UNIVERSIDADE FUMEC
FACULDADE DE CIÊNCIAS EMPRESARIAIS
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

GUILHERME HENRIQUE LUIZ E. PEREIRA

KAREN PEREIRA DA SILVA MOREIRA

ATIVIDADE AUTO-INSTRUNCIONAL
NODE.JS

ORIENTADOR: Prof. FLAVIO LAPER

DISCIPLINA: DESENVOLVIMENTO WEB BACKEND

BELO HORIZONTE

2023

UNIVERSIDADE FUMEC
FACULDADE DE CIÊNCIAS EMPRESARIAIS
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

GUILHERME HENRIQUE LUIZ E. PEREIRA

KAREN PEREIRA DA SILVA MOREIRA

ATIVIDADE AUTO-INSTRUNCIONAL
NODE.JS

Pesquisa apresentada ao Prof. Flavio Velloso Laper como requisito PARCIAL para aprovação na disciplina Desenvolvimento Web Backend do curso de Ciência da Computação da Universidade FUMEC.

BELO HORIZONTE

2023

UNIVERSIDADE FUMEC

SUMÁRIO

UNIVERSIDADE FUMEC

INTRODUÇÃO

O Node.js, concebido por Ryan Dahl em 2009, surgiu como resposta à demanda por aplicações em tempo real que exigissem escalabilidade. Prévio ao advento do Node.js, o JavaScript era predominantemente empregado no navegador para interações no lado do cliente. A visão de Dahl consistiu em estender a utilização do JavaScript para o lado do servidor, viabilizando a construção de aplicações eficientes e escaláveis.

O propósito primordial do Node.js é proporcionar uma plataforma para o desenvolvimento de servidores web eficientes e não bloqueantes. Essa premissa é concretizada por meio do emprego do modelo assíncrono orientado a eventos, que permite a execução simultânea de diversas operações sem a necessidade de aguardar a conclusão de uma antes de iniciar outra. Essa abordagem revela-se particularmente vantajosa em cenários que demandam interações em tempo real, como em chats, jogos online e aplicações colaborativas.

Ao longo do tempo, o Node.js consolidou sua posição de destaque devido à sua eficiência, escalabilidade e à habilidade de compartilhar código entre o lado do cliente e do servidor, utilizando a mesma linguagem de programação, o JavaScript. Tornou-se uma escolha recorrente para desenvolvedores que buscam criar aplicações web modernas, apoiado por uma comunidade ativa de desenvolvedores e bibliotecas (módulos) acessíveis por meio do npm (Node Package Manager).

A governança do Node.js adota um modelo colaborativo de tomada de decisões denominado Consenso. A condução do projeto é mantida pelos Colaboradores, continuamente incorporados pelo Comitê de Direção Técnica (TSC), baseando-se em contribuições valiosas. O TSC desempenha um papel essencial na orientação de alto nível e na nomeação de Colaboradores, assegurando eficácia e colaboração no desenvolvimento da plataforma. Este trabalho explora a singular governança do Node.js e seu impacto no sucesso e evolução contínua da tecnologia.

No tocante às políticas de segurança e suporte ao usuário, o Node.js ostenta políticas de segurança ativas, detalhadas em sua página específica. Para relatar bugs de segurança, recomenda-se a utilização do HackerOne, garantindo uma resposta em até 5 dias. Ademais, o projeto mantém um programa de recompensas destinado a pesquisadores de segurança, gerenciado pelo HackerOne. Eventuais falhas em módulos de terceiros devem ser comunicadas a seus respectivos mantenedores. O processo de divulgação de segurança compreende a confirmação, auditoria de código, preparação de correções e uma data de divulgação coordenada, assegurando uma abordagem consistente. As atualizações de segurança são disseminadas por métodos específicos, como a lista de discussão de segurança do Node.js e o blog. A política alinha-se às Melhores Práticas da OpenSSF, sendo possível a certificação do projeto com o distintivo de Melhores Práticas da OpenSSF para indicar conformidade.

Ressalta-se que o Node.js é atualizado regularmente para incorporar melhorias contínuas, refletindo seu compromisso com a inovação e adaptação às demandas do desenvolvimento de software contemporâneo.

