PROGRAMAÇÃO WEBII

1

Ferramentas de Trabalho

Ferramentas de Trabalho

NodeJS
É uma plataforma para construir aplicações web escaláveis de alta performance usando JavaScript.

NPM
É usado como um repositório para publicação de projetos
com código aberto

ANGULAR
É uma plataforma e framework para construção da interface de aplicações usando HTML, CSS e, principalmente, JavaScript, criada pelos desenvolvedores da Google.

Visual Studio Code
Editor de código-fonte desenvolvido pela Microsoft com controle Git incorporado, realce de sintaxe, complementação inteligente de código, etc.

Ferramentas de Trabalho

Git
Git é um sistema de controle de versão de arquivos com contribuição simultâneas entre pessoas.

Github
GitHub é uma plataforma de hospedagem de códigofonte com controle de versão usando o Git.

Insomnia
É uma ferramenta para teste de API´s que utilizam o padrão REST, permitindo simular requisições HTTP de forma rápida.

2

Node.js

Instalação

Agenda

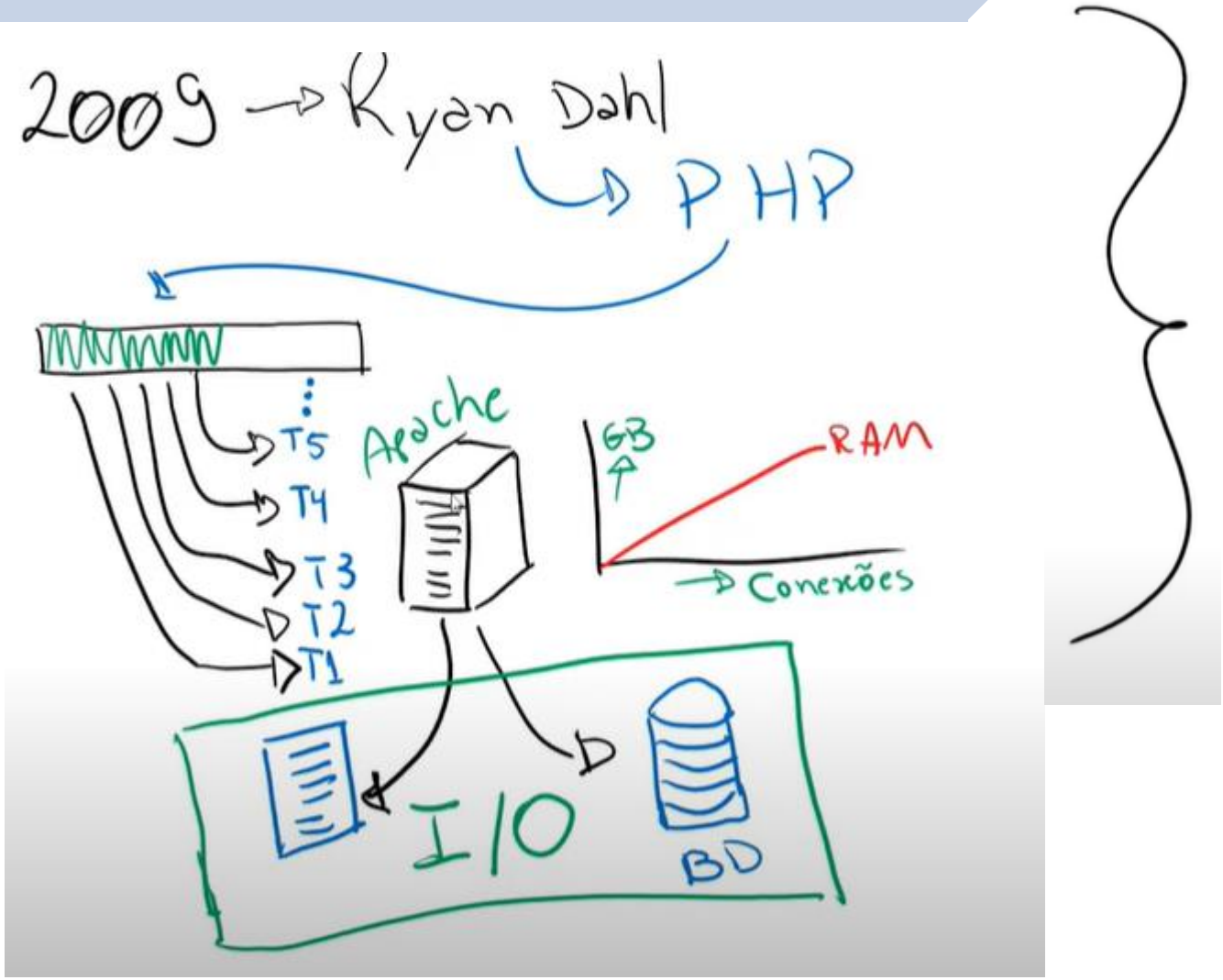
- História
- O que é
- Funcionamento
- NPM
- Pacote / Módulo
- Utilização
- Instalação

