



Desenvolver aplicações Backend para WEB

Prof. Renan Ponick

Lema

**"Nenhuma quantidade de ansiedade faz
qualquer diferença para qualquer coisa que vai
acontecer"**

Allan watts

Agenda

09/08/2023 - Banca

10/08/2023 - NodeJS e NPM - Configurando

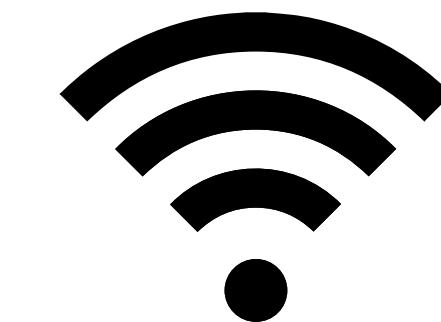
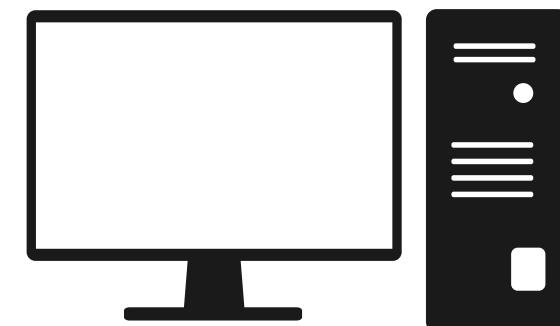
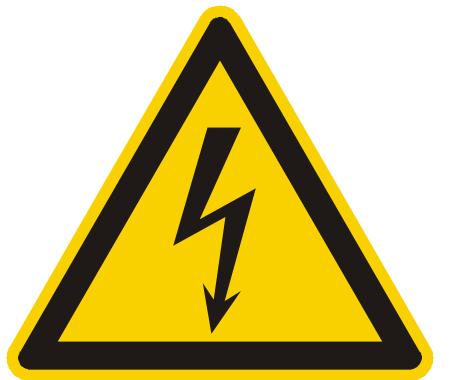
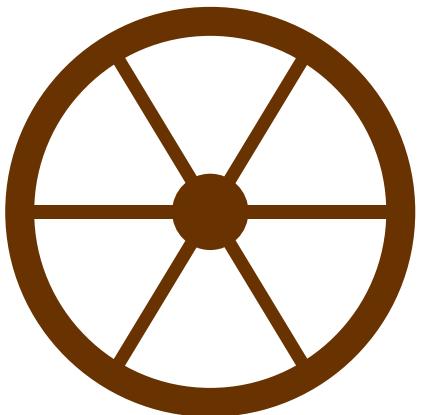
11/08/2023 - Inicialização

17/08/2023 - Exercícios

18/08/2023 - Exercícios

O começo

Boa parte das coisas que nós conhecemos hoje
surgiram de algum problema.





E o Git?

Os Problemas

- Preciso voltar o arquivo na versão que estava antes - rollback - Ctrl+Z;
- Deleção de arquivo sem a possibilidade de recuperação;
- Cópias e mais cópias de um arquivos:
 - tcc.docx;
 - tcc-final.docx;
 - tcc-agora-vai.docx;

A solução

Controle de Versão

Os sistemas de controle de versão são ferramentas de software que ajudam as equipes de software a gerenciar as alterações ao código-fonte ao longo do tempo;

BitKeeper

Ferramenta de versionamento que começou a exigir pagamento das outras empresas de software para que fosse usada...

