Desenvolvimento Web com Node.js e Express

Ítalo Berg M. S. Reis

INSTITUTO METRÓPOLE DIGITAL

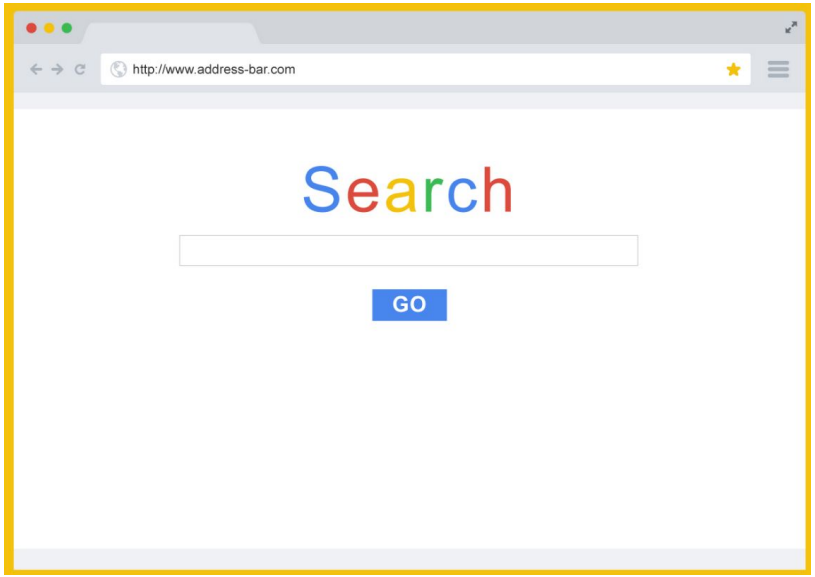
Ementa

- segunda 20/01 - Aula Teórica + Prática
- terça 21/01 - Aula Teórica + Prática
- quarta 22/01 - Desenvolver projeto em sala
- quinta 23/01 - Apresentação do projeto

INSTITUTO METRÓPOLE DIGITAL

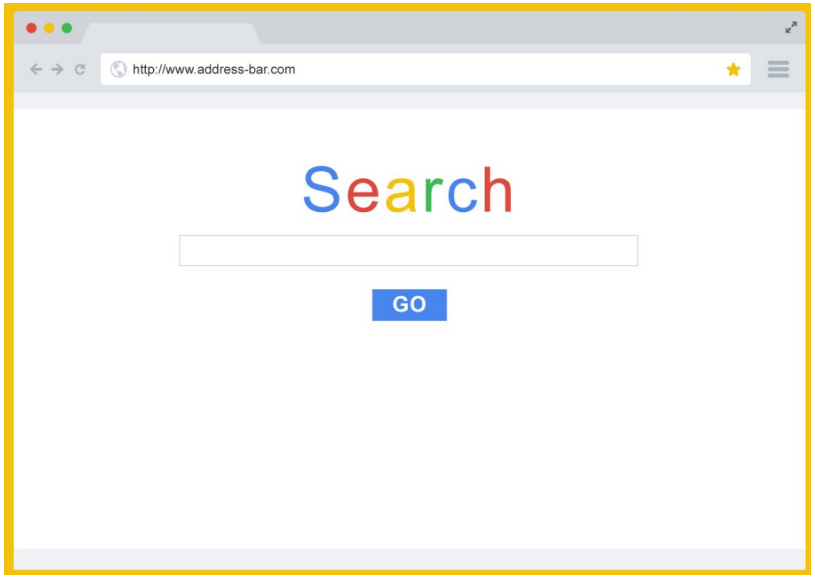
O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript



