Instituto **Janela** para o **Mundo**

Let's code

JS

Prof.: José Santos

O QUE É ?

- JavaScript, ou JS é uma linguagem de programação, sendo a principal linguagem do frontend. Ela é responsável pelo dinamismo de uma página, ou seja, promove a interação com o usuário. Um exemplo clássico de JavaScript são os menus mobile, quando clicamos nele é feito uma interação com o usuário (no caso abrindo os itens do menu).

INICIANDO...

- JavaScript é escrito em arquivos .js. Para criar um documento JavaScript é fácil, basta salvar o arquivo como script.js (lembro que você pode nomeá-lo como quiser, mas não esqueça do .js) dessa forma teremos nosso documento semipronto para o uso

INICIANDO...

- Agora iremos criar a conexão do JavaScript com nosso arquivo HTML. Antes do fechamento da tag <body> de seu documento HTML, vamos adicionar uma tag <script> para conectá-los. Isso é, na tag <script> atribuímos o atributo src="" (que vem do inglês source), esse atributo é responsável em fazer a conexão com outros arquivos.

VARIÁVEIS

- Variáveis são lugares na memória onde colocamos valores para podermos trabalhar com eles posteriormente. As variáveis são um dos fatores para mantermos o código dinâmico, fácil de ser lido e compreendido Isso é, uma vez armazenado um valor em uma memória podemos utilizar seu valor ao longo do nosso código.

VARIÁVEIS

- No JavaScript não há necessidade de declarar o tipo da variável, mas isso não significa que ela não tem tipo. Isso torna o JavaScript uma linguagem de tipagem dinâmica, o tipo é inferido pelo valor do dado e a checagem (type checking) é feita em tempo de execução (runtime). Então se você tiver uma variável chamado “nome” com o valor de “josé” e ao longo do nosso código mudar seu valor para 10, para o JavaScript isso está certo. Isso tem seus lados bons e ruins

VARIÁVEIS

- Por exemplo, essa é a declaração de uma variável:
- const nomeVariavel = “valor”
- nomeVariavel = “valor1”

VARIÁVEIS

- Bem, para declarar uma variável temos três opções: var, let e const.
- Uso do var: Uma variável declarada com var possui o que chamamos de escopo de função. Isso significa que se criarmos uma variável deste tipo dentro de uma função, você pode modificar em qualquer parte desta função, mesmo se criar outra função dentro dela
- Uso do let: Diferente de var, declarar como let leva em conta o bloco de código onde foi declarada. Isso significa que se a declararmos dentro de um uma função, ela será “enxergada” apenas dentro deste escopo.

VARIÁVEIS

- Uso do const: Uma variável const é usada para definir uma constante. Diferente de var e let, as variáveis de const não podem ser atualizadas nem declaradas novamente.

TIPOS DE VARIÁVEIS

- String
- Number
- Boolean
- Object
- Undefined
- Null

OPERADORES

Operador	Descrição
Módulo (%)	Retorna o inteiro restante da divisão dos dois operadores.
Incremento (++)	Adiciona um ao seu operando.
Decremento (--)	Subtrai um de seu operando.
Negação (-)	Retorna a negação de seu operando.
Adição (+)	Tenta converter o operando em um número.

OPERADORES DE COMPARAÇÃO

Operador	Descrição
Igual (==)	Retorna true caso os operandos sejam iguais.
Não igual (!=)	Retorna true caso os operandos não sejam iguais.
Estritamente igual (===)	Retorna true caso os operandos sejam iguais e do mesmo tipo.
Estritamente não igual (!==)	Retorna true caso os operandos não sejam iguais e do mesmo tipo.
Maior que (>)	Retorna true caso o operando da esquerda seja maior que o da direita.
Maior que ou igual (>=)	Retorna true caso o operando da esquerda seja maior ou igual que o da direita.
Menor que (<)	Retorna true caso o operando da esquerda seja menor que o da direita.
Menor que ou igual (<=)	Retorna true caso o operando da esquerda seja menor ou igual que o da direita.