História

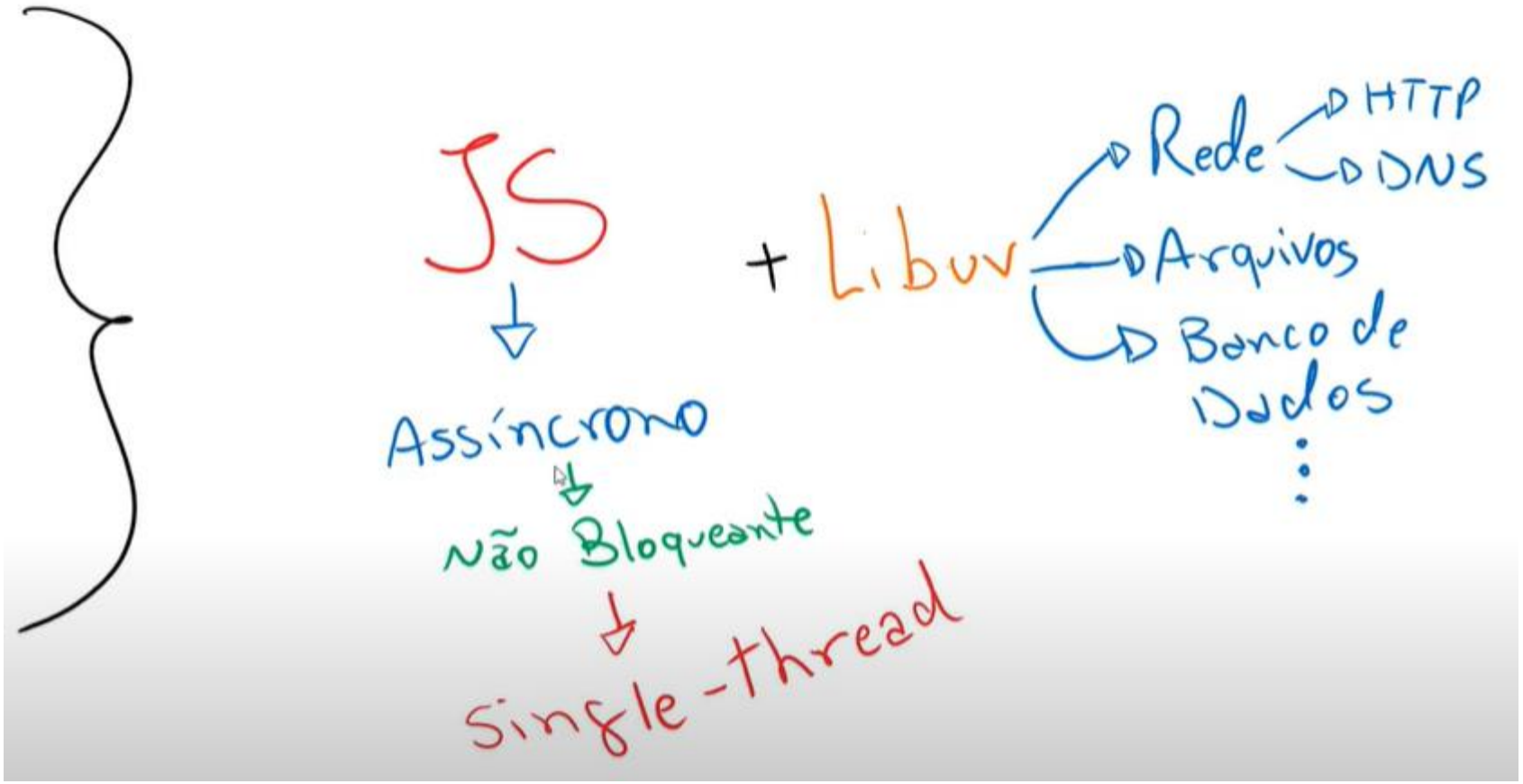
- Foi anunciado em 2009 pelo pesquisador Ryan Dahl e desenvolvido junto a mais 14 Colaboradores. Ele nasceu pela observação de como os servidores web trabalhavam e se comportavam nesta época.
- Ele analisou que ações simples de uma aplicação web podem realizar muitas requisições, como o carregar de uma barra de progresso, gerando consumo excessivo de recursos no servidor(CPU, RAM, etc.
- Testou várias linguagens, antes de escolher o Javascript.
- Apresentou na JSConf EU" uma combinação do motor Javascript V 8 com um event loop fornecido por uma API de Input/Output de baixo nível chamada libuv feita em C++
- Apresentação original