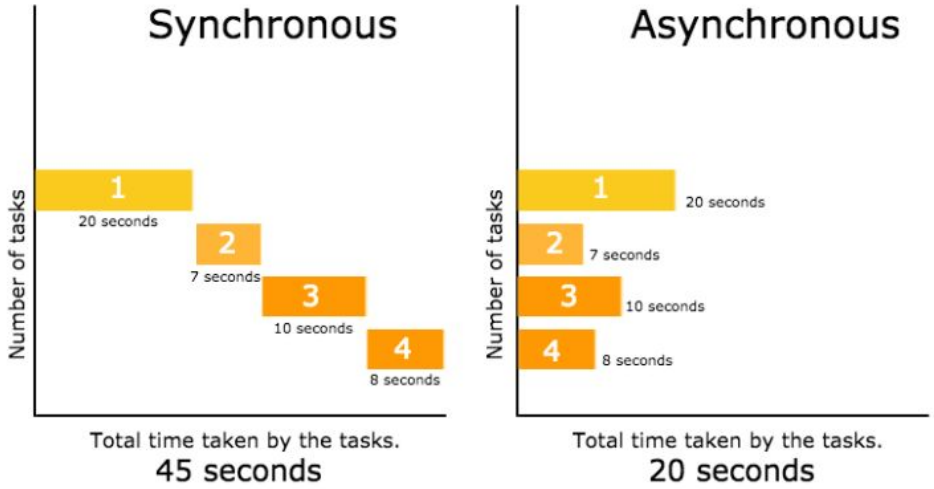
O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript
  - Engine do Google Chrome V8



O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript
- Assíncrono

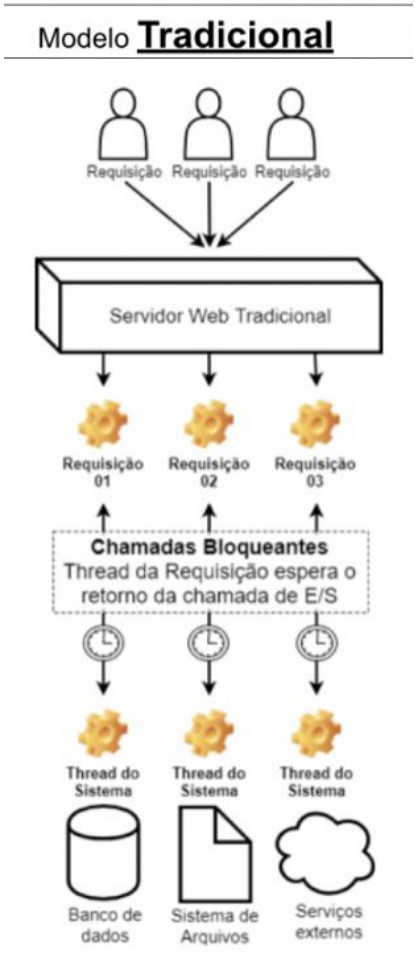


O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript
- Assíncrono
- Arquitetura não bloqueante
  - Thread única
- Baseado em evento

O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript
- Assíncrono
- Arquitetura não bloqueante
  - Thread única
- Baseado em evento



O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript
- Assíncrono
- Arquitetura não bloqueante
  - Thread única
- Baseado em evento

O que é Node?

- Ambiente para execução de código JavaScript
  - Interpretador JavaScript
- Assíncrono
- Arquitetura não bloqueante
  - Thread única
- Baseado em evento

O que é Node?

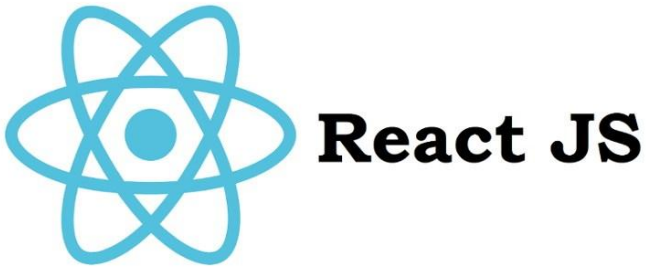
- Interpretador JavaScript
- Assíncrono
- Baseado em evento
- Engine do Google (Chrome e Node)

Vantagens

- Leve
- Usa pouca memória RAM
- Maior aproveitamento de CPU
- Grande quantidade de bibliotecas e módulos
- Uso de JavaScript (back e front)

Vantagens

- Leve
- Usa pouca memória RAM
- Maior aproveitamento de CPU
- Grande quantidade de bibliotecas e módulos
- Uso de JavaScript (back e front)



