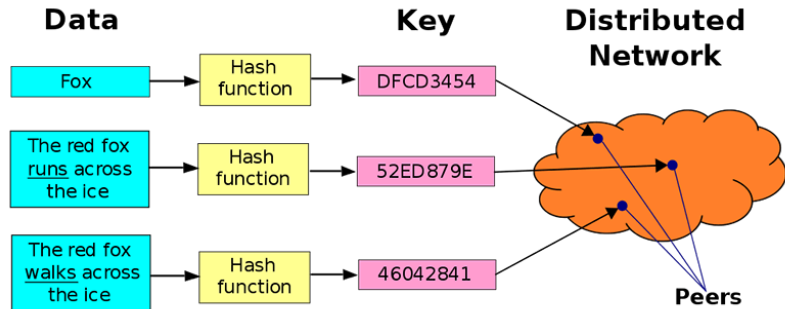
# Características

Entrega de conteúdo mais rápida e eficiente, para baixar pedaços de arquivos de nós geograficamente próximos, minimizando a latência.



# Características

Quando um arquivo é adicionado à rede é dividido em blocos com um ID único.



# Prática

Vamos configurar o IPFS e enviar arquivos para a rede.

Link para o IPFS: <https://ipfs.io/>



# Dúvidas?

- > Fórum/Artigos
- > Comunidade Online (Discord)