UNIVERSIDADE FUMEC

DESCRIÇÃO GERAL DO FRAMEWORK NODE.JS

Node.js é um ambiente de execução de JavaScript do lado do servidor, construído sobre o motor V8 da Google Chrome. Ele permite que os desenvolvedores usem JavaScript para escrever scripts do lado do servidor executando-os de forma assíncrona e orientada a eventos, proporcionando assim desenvolvimento de aplicativos de rede escaláveis. Em um exemplo simples, o Node.js consegue lidar com várias conexões simultaneamente, acionando um retorno de chamada (callback) quando ocorre uma conexão. Se não há tarefa pendente, o Node.js entra em modo de espera (sleep).

```
const http = require('node:http');
const hostname = '127.0.0.1';
const port = 3000;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});
server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Este código cria um servidor web básico que responde com "Hello World" a cada solicitação de conexão. O Node.js adota um modelo de concorrência não bloqueante, dispensando o uso de threads do sistema operacional, o que o torna mais eficiente e fácil de usar em comparação com modelos que dependem de threads.

Diferentemente de abordagens convencionais que empregam threads para concorrência, o Node.js supera a limitação de bloqueios de processos, uma vez que não faz uso de travas. A maioria das funções no Node.js evita bloqueios, a menos que métodos síncronos da biblioteca padrão sejam explicitamente utilizados.

O Node.js, inspirado em frameworks como Event Machine em Ruby e Twisted em Python, avança ao incorporar o loop de eventos como parte integral da construção de tempo de execução. Contrariamente a outros sistemas, não há uma chamada de bloqueio para iniciar o loop de eventos no Node.js. Ele inicia o loop após executar o script e encerra quando não há mais retornos de chamada para serem processados.

O tratamento do HTTP é fundamental para o Node.js, que é projetado para streaming e baixa latência, tornando-o adequado como base para bibliotecas e estruturas web.

Apesar da ausência de threads em seu design, o Node.js permite a utilização eficaz de vários núcleos, possibilitando a geração de processos filhos por meio da API child_process.fork(). Além disso, o módulo cluster, baseado na mesma interface, facilita o compartilhamento de soquetes entre processos para balanceamento de carga em diferentes núcleos.

UNIVERSIDADE FUMEC

APLICAÇÕES

O Node.js, sendo uma plataforma de execução de JavaScript assíncrona e orientada a eventos, possui uma variedade de aplicações amplamente utilizadas em contextos acadêmicos e empresariais. Abaixo, destacam-se algumas das principais aplicações do Node.js:

1. Desenvolvimento de Servidores Web Eficientes:

O Node.js é amplamente adotado no desenvolvimento de servidores web eficientes e escaláveis. Sua arquitetura não bloqueante e orientada a eventos torna-o adequado para lidar com um grande número de solicitações simultâneas, proporcionando um desempenho superior em ambientes de tráfego intenso.

2. Aplicações em Tempo Real:

Devido à sua capacidade de lidar com operações em tempo real de forma eficiente, o Node.js é frequentemente utilizado no desenvolvimento de aplicações que exigem interações instantâneas, como chats online, transmissões ao vivo, jogos em tempo real e colaborações em tempo real.

3. APIs e Microsserviços:

O Node.js é uma escolha comum para o desenvolvimento de APIs (Interface de Programação de Aplicações) e microsserviços. Sua leveza e eficiência tornam-no adequado para criar componentes de software modulares e escaláveis.

4. Desenvolvimento de Aplicações de Rede:

Dada a sua natureza orientada a eventos, o Node.js é frequentemente empregado no desenvolvimento de aplicações de rede, incluindo ferramentas de monitoramento, proxies, balanceadores de carga e outras soluções que requerem comunicação eficiente pela rede.

5. Ferramentas de Desenvolvimento:

O Node.js é utilizado na construção de ferramentas de desenvolvimento, como bundlers, task runners e transpiladores de código, devido à sua capacidade de automação de tarefas e manipulação eficiente de arquivos.

6. Aplicações de Streaming:

Sua capacidade de lidar com operações de streaming o torna adequado para o desenvolvimento de aplicações que envolvem processamento contínuo de dados, como transmissões de áudio e vídeo, processamento de dados em tempo real e manipulação de fluxos de dados.

7. Internet das Coisas (IoT):

O Node.js é aplicado em cenários de Internet das Coisas devido à sua eficiência e capacidade de lidar com operações assíncronas, permitindo o desenvolvimento de aplicações para dispositivos IoT.

8. Desenvolvimento de Aplicações de Desktop:

   Com o auxílio de frameworks como Electron, o Node.js é utilizado no desenvolvimento de aplicações de desktop multiplataforma, permitindo que desenvolvedores utilizem tecnologias web para criar aplicativos nativos.

Essas aplicações do Node.js destacam sua versatilidade e eficácia em diversos contextos, tornando-o uma escolha proeminente na comunidade de desenvolvimento de software.

PRINCIPAIS CARACTERÍSTICAS

Node.js é um ambiente de execução de JavaScript do lado do servidor, construído sobre o motor V8 da Google Chrome. Ele permite que os desenvolvedores usem JavaScript para escrever scripts do lado do servidor, proporcionando assim uma execução eficiente e escalável de código.

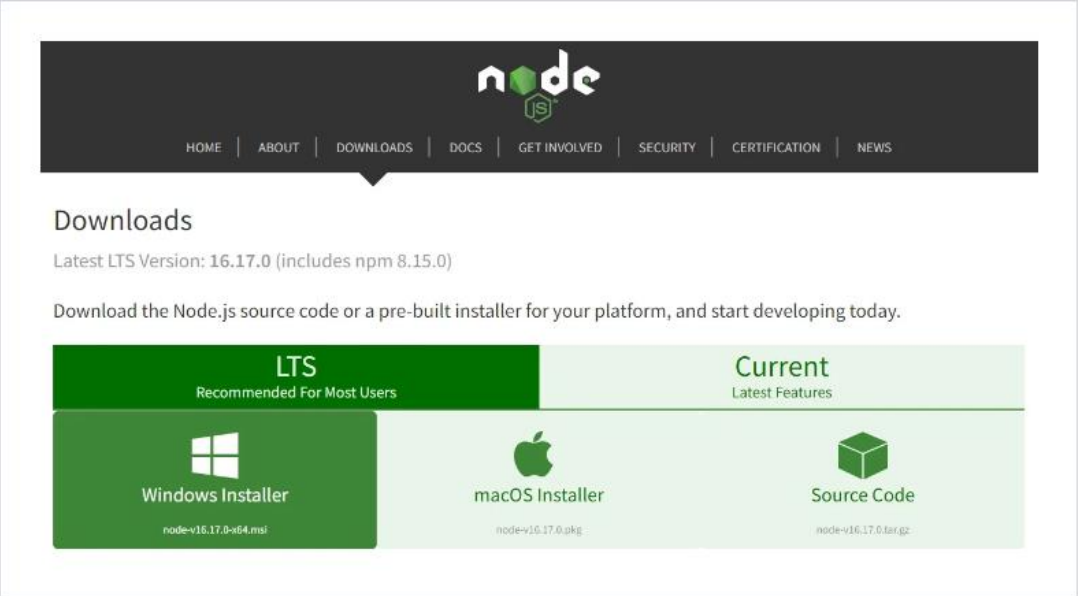
As principais características do Node.js incluem:

- JavaScript no Servidor: Antes do Node.js, o JavaScript era principalmente usado no navegador para interações do lado do cliente. Com o Node.js, os desenvolvedores podem usar JavaScript também no servidor.
- Assincronismo: Node.js é conhecido por seu modelo de I/O não bloqueante, o que significa que ele pode lidar com muitas operações simultaneamente, sem esperar que uma operação seja concluída antes de começar outra. Isso é feito usando chamadas de retorno (callbacks) e, mais recentemente, Promises e async/await.
- Módulos: Node.js utiliza um sistema de módulos que permite aos desenvolvedores organizar seu código em módulos reutilizáveis e fáceis de manter. Módulos podem ser incorporados nos scripts Node.js usando a função require().
- V8 JavaScript Engine: O Node.js é construído sobre o motor V8, o mesmo motor JavaScript de alto desempenho usado no navegador Google Chrome.
- Comunidade Ativa: Node.js tem uma comunidade robusta e ativa, o que significa que há uma grande quantidade de módulos e bibliotecas disponíveis através do npm (Node Package Manager), facilitando o desenvolvimento de aplicações complexas.
- Escalabilidade: Por causa do seu modelo assíncrono e da capacidade de lidar com muitas conexões simultâneas, o Node.js é conhecido por sua capacidade de escalabilidade, sendo muitas vezes utilizado em aplicações em tempo real, como chat em tempo real, jogos online, entre outros.

Node.js é amplamente utilizado para desenvolvimento web, especialmente em aplicações que requerem comunicação em tempo real e alto desempenho. Ele se tornou uma escolha popular para construção de servidores e APIs, contribuindo para o desenvolvimento eficiente de aplicações modernas.

INSTALAÇÃO

A instalação do Node.js pode variar dependendo do sistema operacional que você está usando. Abaixo, forneço instruções básicas para os sistemas operacionais mais comuns: Windows, macOS e Linux. No geral, é possível fazer as instalações seguindo as orientações do site (https://nodejs.org/en/download)