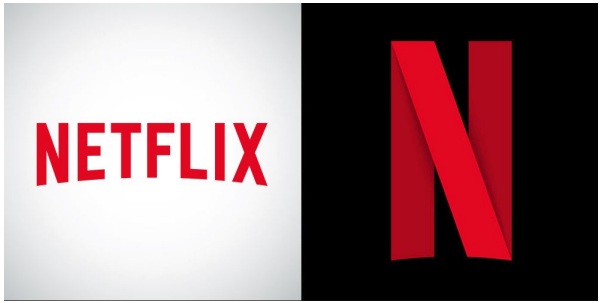
Quando usar?

- Aplicações em Tempo Real
- Uso de I/O intenso
- Chat
- Streaming
- Servir mais usuários utilizando menos hardware

Quando não usar?

- Uso intenso de CPU
- Manipulação de Vídeo
- Processamento de Imagens

Quem está usando node?



Instalando o Node

- `sudo apt-get update`
- `sudo apt-get install nodejs`
- `node --version`

Executando arquivos .js com Node

- Crie uma pasta chamada “simple_app”
- Entre na pasta com o comando “cd simple_app”
- Crie um arquivo com chamado “app .js"
    - Utilize o comando:

  `touch app.js`
- Abra o arquivo app.js com qualquer editor de texto
- Digite dentro do arquivo app.js:

  `console.log(“Hello world”)`
- Execute o arquivo com o comando:

  `node app.js`

Funções JavaScript

```
function nome(parametros){
   //código
}
```

Funções JavaScript

```
function soma(a, b){
   return a+b;
}
```

Funções JavaScript

```
var funSoma = function soma(a, b){
   return a+b;
}

funSoma(2,3)
```

Funções JavaScript - Arrow Function

- Sintaxe mais curta

```
(parameters) => {
  //código
}
```

Funções JavaScript - Arrow Function

- Sintaxe mais curta

```
var nome = (parameters) => {
  //código
}
```

Funções JavaScript

- Crie um novo arquivo chamado “app .js"
  - Utilize o comando:
    touch cambio.js
- Abra o arquivo app.js com qualquer editor de texto
- Digite dentro do arquivo cambio.js:

```
function converter_dolar_para_real(qtd) {
  var cotacao = 4.16;
  var conversao = qtd * cotacao;
  return conversao;
}

var resultado = converter_dolar_para_real(10);

console.log("Valor convertido: " + resultado);
```

- Execute o arquivo com o comando:
  node calc.js

Exercícios node e funções

1. Crie uma função sem parâmetros
2. Crie uma função com mais de um parâmetro
3. Crie uma arrow function
    a. Armazene em uma variável
    b. Execute a função

Node Module - fs (File System)

- Manipula Arquivos
  - Ler
  - Criar
  - Atualizar
  - Deletar
  - Renomear

Node Module - fs (File System)

- Ler arquivo
  - Crie uma arquivo chamado livros.txt
    - Use o comando “touch livros.txt”
    - Digite o nome de alguns livros
  - Crie um arquivo chamado app.js
    - Digite:

```
var fs = require('fs');

fs.readFile('livros.txt', 'utf8', function(err, data) {
        console.log(data);
});
```

  - Execute o comando “node app.js”

Exercício - fs (File System)

- Implementar
  - Ler
  - Criar
  - Atualizar
  - Deletar
  - Renomear

NPM

- Gerenciador de Pacotes do Node
- Node Package Manager

Instalando o NPM

- `sudo apt-get install npm`
- `apt-get install -y build-essential`

```
npm install <nome do módulo>
```

```
npm install request
```

- `Pasta “node_modules”`
- `package.json (--save )`

NPM - package.json

```
{
  "name": "app-generated",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "cookie-parser": "~1.4.4",
    "debug": "~2.6.9",
    "ejs": "~2.6.1",
    "express": "~4.16.1",
    "request": "~1.6.3",
    "morgan": "~1.9.1",
    "pg": "^7.15.1"
  }
}
```

