

JS PRO

Próximos passos:

- ES6
- Let statement [1-3]
- Const statement [4]
- Template Literals [5-9]
- Arrow function [10-15]
- Spread Operator [16-21]
- DOM [22-24]
- Eventos [25]
- Projeto 1 [26]
- Requisições HTTP [27]
- Projeto 2 [28]

LET

let permite criar variáveis com escopo de bloco

```
let myVar = 'Some value'  
let anyNumber = Math.random()  
let iAmTrue = true  
let emptyObj = {}  
let emptyArray = []
```

LET

Então qual a diferença pro var?

Loading

LET

Lembra do escopo em bloco?

Loading

CONST

Declarar uma **const** cria um valor que é uma referência de **somente leitura**.

Ela pode ser tanto global ou local a função ou bloco em que foi definida.

Dessa forma ela deve ser **obrigatoriamente** inicializada.

CONST

Loading

Template Literals

Template literals são usadas com as aspas(`` ``) ao invés de `' '` ou `" "`

Elas podem conter expressões JS dentro.

Todas as partes, incluindo o resultado dessa(s) expressão(ões) são concatenadas e viram uma única string

Template Literals

Loading

Template Literals

Loading

Template Literals

Loading

Arrow functions

Arrow function é uma maneira diferente de se declarar uma Function Expression.

Isso implica em uma grande mudança: o escopo do 'this'.
Representada pelo sinal de '=>'

Arrow functions

Loading

Arrow functions

ES6

- menos código
- sem a necessidade da palavra function
- () são opcionais às vezes
- retorno implícito sem as chaves {}

Arrow functions

Diferentes maneiras declarar arrow functions

Arrow functions

1 parâmetro

Loading

Arrow functions

2 ou mais parâmetros

Loading

Spread operator

A sintaxe de propagação (Spread) permite que um objeto iterável, como um array ou string, seja expandida em locais onde zero ou mais argumentos (para chamadas de função) ou elementos (para literais de array) sejam esperados ou uma expressão de objeto seja expandida em locais onde zero ou mais pares de chave-valor (para literais de objeto) são esperados.

(MDN)

Spread Operator – Arrays

Loading

Spread Operator – Arrays

Loading

Spread Operator – Arrays

ES6

Loading

Spread Operator – Functions

Loading

Spread Operator – Functions

ES6

Loading

DOM (Document Object Model)

O Modelo de Objeto de Documento (DOM) é uma interface de programação para documentos HTML, XML e SVG . Ele fornece uma representação estruturada do documento como uma árvore. O DOM define métodos que permitem acesso à árvore, para que eles possam alterar a estrutura, estilo e conteúdo do documento. O

DOM fornece uma representação do documento como um grupo estruturado de nós e objetos, possuindo várias propriedades e métodos. Os nós também podem ter manipuladores de eventos que lhe são inerentes, e uma vez que um evento é acionado, os manipuladores de eventos são executados. Essencialmente, ele conecta páginas web a scripts ou linguagens de programação.

(MDN)

DOM – Window Object

```
console.log(window.innerWidth)
console.log(window.outerWidth)
console.log(window.innerHeight)
console.log(window.outerHeight)
console.log(window)
localStorage e sessionStorage
document
location
```


DOM – Document

- title
- children
- como modificar o próprio DOM
- adicionando estilos
- pegando os elementos do DOM
- criando elementos com JS
- inserindo novos elementos no DOM
- deletar elementos

DOM

```
const title = document.title
const children = document.children
document.title = 'Escolha um título'
const h1 = `
  <h1>
    Olá Target
  </h1>
`

const app = document.getElementById('app')
app.innerHTML = h1
```

Eventos

Vamos para o [codesandbox](#) :)

- `onclick`
- `addEventListener`
- `target`

Projeto 1 – TODO App

Funcionalidades:

- Criar um TODO
- Deletar um TODO
- Limpar a lista
- Buscar TODO
- Remover espaços em branco das tarefas
- Limpar formulário após submissão
- Não permitir tarefa em branco
- Persistir dados usando localStorage (Bônus)

HTTP Requests

Vamos para o codesandbox :)

- Tipos de requisições
- GET / POST
- Fetch API

Projeto 2 – App de Clima