CONDICIONAL

- IF
- ELSE
- ELSE IF
- Ternário
- SWITCH CASE

FUNCTION

- Uma função, ou function é um conjunto de instruções que executa uma tarefa.
- Imagine que você tenha duas tarefas, uma é executar a soma de dois números e a outra é mostrar o nome do usuário no sistema. Percebeu que essas tarefas são duas funções diferentes. Imagine essas duas tarefas espalhadas em nosso código, iria atrapalhar muito, mas com uma função teremos um agrupamento de código que faz uma determinada tarefa em nossa aplicação.

FUNCTION

- A definição da função (também chamada de declaração de função) consiste no uso da palavra-chave function, seguida por:
  - Nome da função
  - Lista de argumentos para a função, entre parênteses e separados por vírgulas.
  - Declarações JavaScript que definem a função, entre chaves { }.
  - Por exemplo, o código a seguir define uma função simples chamada somar:

FUNCTION

- Veja que a função de somar recebe um argumento chamado numero. Isso significa que a nossa função está esperando o valor de numero (que no exemplo é 10) para multiplicar por si mesmo (numero * numero). A declaração return especifica o valor retornado pela função.

```
1  function somar(numero) {
2   return numero * numero;
3  }
4
5  console.log(somar(10));
```

EXERCÍCIO

- Fazer uma calculadora usando JavaScript e HTML
- Fazer um contador de clicks e ter a opção reset
- Fazer um incremento e decremento do valor

DESAFIO

ESTRUTURA DE REPETIÇÃO

- Array
- For
- While

EXERCÍCIOS

- 1 - Exibir os 10 primeiros números pares
- 2 - Exibir os 10 primeiros números impares
- 3 - Exibir os 10 primeiros números primos
- 4 - imprimir o menor numero do array
- 5 - imprimir o maior numero do array

EXERCÍCIOS

- 6 - imprimir todos os números menores que 7 do array
- 7 - imprimir todos os números maiores que 7 do array
- 8 - imprimir a soma total do array
- 9 - imprimir a media aritmética do array
- 10 - imprimir o fatorial de cada numero do array > 5! = 5*4*3*2*1 = 120 | 6 * 5! = 720

EXERCÍCIOS

- 11 - imprimir a enésima potenciação de um numero
- 12 - imprimir a sequencia de Fibonacci até o maior numero do array (Math.max)

  0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144
- 13 - verificar de uma dada string é palíndromo

  ex: banana ≠ ananab | arara = arara
- 14 - imprima todas as permutações dos caracteres de uma cadeia de caracteres (string).

  Exemplo: as permutações da cadeia ABC são: ABC, ACB, BAC, BCA, CAB, CBA.
- 15 - Ordenar o array, imprimir na ordem crescente e decrescente.

MÉTODOS DE ARRAY

- Concat
- Join
- Push
- Pop
- Shift
- Slice
- Splice
- Reverse

MÉTODOS DE ARRAY - CONCAT

- Serve para concatenar dois arrays, no exemplo vamos concatenar o array front com o array back.

```
1  var front = ["html", "css", "javascript"];
2  var back = ["java", "php", "node"];
3
4  front = front.concat(back);
5  console.log(front);
6  // Resultado será um array com os elementos dois arrays
```

MÉTODOS DE ARRAY - JOIN

- O método “join()” puxa elementos de um array e lista no formato de string.

```
var front = ["html", "css", "javascript"];
front = front.join("-");

console.log(front);
// Resultado será as propriedades do array
// separar com um traço
```

MÉTODOS DE ARRAY - PUSH

- O método “push()” serve para adicionarmos elementos no final do array.

```
1 var front = ["html", "css", "javascript"];
2 front.push("react");
3
4 console.log(front);
5 // Resultado será um novo elemento no final do array
```

MÉTODOS DE ARRAY - POP

- O método “pop()” remove o último elemento de um array.

```
1  var front = ["html", "css", "javascript"];
2  front.pop();
3
4  console.log(front);
5  // Resultado será a remoção do último elemento do array
```