O nascimento

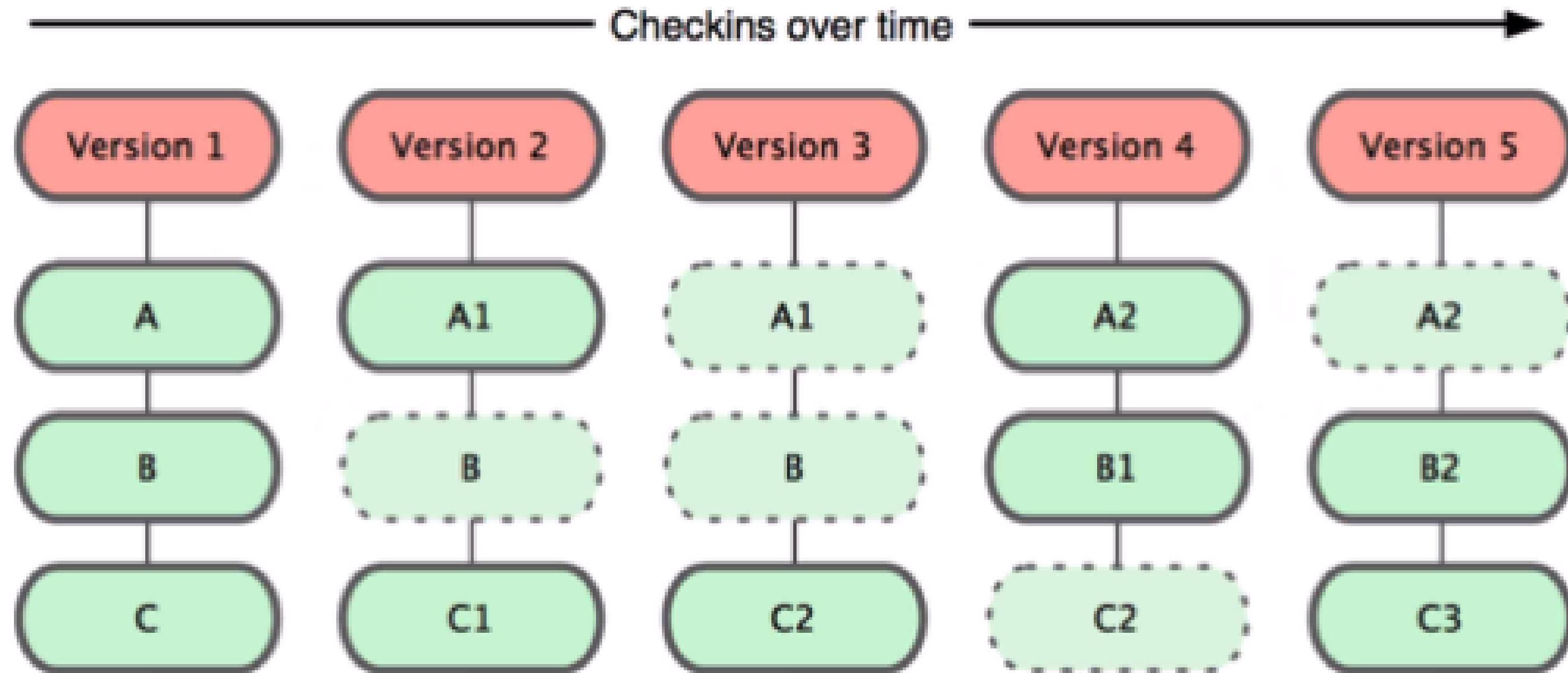
Descontente com as cobranças do BitKeeper, o dono
do Linux criou o Git.

Corrigindo várias deficiências do BitKeeper.

Git

O Git é uma ferramenta de versionamento local, que salva o estado de cada arquivo na hora do versionamento e caso o arquivo não sofra alteração, ele cria um link simbólico para o arquivo não editado.

Git





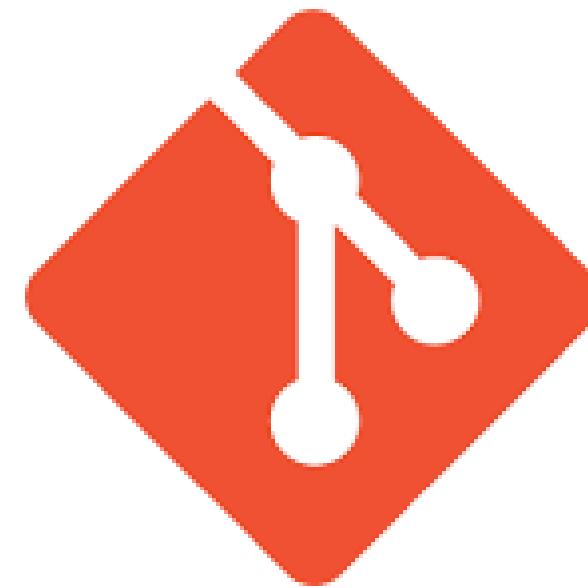
E o GitHub?