https://pt.slideshare.net/AartiParikh/original-slides-from-ryan-dahls-nodejs-intro-talk

História



História


JS
+ Libuv
Rede
HTTP
DNS
Arquivos
Banco de Dados
Assíncrono
Não Bloqueante
Single-thread

O que é?

- O Node js (para o Ryan Dahl é um “conjunto de bibliotecas que são executadas no mecanismo V 8 permitindo executar o código Javascript no servidor”
- De forma muito resumida, ele permite que executemos Javascript no servidor, mantendo seu conceito de single thread, sem bloquear as requisições ou execuções.

ST só pode tratar uma requisição de cada vez, então o processamento de cada uma não pode ser demorado, nem pode bloquear (por exemplo, ficar esperando pelo banco de dados). Um servidor ST pode ser eficaz, desde que nunca bloqueie. O Node.js é assíncrono de modo a não bloquear. Qualquer processamento demorado deve ser delegado a um outro processo, o que também pode ser feito no Node com subprocess.

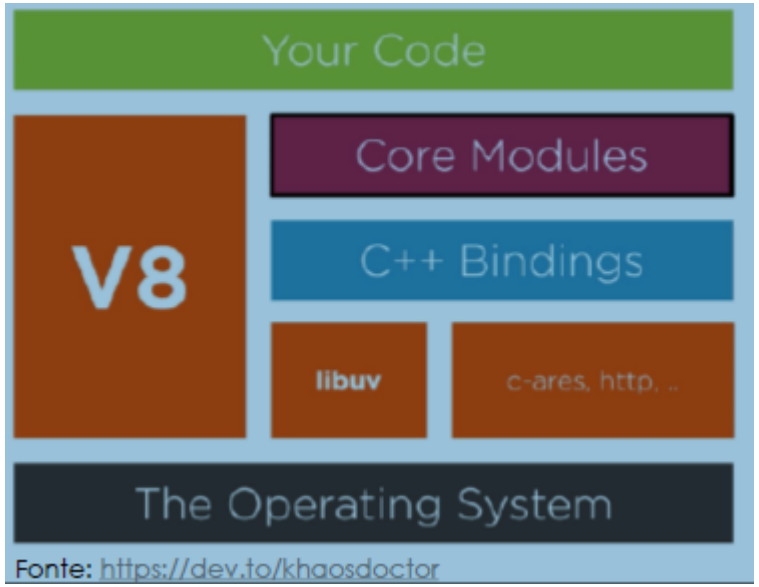
MT, assumindo que se crie uma thread por requisição, pode tratar várias requisições em paralelo, mesmo que demorem ou bloqueiem.

O que é?

