

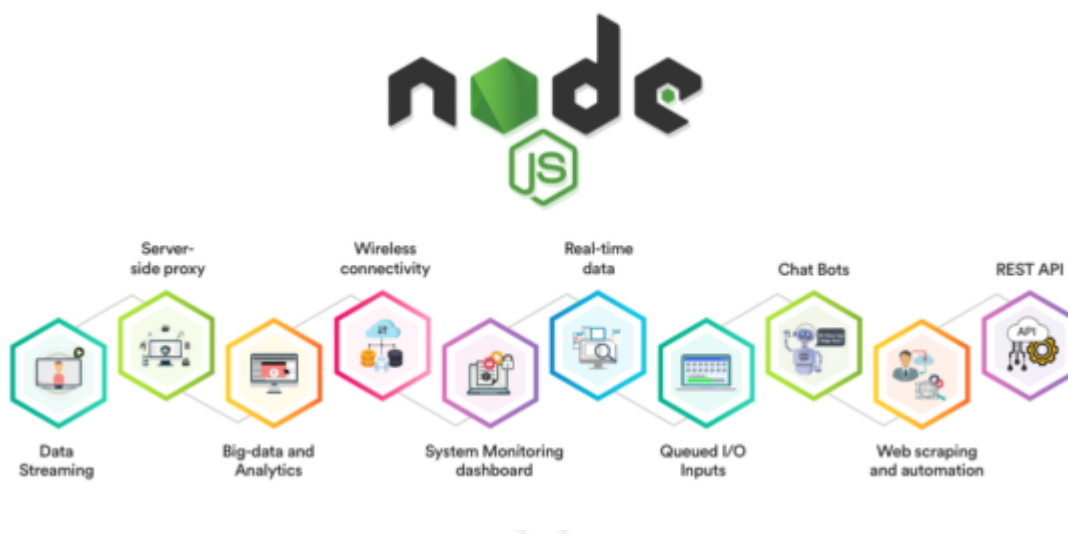
MATERIAL EXTRA

NodeJS



INTRODUÇÃO AO NODEJS

O **Node.js** é uma tecnologia que revolucionou o desenvolvimento web, permitindo que desenvolvedores usem JavaScript não apenas no Front-end, mas também no Back-end. Essa plataforma possibilita a execução de código JavaScript fora do navegador, sendo amplamente utilizada para criar servidores, APIs e aplicações escaláveis. Lançado em 2009 por Ryan Dahl, o **Node.js** foi projetado para ser leve, eficiente e capaz de lidar com conexões simultâneas, características fundamentais para a criação de sistemas modernos.



Disponível em [O que é Node.js e quais são as vantagens para as empresas?](#)

Antes do Node.js, o JavaScript era usado principalmente para criar interatividade em páginas web no navegador. Com essa plataforma, tornou-se possível desenvolver aplicativos completos apenas com JavaScript, unificando o Front-end e o Back-end em uma mesma linguagem. Esse diferencial simplifica o aprendizado e o desenvolvimento, além de possibilitar maior integração entre as equipes de programação.

O Node.js utiliza o [motor V8](#), desenvolvido pelo Google para o navegador Chrome, conhecido por sua alta performance. Essa base tecnológica garante que o Node.js seja rápido e eficiente, características ideais para aplicações como chats em tempo real, jogos multiplayer, APIs e até mesmo automações no desenvolvimento.

Por que aprender Node.js?

O Node.js é uma das tecnologias mais populares no mercado atual, com alta demanda entre desenvolvedores. Ele é ideal para iniciantes que já possuem algum conhecimento em JavaScript e desejam expandir suas habilidades para o Back-end. Além disso, sua vasta biblioteca de pacotes, disponível através do [NPM \(Node Package Manager\)](#), torna o desenvolvimento mais rápido e acessível, mesmo para quem está começando.

A plataforma também se destaca por sua **arquitetura não bloqueante** e programação assíncrona, que permitem gerenciar múltiplas conexões ao mesmo tempo sem comprometer o desempenho. Isso é especialmente importante em sistemas que precisam lidar com muitas solicitações simultâneas, como aplicativos de streaming ou redes sociais.

Iniciando com Node.js

Para começar a usar o Node.js, é necessário fazer o download e a instalação. O processo é simples:

1. Acesse o site oficial em <https://nodejs.org> e baixe o instalador.

2. Escolha a versão mais adequada para suas necessidades. A versão LTS (Long Term Support) é recomendada para iniciantes devido à sua estabilidade.
3. Após a instalação, é possível verificar se tudo está funcionando corretamente executando os comandos `node -v` e `npm -v` no terminal. Esses comandos mostram as versões do Node.js e do NPM instaladas.

Depois de instalado, você pode criar seu primeiro projeto. Para isso, basta inicializar um novo repositório usando o comando `npm init -y`, que gera automaticamente o arquivo `package.json`. Esse arquivo é como um guia do projeto, contendo informações como nome, versão, dependências e scripts.

Primeiros Passos: Hello World com Node.js

Uma das melhores maneiras de começar é criar um simples "Hello World". Para isso, crie um arquivo chamado `app.js` e escreva o seguinte código:

```
console.log("Hello World!");
```

Depois, no terminal, execute o arquivo com o comando:

```
node app.js
```

Você verá a mensagem "Hello World!" no terminal. Esse exemplo simples demonstra como o Node.js pode executar código JavaScript diretamente no servidor.

Gerenciando Dependências com o NPM

O **Node Package Manager (NPM)** é uma ferramenta poderosa incluída no Node.js, usada para gerenciar bibliotecas e pacotes. Esses pacotes são módulos reutilizáveis que ajudam a acelerar o desenvolvimento. Por exemplo, ao invés de criar toda lógica do zero para encontrar números ímpares contidos em um array, você pode usar o [pacote odd](#), que simplifica essa tarefa.

Para instalar um pacote, basta usar o comando `npm install`. Por exemplo:

```
npm install odd
```

Isso adiciona o pacote ao projeto e registra a dependência no arquivo `package.json`. O NPM também permite buscar pacotes em seu repositório oficial, disponível em <https://www.npmjs.com>.

Executando Pacotes com o NPX

O **NPX (Node Package Executor)** é outra ferramenta útil para rodar pacotes sem instalá-los no projeto. Ele é ideal para tarefas temporárias, como criar novos projetos ou executar comandos únicos. Um exemplo simples de seu funcionamento pode ser visto executando o comando:

```
npx cowsay "Hello World!"
```

Esse comando executa o pacote [cowsay](#) disponível no NPM sem a necessidade de instalação global do pacote. A saída para esse comando será:

```
< Hello World! >
```

```

      ^ _ ^
     \   /
    (oo)\_____
      (__)\       )\/\
           ||----w |
           ||     ||

```

Scripts no Node.js

No arquivo `package.json`, é possível definir scripts que automatizam tarefas no projeto. Por exemplo, para rodar as instruções escritas no arquivo `app.js`, você pode adicionar o seguinte script:

```
"scripts": {
  "start": "node app.js"
}
```

Com isso, você pode simplesmente digitar `npm run start` no terminal para rodar seu código JavaScript com o Node.js. Os Scripts tornam o fluxo de desenvolvimento mais organizado e eficiente.

BONS ESTUDOS