Os Problemas

- Como trabalhar em equipe, utilizando o controle do Git, para que tenhamos um controle de versão?
- Como compartilhar o conhecimento(o código fonte)?

GitHub

Serviço web compartilhado para projetos que utilizam o Git para versionamento. É um local de armazenamento de código.



git

Controle de Versão Local

!=



GitHub

Repositório remoto

Check Point



Quem veio primeiro?
Um poderia existir sem o outro?
Eles são os únicos?

Dia a dia



Para que, como usá-los?



Dia a Dia

O desenvolvimento de software possui uma característica forte quanto ao desenvolvimento paralelo(em conjunto) do software pela equipe de desenvolvedores.

EX: Login, Cadastro, BD

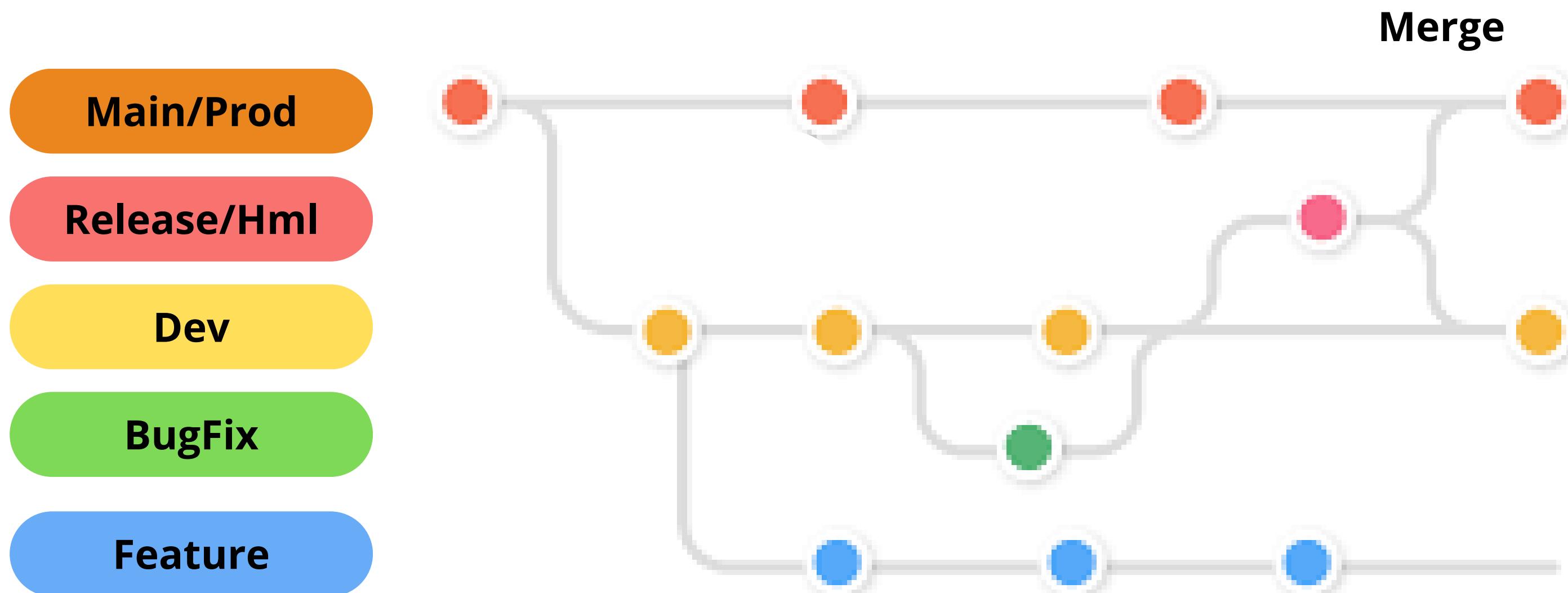


Branchs

Pesquisa rápida no google

É útil em situações nas quais você deseja adicionar um novo recurso ou corrigir um erro, gerando uma nova ramificação garantindo que o código instável não seja mesclado nos arquivos do projeto principal.