1. Windows:
    a. Baixe o Instalador:
        i. Acesse o site oficial do Node.js em https://nodejs.org/.
        ii. Na página inicial, você verá a opção para baixar o "LTS" (Long-Term Support), que é recomendado para a maioria dos usuários.
    b. Execute o Instalador:
        i. Após o download, execute o instalador baixado (o arquivo .msi).
        ii. Siga as instruções do assistente de instalação.
        iii. Aceite os termos do contrato de licença e clique em "Next".
    c. Configurações Padrão:
        i. Deixe as configurações padrão durante a instalação, a menos que você tenha um motivo específico para alterá-las.
    d. Conclua a Instalação: Clique em "Next" e, em seguida, em "Finish" para concluir a instalação.
    e. Verifique a Instalação: Abra o Prompt de Comando (CMD) ou PowerShell e execute o comando node -v para verificar se o Node.js foi instalado corretamente.

2. MacOS:
    a. Usando o Homebrew (recomendado):
        i. Abra o Terminal.
        ii. Instale o Homebrew se ainda não estiver instalado: /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
        iii. Em seguida, instale o Node.js com o comando: brew install node
    b. Usando o Instalador Gráfico:
        i. Baixe o instalador do Node.js no site oficial https://nodejs.org/.

        ii. Execute o instalador gráfico e siga as instruções.
    c. Verifique a Instalação:
        i. Abra o Terminal e execute node -v para verificar a versão do Node.js.
3. Linux:
    a. Usando o Gerenciador de Pacotes (Debian/Ubuntu):
        i. Abra o Terminal.
        ii. Execute os seguintes comandos:
            sudo apt update
            sudo apt install nodejs npm
    b. Usando o Gerenciador de Pacotes (Fedora):
        i. Abra o Terminal.
        ii. Execute os seguintes comandos:
            sudo dnf install nodejs npm
    c. Outras Distribuições:
        i. Para outras distribuições Linux, consulte a documentação oficial do Node.js para instruções específicas.
    d. Verifique a Instalação:
        i. No Terminal, execute node -v para verificar a versão do Node.js.

UNIVERSIDADE FUMEC

MODO DE UTILIZAÇÃO

O Node.js é altamente modular e pode ser utilizado com uma variedade de frameworks e bibliotecas para atender a diferentes necessidades de desenvolvimento. Além disso, sua interoperabilidade com várias linguagens é possível por meio de módulos nativos e ferramentas de integração. Abaixo estão alguns dos frameworks populares e linguagens com as quais o Node.js pode ser utilizado:

1. Frameworks de Backend Populares para Node.js:
    a. Express.js: é um framework popular para o Node.js, conhecido por sua simplicidade e flexibilidade. Ele é amplamente utilizado para criar aplicativos web e APIs de maneira eficiente, oferecendo recursos úteis, como gerenciamento de rotas e suporte a diferentes tipos de solicitações HTTP.
    b. Socket.IO: é um framework essencial para comunicações em tempo real. Ele permite que servidores e clientes troquem mensagens instantâneas, sendo especialmente útil em aplicações como chats e jogos online.
    c. Nest.js: é mais recente, focado em escalabilidade para aplicações Node.js. Ele segue padrões arquiteturais sólidos e é inspirado em Angular, proporcionando uma abordagem estruturada e modular para o desenvolvimento.
    d. Adonis.js: é um framework completo que segue o padrão MVC (modelo, visão e controlador), oferecendo uma ampla gama de funcionalidades. É uma escolha sólida se você busca convenções claras e estrutura organizada em seus projetos.
    e. Koa.js: O Koa.js, mais recente que o Express.js, é conhecido por sua leveza e modularidade. Utilizando conceitos modernos do JavaScript, é uma escolha expressiva para desenvolvimento.