Removendo módulos - NPM

- `sudo npm uninstall <nome do módulo>`
- Exemplo: `sudo npm uninstall request`

Jade Template Engine

```
!!! 5
html(lang="en")
  head
    title= pageTitle
    :javascript
      | if (foo) {
      |    bar()
      | }
  body
    h1 Jade - node template engine
    #container
      - if (youAreUsingJade)
         You are amazing
      - else
         Get on it!
         Get on it!
         Get on it!
         Get on it!
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Jade</title>
    <script type="text/javascript">
      //<![CDATA[
      if (foo) {
        bar()
      }
      //]]>
    </script>
  </head>
  <body>
    <h1>Jade - node template engine</h1>
    <div id="container">
      <p>You are amazing</p>
    </div>
  </body>
</html>
```

Outras Template engines

- **Pug**: Haml-inspired template engine (formerly Jade).
- **Haml.js**: Haml implementation.
- **EJS**: Embedded JavaScript template engine.
- **hbs**: Adapter for Handlebars.js, an extension of Mustache.js template engine.
- **Squirrelly**: Blazing-fast template engine that supports partials, helpers, custom tags, and caching. Not white-space sensitive, works with any language.
- **React**: Renders React components on the server. It renders static markup and does not support mounting those views on the client.
- **h4e**: Adapter for Hogan.js, with support for partials and layouts.
- **hulk-hogan**: Adapter for Twitter's Hogan.js (Mustache syntax), with support for Partials.
- **combyne.js**: A template engine that hopefully works the way you'd expect.
- **swig**: Fast, Django-like template engine.
- **Nunjucks**: Inspired by jinja/twig.
- **marko**: A fast and lightweight HTML-based templating engine that compiles templates to CommonJS modules and supports streaming, async rendering and custom tags. (Renders directly to the HTTP response stream).
- **whiskers**: Small, fast, mustachioed.
- **Blade**: HTML Template Compiler, inspired by Jade & Haml.
- **Haml-Coffee**: Haml templates where you can write inline CoffeeScript.
- **Webfiller**: Plain-html5 dual-side rendering, self-configuring routes, organized source tree, 100% js.
- **express-hbs**: Handlebars with layouts, partials and blocks for express 3 from Barc.
- **express-handlebars**: A Handlebars view engine for Express which doesn't suck.
- **express-views-dom**: A DOM view engine for Express.
- **rivets-server**: Render Rivets.js templates on the server.
- **Exbars**: A flexible Handlebars view engine for Express.
- **Liquidjs**: A Liquid engine implementation for both Node.js and browsers.
- **express-tl**: A template-literal engine implementation for Express.
- **vuexpress**: A Vue.js server side rendering engine for Express.js.
- **Twing**: First-class Twig engine for Node.js.

Prática 3 - módulos

- Crie uma pasta chamada “node_modulos”
- Dentro da pasta crie 3 arquivos
  - dividir.js, somar.js, main.js

Prática 3 - módulos

- No arquivo somar.js:

```
var somar = function soma(a,b){
        return a+b;
}
module.exports = somar;
```

- No arquivo dividir.js:

```
var dividir = function soma(a,b){
        return a/b;
}
module.exports = dividir;
```

Prática 3 - módulos

- No arquivo main.js:

```
var somafunc = require("./somar");
var divfunc = require("./dividir");

console.log(somafunc(10,20));
console.log(divfunc(10,2));
```

Servidor node básico

- No arquivo main.js:

```
var http = require('http');

http.createServer(function(req, res){
  res.end('Hello World!');
}).listen(8084);

console.log("Servidor rodando na porta 8084");
```

- Digite no navegador: http://localhost:8084/

Express

O que é Express?

