Parada Tech da Comunidade

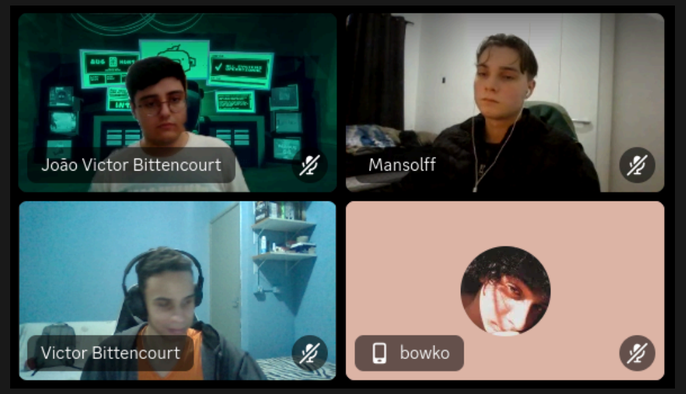
Iniciação em Front-End - HTML/CSS e JS

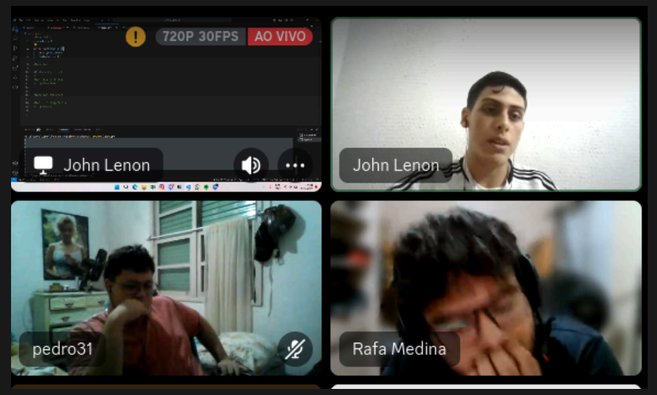
Criado por Victor Bittencourt, trilha de Python

O que é a

Parada Tech da Comunidade?

É um evento dos alunos que tenta simular as mesmas *Paradas Tech* que tivemos no decorrer do estudo assíncrono.





Trazemos introduções, resumos e tentamos tirar dúvidas de assuntos relacionados a desenvolvimento num geral.

Ferramentas Utilizadas

VSCode

Com liveserver

ou

OneCompiler

Alunos Instrutores

Julio Quevedo

Victor Bittencourt

Não possuímos nenhum vínculo empregatício com o Instituto Caldeira, somos somente alunos ensinando outros alunos;

Who am I?

Julio Quevedo

Porto Alegre, RS
Programador Back-End
Entusiasta em IA

julioquevdo

julioquevdo

Não possuímos nenhum vínculo empregatício com o Instituto Caldeira, somos somente alunos ensinando outros alunos;

Who I am?

Victor Bittencourt

20 anos, de Jundiaí, SP
em Gravataí, RS
Programador Fullstack

Atualmente estudo:
> React;
> TypeScript;
> Ruby;

VicourtBitt

vicourtbitt

Não possuímos nenhum vínculo empregatício com o Instituto Caldeira, somos somente alunos ensinando outros alunos;

Iniciação em Front-End

HyperText Markup Language

O HTML é uma linguagem de marcação (*) utilizada para estruturar layouts de páginas no desenvolvimento web.

A estrutura do HTML é baseada em uma sintaxe que utiliza tags (<h1>) para definir elementos.

```html
<label>
    <select>
        <option>Selecione</option>
        <option>Masculino</option>
        <option>Feminino</option>
        <option>Não-Binário</option>
    </select>
</label>
```

Estrutura de uma Tag em HTML

Abertura da Tag

Atributos da Tag

```
<div class="container" id="banner">
    <!--
        Estruturas filhas vem aqui!
    -->
</div>
```