    f. Meteor.js: Uma plataforma full-stack que simplifica o desenvolvimento de aplicações web em tempo real, proporcionando integração fácil entre o frontend e o backend.

2. Bibliotecas de Frontend Populares para Node.js:
    a. React.js: é uma biblioteca JavaScript mantida pelo Facebook. É utilizada para construir interfaces de usuário interativas, sendo eficiente no gerenciamento de estados dinâmicos e na atualização eficiente do DOM. A integração com Node.js permite a construção de aplicações full-stack usando a mesma linguagem (JavaScript) em ambos os lados, simplificando o desenvolvimento.
    b. Angular.js:mantido pelo Google, é um framework JavaScript completo para o desenvolvimento de aplicações web. Ele oferece uma estrutura robusta para a criação de aplicativos frontend complexos. Embora seja mais pesado em comparação com bibliotecas como React, é poderoso para projetos que exigem uma arquitetura completa e padronizada.
    c. Vue.js: é uma biblioteca JavaScript progressiva para a construção de interfaces de usuário. É conhecida por sua simplicidade e flexibilidade. Vue.js pode ser facilmente integrado a projetos

Node.js para o desenvolvimento de interfaces de usuário reativas e dinâmicas.

d. Svelte: é uma abordagem diferente para o desenvolvimento frontend. Ao contrário de outras bibliotecas que são interpretadas no navegador, o Svelte move a maior parte do trabalho para o tempo de compilação, resultando em um código otimizado e eficiente. Pode ser integrado com Node.js para criação de componentes web rápidos e eficientes.

3. Linguagens e Módulos Nativos:

   a. JavaScript: A linguagem principal usada com Node.js. Permite o desenvolvimento tanto no lado do servidor quanto do cliente.
   b. TypeScript: Uma superset da linguagem JavaScript que adiciona tipagem estática. Muitos desenvolvedores optam por usar TypeScript com Node.js para obter benefícios adicionais durante o desenvolvimento.
   c. C/C++: Módulos nativos do Node.js podem ser escritos em C ou C++, permitindo integração com bibliotecas existentes escritas nessas linguagens.
   d. Rust: Algumas iniciativas e módulos experimentais têm explorado a integração do Rust com Node.js para melhorar o desempenho em certos casos.
   e. Python, Java, .NET, etc.: Embora o Node.js seja predominantemente JavaScript, é possível integrar módulos de outras linguagens, geralmente por meio de APIs ou chamadas de sistema externas.

A interoperabilidade do Node.js com diversas linguagens e a abundância de frameworks tornam-no uma escolha flexível para desenvolvedores, permitindo a construção de uma ampla gama de aplicações. Unido a bibliotecas populares no desenvolvimento frontend, o Node.js possibilita a criação de aplicações web coesas e eficientes, abrangendo tanto o lado do servidor quanto o lado do cliente.

UNIVERSIDADE FUMEC

CONCLUSÃO

Em uma análise abrangente, o Node.js, concebido por Ryan Dahl em 2009, apresenta-se como uma peça fundamental no panorama do desenvolvimento de software, marcando uma mudança paradigmática ao estender a utilização do JavaScript para o ambiente do servidor. Ao superar as limitações dos modelos tradicionais e introduzir uma arquitetura não bloqueante orientada a eventos, o Node.js conquistou reconhecimento por sua eficiência em ambientes de alta concorrência, onde operações simultâneas e interações em tempo real são imperativos.

A governança do Node.js, pautada pelo modelo colaborativo de consenso, destaca-se como um componente essencial para sua evolução contínua. Sob a gestão dos Colaboradores, continuamente incorporados pelo Comitê de Direção Técnica (TSC), o projeto demonstra uma abordagem participativa na tomada de decisões e na nomeação de contribuidores valiosos. Essa estrutura colaborativa é essencial para orientar o desenvolvimento de forma eficaz.