- Esta tecnologia possui um modelo inovador: sua arquitetura é totalmente *non-blocking thread* (não-bloqueante) que apresenta uma boa performance com consumo de memória e utiliza ao máximo e de forma eficiente o poder de processamento dos servidores, principalmente em sistemas que produzem uma alta carga de processamento.
- **Sistemas criados com Node.js** estão livres de aguardarem por muito tempo o resultado de seus processos, e mais importante, não sofrem *dead-locks*, pelo simples fato de trabalharem em *single-thread*. Além dessas vantagens, desenvolver sistemas nessa plataforma é super simples e prático.
- O Node.js é uma plataforma altamente escalável e de baixo nível, logo, você terá a possibilidade de programar diretamente com diversos protocolos de rede e internet ou utilizar bibliotecas que acessam recursos do sistema operacional.

O que é?

- Ele é composto por:
  - **Motor Javascript V 8 da Google:** https://v8.dev/
  - **Libuv:** https://libuv.org/
  - http-parser: https://github.com/nodejs/http-parser
  - c-ares: https://c-ares.haxx.se/
  - Open SSL: https://www.openssl.org/
  - Zlib: https://zlib.net/



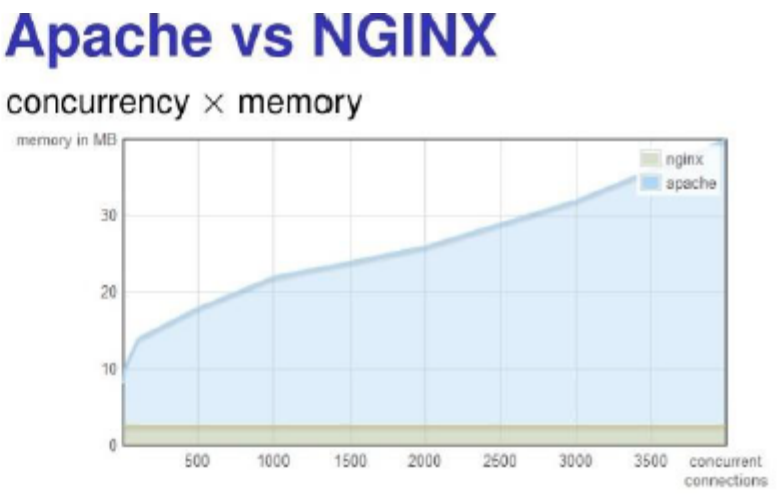
Fonte: https://dev.to/khaosdoctor

Funcionamento – Outros Serviços

- Normalmente, os servidores de aplicações web criam uma thread para cada requisição, podendo assim processar o que foi solicitado, sem bloquear novas requisições.
- Essas requisições podem demorar para serem processadas, deixando os recursos alocados parados até que tudo seja concluído.
- Quanto mais requisições, mais poder de CPU e quantidade de RAM são necessários.

Ao lado temos o comparativo utilizado por Ryan Dahl em sua apresentação do Node, que demonstra um comparativo entre o consumo de memória entre o Apache(*multi-thread*) e o Nginx(*single-thread*), ambos servidores de páginas PHP.