Equipe



Comandos

git status

git add <filename ou .>

git restore --staged <filename ou .>

git branch <branchname>

git checkout <branchname>

git checkout -b <branchname>

git commit -m "<description>"

git push

git branch -D <branchname>

git fetch

git pull

Boas práticas

NUNCA COMMITAR NA MAIN/MASTER/PROD

Para isso abrimos branchs:

feat: Algo relacionado com features que você adicionar;

fix: Algo para corrigir;

docs: Algo relacionado a documentações, README e afins;

style: Algo relacionado com estilização;

refactor: Algo relacionado com refatoração(refazer);

perf: Algo relacionado a performance;

test: Algo com testes;

chore: Algo para coisas relacionados a build, configs e afins. Exemplo: Use esse, seja atualizando a versão do pacote ou instalando novas dependências.

Inicializando Projeto

Terminal VSCode

Execute **npm init** e preencha as informações do seu projeto, para deixar tudo default basta dar enter e digitar yes no final, por exemplo:

```
● renan@ng0502:~/Documents/senac/JP/javascript$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (javascript)
version: (1.0.0)
description:
entry point: (exercicio1.js)
test command:
git repository: (https://github.com/renanponick/javascript.git)
keywords:
author:
license: (ISC)
About to write to /home/renan/Documents/senac/JP/javascript/package.json:

{
  "name": "javascript",
  "version": "1.0.0",
  "description": "0. Faça um programa (FUP) que escreva na tela \"Hello World!\\"",
  "main": "exercicio1.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/renanponick/javascript.git"
  },
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/renanponick/javascript/issues"
  },
  "homepage": "https://github.com/renanponick/javascript#readme"
}

Is this OK? (yes) yes
```

Arquivo de configuração

```
npm package.json ✘ ×  
npm package.json > ...  
1 {  
2   "name": "javascript",  
3   "version": "1.0.0",  
4   "description": "",  
5   "main": "index.js",  
6   ▷ Debug  
7   "scripts": {  
8     "test": "echo \"Error: no test specified\" && exit 1"  
9   },  
10  "repository": {  
11    "type": "git",  
12    "url": "git+https://github.com/renanponick/javascript.git"  
13  },  
14  "author": "",  
15  "license": "ISC",  
16  "bugs": {  
17    "url": "https://github.com/renanponick/javascript/issues"  
18  },  
19  "homepage": "https://github.com/renanponick/javascript#readme"  
20 }
```

Desafio

Imprima Hello world com node e js

Desafio 2

Imprima Hello world com node e js
Usando a lib express e metodo GET

Desafio

Imprima Hello world com node e js

Começando fácil

Crie um arquivo index.js

Dentro dele escreva `console.log('Hello World')`

Abra o terminal e execute node index.js

The screenshot shows a dark-themed interface of the Visual Studio Code (VS Code) code editor. On the left, the Explorer sidebar is open, displaying a file tree under the 'JAVASCRIPT' section. The tree includes 'docs', 'routes', 'index.js' (which is selected), 'package.json', and 'README.md'. In the center, the main workspace shows a single line of code in 'index.js': '1 console.log('hello World!')'. At the bottom, the Terminal tab is active, showing the command 'node index.js' followed by the output 'hello World!'.

Dificultando...

Imprima Hello world com node e js
Usando a **lib** express e **metodo GET**

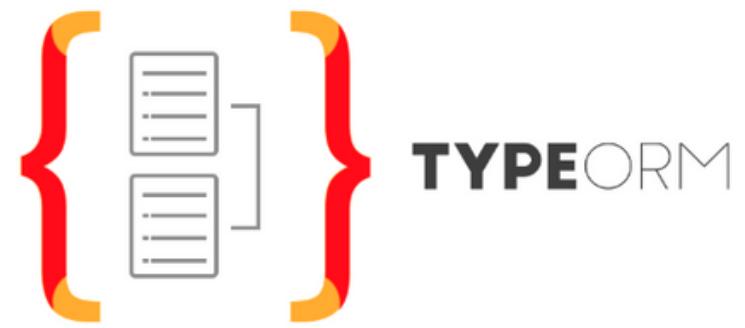
Aqui nós nos deparamos com duas coisas novas, LIB e Metodo GET.