Fechamento da Tag

Semanticidade das Tags HTML

Utiliza tags semânticas

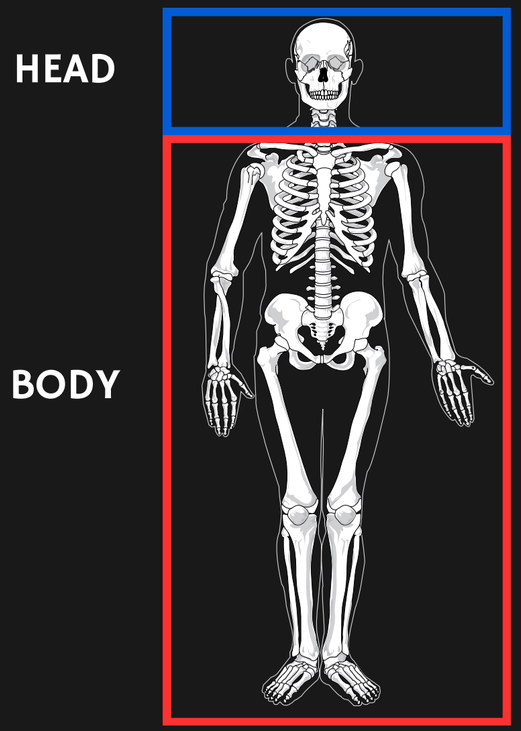
```html
<!-- Corpo do Site -->
<body>

    <!-- Cabeçalho -->
    <header>
        <!-- Conteudos Cabeçalho -->
    </header>

        <!-- Principal -->
        <main>
            <!-- Conteudos Principal -->
        </main>

    <!-- Rodapé -->
    <footer>
        <!-- Conteudos Rodapé -->
    </footer>
</body>
```

Não utiliza tags semânticas

```html
<!-- Corpo do Site -->
<body>

    <!-- Cabeçalho -->
    <div>
        <!-- Conteudos Cabeçalho -->
    </div>

        <!-- Principal -->
        <div>
            <!-- Conteudos Principal -->
        </div>

    <!-- Rodapé -->
    <div>
        <!-- Conteudos Rodapé -->
    </div>
</body>
```

Estrutura do Documento HTML

```html
<!-- Referência da versão do HTML -->
<!DOCTYPE html>
<!-- Abertura Script HTML e Linguagem Utilizada/Referenciada -->
<html lang="en">
    <!-- Header/Cabeça do Site -->
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Document</title>
    </head>

    <!-- Corpo do Site -->
    <body>
        <!-- Conteúdos do Site -->
    </body>
</html>
```



Como Vincular o HTML/CSS

REL representa o tipo de relação
TYPE representa a tipagem da relação
HREF representa o local onde o arquivo está

```html
<!-- Header/Cabeça do Site -->
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" type="text/css" href="styles.css">

    <title>Document</title>
</head>
```

Cascading Style Sheets

CSS Reset >

HTML Elem >

Classe CSS >

```
* {
    margin: 0px;
    padding: 0px;
    border: 0px;
}

body {
    width: 100vw;
    height: 100vh;
}

.container {
    border: 1px solid black;
    width: 200px;
    height: 200px;
}
```

O CSS é uma linguagem de estilização (*) utilizada em conjunto com o HTML para dar mais identidade visual para a página. A estrutura do CSS se baseia em classes, ID’s, atributos e elementos do HTML (assim como também, estados dos mesmos).

Como criar estilos?

```
/* Para criar ID's utilizamos a '#' no
inicio, ID's são únicos, devem ser usados
uma única vez Em determinada página (JS*)*/
#novo-id {
    /* estilizações do ID */
    border: 2px solid green;
    background: #c6c6c6;
    height: 100px;
    width: 100px;
}
```

```
/* Para criar classes utilizamos o '.'
no inicio, classes podem e devem ser
utilizadas em multiplos locais */
.nova-classe {
    /* estilizações da classe */
    border: 1px solid black;
    box-sizing: border-box;
    display: flex;
    justify-content: center;
    align-items: center;
}
```

height = altura
width = largura
display = exibição
background = 'fundo'
border = borda

```
/* Também podemos alterar a estilização dos
elementos HTML, o que afetará todos os elementos
daquele tipo em vez de somente afetar os que
possuem determinada classe */
input {
    /* estilização do elemento */
    height: 40px;
    width: 360px;
    border: 2px inset black
}
```

As estilizações possuem diversos tipos de atributos, alguns sendo mais restritos em questão de regras de aplicação e outros muito mais comuns quando se trata de implementação.