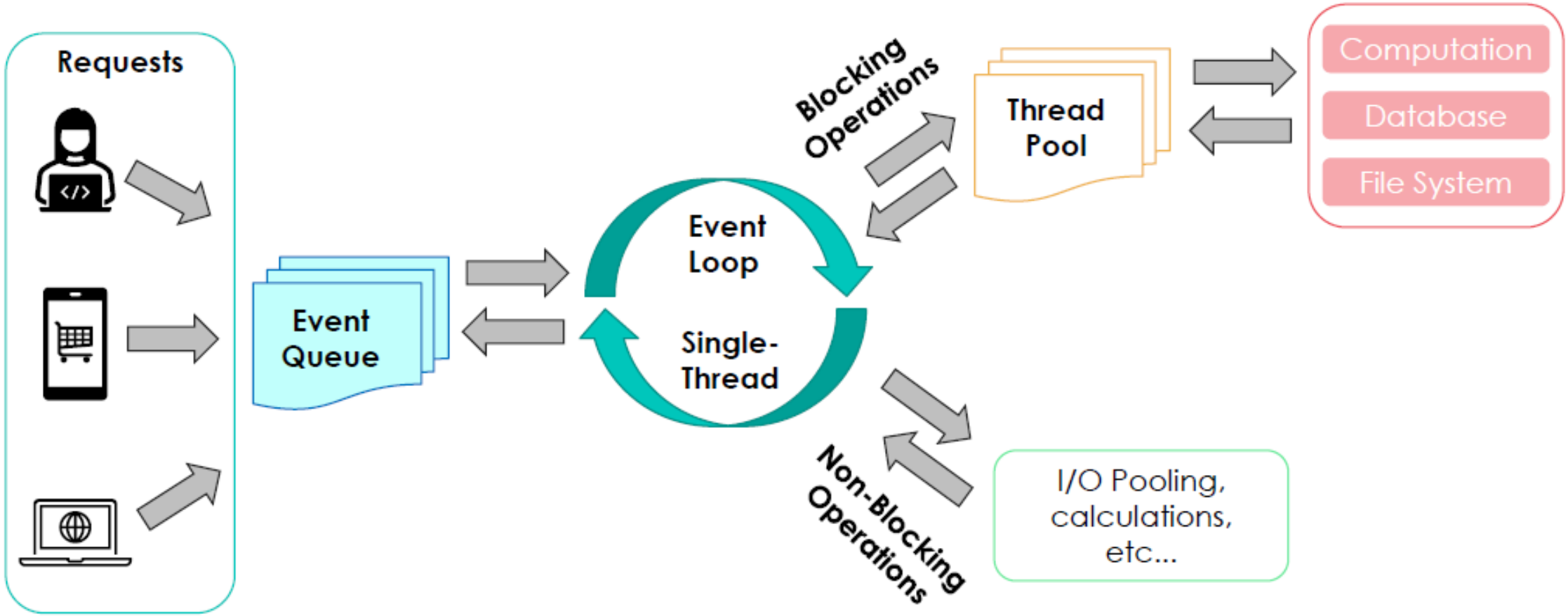
Quanto mais requisições, maior foi o consumo.



Funcionamento – Node

- O Node utiliza um Event Loop que é single thread para gerenciar as requisições. Cada nova requisição é tratada como um evento e colocado em uma fila.
- O V8 gerencia essa fila através de processamento assíncrono (não bloqueante), nenhuma operação precisa que a anterior seja finalizada.
- Toda vez que é solicitado um processamento bloqueante(Banco de dados, Arquivos do sistema), o V8 delega essa atividade para a Libuv , ficando disponível para novos
- A Libuv é multi thread (inicialmente com 4), aqui chamado de pool de threads , ela faz os processamentos de rede, bancos de dados, arquivos de sistema, entre outras operações.
- Após o processamento, a reposta(callback) é enviada de volta para o Event Loop

Funcionamento – Visão Gráfica



NPM

- O Node Package Manager(Gerenciador de Pacotes do Node) não é a mesma coisa que o Node.
- Atualmente ele faz parte do instalador padrão do Node.
- É um cliente de linha de comandos para auxiliar na instalação, gerenciamento da versões e desinstalação de pacotes(dependências) de um projeto.
- É um repositório utilizado para a publicação de projetos Node de código aberto (open source), ou seja, uma plataforma online onde qualquer pessoa pode publicar e compartilhar ferramentas escritas em Javascript

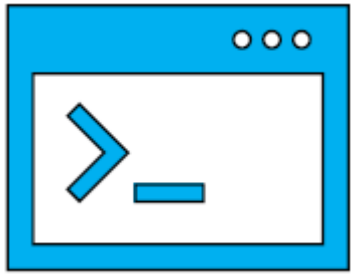
Pacote / Módulo

- Um pacote em Node.js contém todos os arquivos de que você precisa para um módulo.
- Módulos são bibliotecas Javascript que você pode incluir em seu projeto.
- Instalar e gerenciar módulos é uma das coisas mais básicas que você deve aprender a fazer quando começar a usar o gerenciador de pacote Node.

Utilização

- Utilizado para criação de APIs(Application Programming Interface que são interfaces que facilitam a troca de informações entre projetos, aplicações, equipamentos, mesmo que estes não tenham sido escritos com a mesma linguagem.