Principais recursos

- Gerencia as requisições, rotas e URLs
- Renderização de HTML
- Configurações (Porta, Banco de Dados)

Instalando Express

- npm init -y
- npm install express --save
- Crie arquivo index.js
    - touch index.js

Criando aplicação Express

- Digite no arquivo index.js

```
var express = require('express');

var app = express();

app.get('/', function (req, res) {
  res.send('Hello World!');
});

app.listen(3000, function () {
  console.log('Example app listening on port 3000!');
});
```

Servindo páginas html

- Modifique o arquivo index.js

```
var express = require('express');

var app = express();
var path = __dirname + '/views/';


app.get('/', function (req, res) {
  res.sendFile(path + "index.html");
});

app.listen(3000, function () {
  console.log('Example app listening on port 3000!');
});
```

Servindo páginas html

- Crie uma pasta chamada “views”
- Crie o arquivo “index.html”
- No arquivo “index.html’ digite:

```
<!DOCTYPE html>
<html>
<body>

<h1>Express</h1>
<p>renderizando html</p>

</body>
</html>
```

Bootstrap

- Biblioteca de Componentes
  - Front-end
- Muito Popular
- Responsiva
- HTML, CSS e JavaScript
- Desenvolvimento Rápido
- Criado em 2011 pelo Twitter

Bootstrap - Configuração

Siga o roteiro

Bootstrap - Mais recursos

https://getbootstrap.com/docs/4.4/components/alerts/

Bootstrap - Navbar (Menu)

- Vá até o site: https://getbootstrap.com/docs/4.4/examples/starter-template/
- Clique com botão direto na página
    - Selecione inspecionar elemento
    - Procure o elemento: <nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
    - Copie o elemento (todo)
    - Cole dentro do body do seu arquivo “index.html”
    - Execute o comando “node index.html”
    - Observe a barra de navegação

Nodemon

- sudo npm install nodemon -g
- nodemon index.js

Deploy - Heroku

- Plataforma na nuvem
- Hospedagem de aplicações
- Plano Free
- Escalável

Heroku

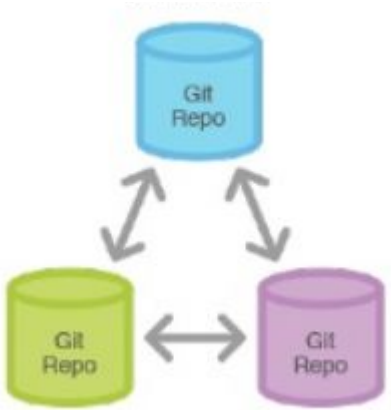
- Plataforma na nuvem
- Hospedagem de aplicações
- Plano Free
- Escalável

Git

- O que é?
  - Sistema de controle de versões
  - Distribuído
  - Usado principalmente no desenvolvimento de software
  - Mas pode ser usado para registrar o histórico de edições de qualquer tipo de arquivo.

Git

- O que é?
  - Sistema de controle de versões
  - Distribuído



Git

- O que é?
  - Sistema de controle de versões
  - Distribuído
  - Usado principalmente no desenvolvimento de software
  - Mas pode ser usado para registrar o histórico de edições de qualquer tipo de arquivo.

Git

- Surgimento:
  - Em Abril de 2005 durante o desenvolvimento do Kernel Linux
  - Ferramenta usado na época: Bitkeeper
  - Linus Torvalds decidiu criar uma solução própria

GitHub

- O que é?
  - Plataforma de hospedagem de código-fonte
  - Utiliza o Git.
  - Open Source / Privado
  - De qualquer lugar do mundo

GitHub

- O que é?
  - Plataforma de hospedagem de código-fonte
  - Utiliza o Git.

GitHub

- O que é?
  - Plataforma de hospedagem de código-fonte
  - Utiliza o Git.
  - Open Source / Privado
  - De qualquer lugar do mundo

GitHub

- Criem uma conta o GitHub

Git e GitHub - Instalação

- (Sudo)
- apt update
- apt install git
- git --version
- Informações do usuário
  - git config --global user.email "you@example.com"
  - git config --global user.name "Your Name"

Teste

- git clone https://github.com/iceberg20/curso-dev-web-node-express.git

Deploy

Seguir o roteiro do deploy

Referências

- Blog do Netflix
- https://nodejs.org/en/
- https://expressjs.com/
- https://getbootstrap.com/
- https://git-scm.com/
- https://www.heroku.com/
- https://www.w3schools.com/