Estilização de Estado?

Estilização de estado é, quando capturamos alguma interação específica suportada pelo CSS em relação a algum elemento HTML, como:

- Passar o mouse;
- Ter posição par ou impar;
- Ter sido/estar sendo clicado;

Caso mouse por cima >

Caso mouse por cima >

```
.state-example {
    height: 80px;
    width: 80px;
    border: 2px solid black;
    background: rgba(0, 0, 0, 0.25);
    transition: 500ms;

    &:hover {
        transform: scale(1.1)
    }
}

.cabecalho, .rodape {
  height: 100px;
  width: 100%;
  background: #b6b6b6;
  transition: 500ms;

  &:hover {
    background: #c6d6c6;
  }
}
```

Como atribuir uma classe/id?

```html
<!-- Corpo do site -->
<body class='align-center'>
    <!-- Cabeçalho -->
    <header class="cabecalho" id="headerBox">
    </header>

        <!-- Principal -->
        <main class="corpo-pagina align-center">
            <div class='state-example'>
                teste
            </div>
        </main>

    <!-- Rodapé -->
    <footer class="rodape"></footer>

    <script src='script.js'></script>
</body>
```

Precisamos ir ao elemento e adicionar o respectivo atributo, se for uma classe, utilizamos o atributo class e depois atribuimos a classe criada. Para ID's é muito semelhante, pois a única mudança é usar o atributo id.

PS: durante a atribuição, não é necessário utilizar "." nem "#"

1° Convidado

Marcus Garim

Art Director e Brand Designer

marcusgarim

marcusgarim

Não possuímos nenhum vínculo empregatício com o Instituto Caldeira, somos somente alunos ensinando outros alunos;

Como vincular um scripts JS

A TAG script serve para códigos JS
SRC serve para referenciar o local do JS
É colocado antes do fechamento do body

```
        <!-- Rodapé -->
        <footer class="rodape"></footer>

        <script src='script.js'></script>
    </body>
</html>
```