- Execução de códigos e frameworks Javascript.



- É gerenciado através do NPM.

- Todos as configurações de projeto e definições de dependências são armazenadas no arquivo package json localizado na raiz do projeto Node.

Comandos comuns

Comando	Descrição
npm init	Cria o arquivo package.json com as definições e configurações do projeto.
npm install <pacote>	Adiciona a dependência do pacote no package.json e faz sua instalação no projeto.
npm update	Atualiza todos os pacotes do projeto.
npm update <pacote>	Atualiza somente o pacote informado.
npm uninstall <pacote>	Remove a dependência do pacote no package.json e faz sua desinstalação no projeto
npm install –g <pacote>	Faz a instalação global do pacote, permite sua execução direta pelo prompt de comando.
npm uninstall –g <pacote>	Faz a desinstalação global do pacote.

Para mais comandos e informações: https://docs.npmjs.com/cli

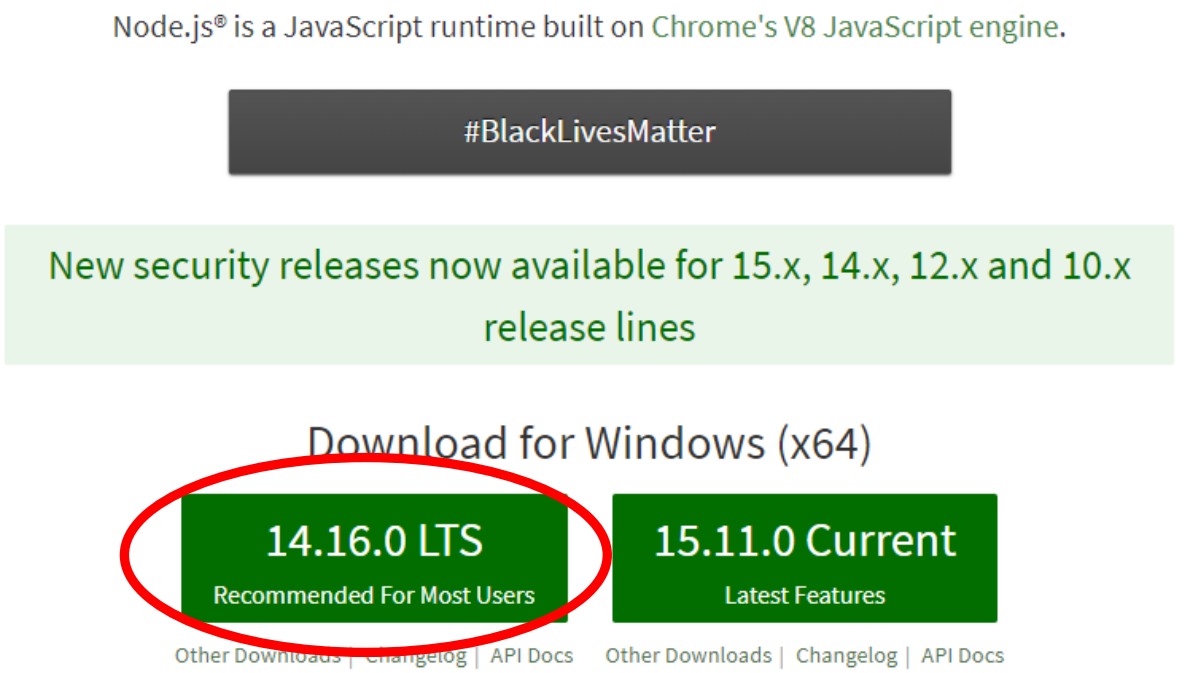
Instalação

- O node está disponível para sistemas operacionais Windows, Linux e Mac.
- Sempre utilize as versões mais recentes do tipo LTS.
- Para instalação no Windows e Mac, recomendo o download através da página oficial:

  https://nodejs.org/
- Para instalação no Linux recomendo a utilização do NodeSource :

  https://github.com/nodesource/distributions

Instalação



LTS – long-term support (suporte de longo prazo)