Lib



Lib é uma abreviação comum para "library" (biblioteca), elas normalmente possuem um conjunto de funções, classes e métodos pré-criados que podem ser usados para realizar tarefas específicas de maneira mais eficiente e conveniente.

Lib



Essas bibliotecas encapsulam funcionalidades comuns e complexas, permitindo que os desenvolvedores reutilizem código em vez de começar do zero toda vez que precisarem realizar uma tarefa específica



Express

É um framework de aplicativo web para Node.js. Ele simplifica muito o processo de criação de aplicativos web e APIs (Interfaces de Programação de Aplicativos) usando Node.js, fornecendo uma estrutura robusta e flexível para lidar com várias tarefas relacionadas ao servidor e ao roteamento.



Roteamento

Dentre outras funcionalidades veremos o roteamento, onde o Express facilita a definição de rotas para diferentes URLs em seu aplicativo. Você pode configurar várias rotas para diferentes métodos HTTP (GET, POST, PUT, DELETE etc.) e definir como cada rota deve ser tratada.

Dificultando...

Instale a lib no projeto, como?

Abra o terminal e execute `npm i express` ou `npm install express`

```
renan@ng0502:~/Documents/senac/JP/javascript$ npm i express
added 58 packages, and audited 59 packages in 6s
8 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

Dificultando...

Modifique o arquivo index, para:

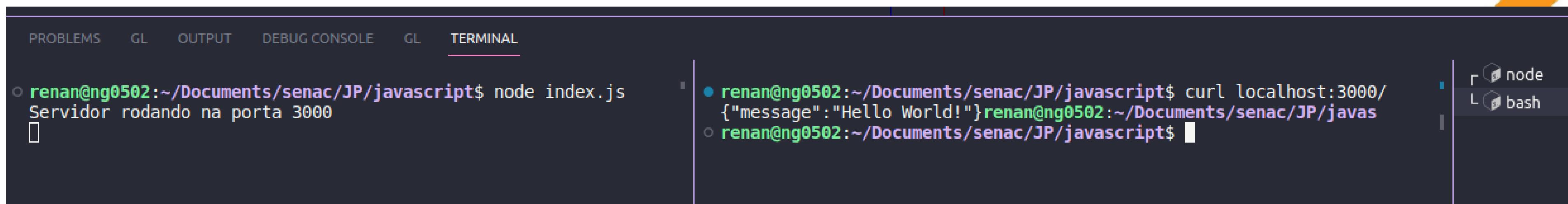
```
1 import express from "express"
2
3 const app = express();
4
5 app.get("/", (req, res) => {
6     res.json({ message: "Hello World!" });
7 }
8
9 app.listen(3000, () => {
10    console.log("Servidor rodando na porta 3000");
11 }
12
```

Dificultando...

Execute `node index.js`

```
○ renan@ng0502:~/Documents/senac/JP/javascript$ node index.js
Servidor rodando na porta 3000
```

Hello world não apareceu, quer dizer que esta errado?
Abra outro terminal e execute `curl localhost:3000/`



A screenshot of a terminal window from a code editor interface. The window has tabs at the top: PROBLEMS, GL, OUTPUT, DEBUG CONSOLE, GL, and TERMINAL. The TERMINAL tab is active, indicated by a blue underline. Below the tabs, there is a scroll bar. The terminal output shows two lines of text:

```
○ renan@ng0502:~/Documents/senac/JP/javascript$ node index.js
Servidor rodando na porta 3000
```

On the right side of the terminal window, there is a vertical toolbar with icons for node and bash.

Explicação