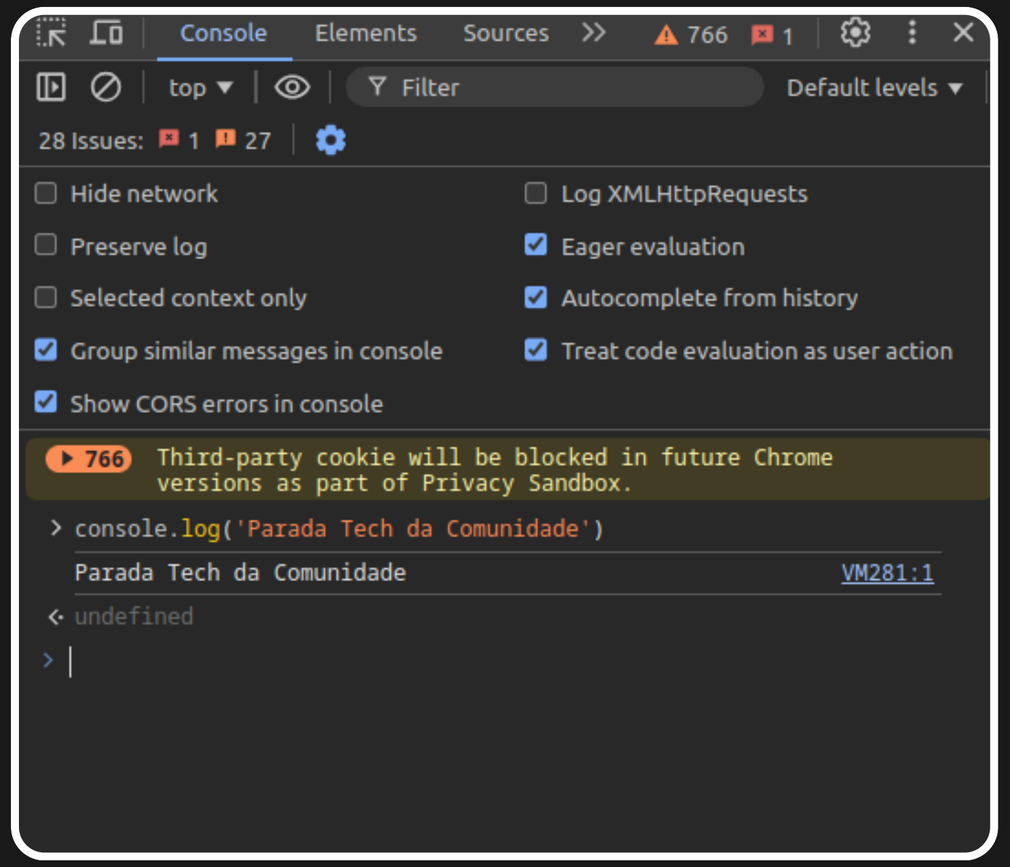
Javascript

O Javascript é uma linguagem de programação com foco no desenvolvimento web, servindo para interagir diretamente com eventos, validações e ocorrências client-side.

Os navegadores hoje em dia possuem engines que suportam o Javascript nativamente, mas ainda assim, deve-se tomar cuidado, pois nem toda nova funcionalidade serve no momento que saí.

```
document.addEventListener('DOMContentLoaded', (e) => {
    const formLogin = document.getElementById('formulary-login')
    const loginEmail = document.getElementById('loginEmail')
    const loginPass = document.getElementById('loginPassword')
    const dialog = document.querySelector('dialog')
    const btnClose = document.getElementById('closeModal')

    // PAY ATENTION TO THE LOGIN FORMULARY, THEN SEND ALL THE
    // INFORMATIONS INTO THE STORAGE FOR A WHILE
    formLogin.addEventListener('submit', async (e) => {
        e.preventDefault()
        const emailValue = loginEmail.value
        const passValue = loginPass.value

        // PUSH THE VALUES
        setIntoSessionStorage('loginInfo', {
            email : emailValue,
            password : passValue
        })

        const state = await validateLogin(sessionStorage.getItem('loginInfo'))
        if (state) {
            setIntoSessionStorage('userId', state)
            window.location.pathname = '/login-profile.html'
        } else {
            dialog.showModal()
        }
    })
```

Engine JS Nativa do Chrome



O Chrome possui uma engine nativa para interpretar Javascript, sendo ela a V8. O que significa que, você, pode rodar comandos Javascript diretamente do console do próprio Chrome.

Se não souber onde fica o console, você pode apertar **Ctrl + Shift + i** para abrir o painel das DevTools.
Ou clique com o botão direito e em inspecionar.

Versatilidade do Javascript

Por ser uma linguagem tão versátil, o JavaScript possuí uma quantidade de bibliotecas/frameworks imensa, assim como o seu superset (TypeScript) e por mais que seja muito presente no Front-End, ela também faz sua aparição no Back-End, por meio do Node e de seus módulos específicos para desenvolvimento.

ID's e Classes no Javascript?

Sim! Eles também estão no Javascript, mas com um propósito mais específico, para servirem como referências de seleção.

Se vários inputs (entradas de texto) possuirem a mesma classe, podemos selecionar todas de uma vez, ou, se um input possuir um id (que é único), podemos pegar só ele.

```
const pegarEspecifico = document.querySelector('#id .classe')
const pegaId = document.getElementById('id')
const pegaClasse = document.getElementsByClassName('classes')
const pegaElementos = document.getElementsByTagName('elemento')
```

ID's e Classes no Javascript?

Sim! Eles também estão no Javascript, mas com um propósito mais específico, para servirem como referências de seleção.

Se vários inputs (entradas de texto) possuirem a mesma classe, podemos selecionar todas de uma vez, ou, se um input possuir um id (que é único), podemos pegar só ele.

```
const pegarEspecifico = document.querySelector('#id .classe')
const pegaId = document.getElementById('id')
const pegaClasse = document.getElementsByClassName('classes')
const pegaElementos = document.getElementsByTagName('elemento')
```

2° Convidado

Gabriel Mansolff

Desenvolvedor Fullstack +3 anos

Gimieski

Não possuímos nenhum vínculo empregatício com o Instituto Caldeira, somos somente alunos ensinando outros alunos;

Projeto Prático

Chegamos ao fim!

Deixe seu feedback ou sugestão!