Na esfera da segurança, o Node.js adota políticas ativas detalhadas em sua página específica. O emprego do HackerOne como plataforma para relatos de bugs de segurança, juntamente com um programa de recompensas para pesquisadores de segurança, reforça o compromisso com a integridade e a segurança da plataforma. Além disso, a política de divulgação de segurança, com suas etapas bem definidas, proporciona uma resposta coordenada e consistente a eventuais vulnerabilidades.

A vitalidade do Node.js é ressaltada pela sua constante evolução. A incorporação regular de melhorias reflete o comprometimento com a inovação e a capacidade de adaptação às demandas dinâmicas do desenvolvimento de software contemporâneo. Essa atenção contínua à qualidade e eficiência consolida o Node.js como uma escolha robusta e confiável para uma variedade de contextos de desenvolvimento.

Em resumo, o Node.js não apenas revolucionou a maneira como concebemos e construímos aplicações, mas também estabeleceu-se como uma base sólida para desenvolvimento em diversos cenários. Sua versatilidade, eficácia e compromisso com a excelência tornam-no uma peça fundamental no arsenal de ferramentas para desenvolvedores modernos.

UNIVERSIDADE FUMEC

BIBLIOGRAFIA

Araújo, Iasmin; Silvério, Rafaela Petelin. Como instalar o Node.js no Windows, Linux e macOS. 24/10/2022. Disponível em: https://www.alura.com.br/artigos/como-instalar-node-js-windows-linux-maco. Acesso em: 27 de novembro, 2023.

Awari. Os Melhores Frameworks de Backend para Node.Js: Guia Completo. Publicado em 26 de agosto de 2023. Disponível em: https://awari.com.br/os-melhores-frameworks-de-backend-para-node-js-guia-completo/?utm_source=blog&utm_campaign=projeto+blog&utm_medium=os%20Melhores%20Frameworks%20de%20Backend%20para%20Node.Js:%20Guia%20Completo#:~:text=O%20Express.,de%20APIs%20e%20servidores%20web. Acesso em: 26 de novembro, 2023.

Bessa, André. Node.JS: o que é, como funciona esse ambiente de execução JavaScript e um Guia para iniciar. Atualizado em 18 de Setembro. Disponível em: https://www.alura.com.br/artigos/node-js. Acesso em: 25 de novembro, 2023.

Desconhecido. Os 15 Melhores Frameworks do Node.JS para Seu Projeto! Publicado em 26/10/2022. Atualizado em 28/10/2022. Tempo de leitura: 11 minutos. Disponível em: <https://blog.betrybe.com/framework-de-programacao/frameworks-nodejs/>. Acesso em: 26 de novembro, 2023.

Desconhecido. Tecnologia Node.js. Disponível em: https://www.devmedia.com.br/guia/node-js/40312. Acesso em: 28 de novembro, 2023.

E., C. (ago 29, 2023). O Que é Node.Js: Aplicações Práticas e Como Instalá-lo. Hostinger. Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-node-js?ppc_campaign=google_search_generic_hosting_all&bidkw=defaultkeyword&lo=1031849&gad_source=1&gclid=CjwKCAiAvJarBhA1EiwAGgZl0PjugoVBNwcGgKsMlgo4rc7RuZkyzBO1l5Fl90murtNjXWfUR_Kv9BoC2_sQAvD_BwE#O_Nodejs_e_Apenas_JavaScript>. Acesso em: 25 de novembro, 2023.

Evaldo, A. (11 de Outubro). Node.js para Frameworks Front-end. Alura. Disponível em: <https://www.alura.com.br/artigos/nodejs-para-frameworks-front-end>. Acesso em: 26 de novembro, 2023.

Eseme, S. (Agosto 22, 2023). Os 10 Tipos mais Populares de Aplicativos Node.js em 2023. Kinsta. Disponível em: <https://kinsta.com/pt/blog/aplicativos-node-js/>. Acesso em: 26 de novembro, 2023.

Node.js. (s.d.). About Node.js®. Recuperado de https://nodejs.org/en/about. Acesso em: 25 de novembro, 2023.

Node.js. (s.d.). Documentation. Recuperado de https://nodejs.org/en/docs. Acesso em: 25 de novembro, 2023.

Wikipédia. Node.js. Disponível em: <https://pt.wikipedia.org/wiki/Node.js>. Acesso em: 25 de novembro, 2023.