```
7 // Configura uma rota para o método HTTP GET na raiz ("/") do aplicativo.  
8 app.get("/", (req, res) => {  
9     // Retorna uma resposta JSON contendo uma mensagem "Hello World!".  
10    res.json({ message: "Hello World!" });  
11}  
12  
13 // Inicia o servidor Express na porta 3000 e define uma função de retorno de chamada  
14 app.listen(3000, () => {  
15     // Exibe uma mensagem no console quando o servidor estiver ativo.  
16     console.log("Servidor rodando na porta 3000");  
17});  
18
```

Rotas HTTP/HTTPS

- Get
 - Utilizada para recuperar uma lista ou um unico recurso.
- Post
 - Utilizada para criar um novo recurso.
- Put
 - Utilizada para atualizar um recurso
- Delete
 - Utilizada para excluir um recurso.

Quer melhorar seu código? Pesquise por padrão REST

Exemplo básico

```
6  
7  app.get("/", (req, res) => {});  
8  
9  app.get("/:id", (req, res) => {});  
10  
11 app.post("/", (req, res) => {});  
12  
13 app.put("/:id", (req, res) => {});  
14  
15 app.delete("/:id", (req, res) => {});  
16
```

Exemplo melhorado

```
6  
7   app.get("/api/pessoa/", (req, res) => {});  
8  
9   app.get("/api/pessoa/:id", (req, res) => {});  
10  
11  app.post("/api/pessoa/", (req, res) => {});  
12  
13  app.put("/api/pessoa/:id", (req, res) => {});  
14  
15  app.delete("/api/pessoa/:id", (req, res) => {});  
16
```



Tudo certo até aqui?



Desafio

Com base no código a seguir:

```
11 app.get("/api/pessoa/:id", (req, res) => {  
12     res.json({ message: "Hello World!" });  
13 } );  
14
```

Exiba o id no lugar de hello world

Parametros

Parametros

Query Parameters (Parâmetros de Consulta)
Route Parameters (Parâmetros de Rota)

* Da pra passar mais info no header da requisição, mas não vem ao caso agora...

Query Params

Esses parâmetros são incluídos na URL após o caractere de interrogação (?). Eles são usados para enviar informações adicionais para a solicitação. No Express, você pode acessar esses parâmetros usando o objeto req.query.

Exemplo

Ajuste o código para o seguinte:

```
11 app.get("/api/pessoa/:id", (req, res) => {
12   const nome = req.query.nome;
13
14   res.json({ message: `Você passou um queryParam: ${nome}` });
15 }
16
```

Realize a seguinte consulta

curl localhost:3000/api/pessoa/123?nome=joao

```
● renan@ng0502:~/Documents/senac/JP/javascript$ curl localhost:3000/api/pessoa/123?nome=joao
○ {"message": "Você passou um queryParam: joao"} renan@ng0502:~/Documents/senac/JP/javascript$ █
```

Route Params

Esses parâmetros são definidos na própria rota como parte da URL. Eles são usados para capturar valores variáveis da URL e são acessados usando req.params.

Route Params

Ajuste o código para o seguinte:

```
11 app.get("/api/pessoa/:id", (req, res) => {
12   const nome = req.query.nome;
13   const pessoaId = req.params.id;
14
15   res.json({
16     message: `Você passou um queryParam: ${nome} e um routParam: ${pessoaId}`
17   });
18 });


```

Realize a seguinte consulta

curl localhost:3000/api/pessoa/123?nome=joao

```
● renan@ng0502:~/Documents/senac/JP/javascript$ curl localhost:3000/api/pessoa/123?nome=joao
{"message": "Você passou um queryParam: joao e um routParam: 123"}renan@ng0502:~/Documents/senac
○ $
```



Até aqui tranquilo?



Exercícios

A lista antiga de exercícios do C# vai voltar contudo!



Porém segura ansiedade, realize somente os 5 primeiros exercícios da lista, utilizando o query params para passar as informações para o exercício.

Exercícios

Depois pesquise sobre nodemon e adicione um script no package.json

Pesquise sobre Postman (e crie uma conta)

Desafio

Transforme os métodos Gets dos exercícios anteriores em Post...