MÉTODOS DE ARRAY - SHIFT

- O método “shift()” remove o primeiro elemento do array.

```
1  var front = ["html", "css", "javascript"];
2  front.shift();
3
4  console.log(front);
5  // Resultado será a remoção do primeiro elemento do array
```

MÉTODOS DE ARRAY - REVERSE

- O método “reverse()” inverte a ordem dos elementos do array.

```
var front = ["html", "css", "javascript"];
front.reverse();

console.log(front);
// Resultado será reversão dos elementos do array
```

D O M

- onload: Disparado quando documento é carregado.
- onunload: Disparado quando documento é descarregado de janela ou de frame.
- onsubmit: Disparado quando formulário é submetido.
- onreset: Disparado quando formulário é “limpado” via botão reset.
- onselect: Disparado quando texto é selecionado numa área de entrada de texto.
- onchange: Disparado quando elemento perde o foco e foi modificado.
- onclick: Disparado quando botão de formulário é selecionado via click do mouse.

D O M

- onfocus: Disparado quando o elemento recebe foco: clicando o mouse dentro do elemento ou entrando no mesmo via Tab.
- ondblclick: Disparado quando ocorre um click duplo do mouse.
- onmousedown: Disparado quando mouse é pressionado enquanto está sobre um elemento.
- onmouseup: Disparado quando mouse é despressionado.
- onmouseover: Disparado quando cursor do mouse é movido sobre elemento.

D O M

- onmouseout: Disparado quando mouse é movido fora do elemento onde estava.
- onkeydown: Disparado quando tecla é pressionada.

SELETORES HTML

- getElementById: Ele método retorna um elemento correspondente ao id passado como parâmetro. Veja o exemplo abaixo:

```
1 document.getElementById("ebook");
```

SELETORES HTML

- getElementByClassName: Como o próprio nome já diz, essa função vai retornar os elementos que possuem uma mesma classe passada. Veja o exemplo abaixo:

```
1 document.getElementsByClassName("ebook");
```

SELETORES HTML

- querySelector: Diferente dos outros métodos, este método utiliza os indicadores para selecionar os elementos: id(#), class(.) ou o próprio nome da tag do elemento. Veja o exemplo abaixo:

```
1  let teste1 = document.querySelector("#souId");
2  let teste2 = document.querySelector(".souClass");
3  let teste3 = document.querySelector("div");
```

SELETORES HTML

- querySelectorAll: Este método é similar ao método querySelector que também utiliza os seletores do CSS, porém retorna uma NodeList com todos os elementos que correspondem ao seletor criado. Veja o exemplo abaixo:

```
let teste = document.querySelectorAll("div");
// Irá retornar todas as divs da página
```

SELETORES HTML

- querySelectorAll: Este método é similar ao método querySelector que também utiliza os seletores do CSS, porém retorna uma NodeList com todos os elementos que correspondem ao seletor criado. Veja o exemplo abaixo:

```
let teste = document.querySelectorAll("div");
// Irá retornar todas as divs da página
```

INTERAÇÃO DO USUÁRIO

- addEventListener: Conhecido como evento de escuta, permite configurar funções a serem chamadas quando um evento acontece, que em nosso exemplo, quando um usuário clica em um botão. No exemplo a seguir mostra que o elemento “nomeElemento” está sendo escutado caso o usuário faça um click.

```
1  alvo.addEventListener(evento, função);
```

INTERAÇÃO DO USUÁRIO

- alvo: É o elemento HTML ao qual você deseja adicionar os eventos.
- evento: É onde você especifica qual é o tipo de evento que irá disparar a ação/função. Uma coisa interessante no evento é que diferente do JavaScript no HTML, ele não tem o “on” dos eventos como onclick, onchange, onblur...
- função: Específica a função a ser executada quando o evento é detectado. Em nossos exemplos foram utilizados a sintaxe arrow function que deixa a expressão da nossa função mais curta e anônima. Mas você pode simplesmente aplicar uma outra função código no lugar.

```
1  alvo.addEventListener(evento, função);
```

INTERAÇÃO DO USUÁRIO - EXEMPLO

```
1  var nomeElemento = document.getElementById("div");
2
3  nomeElemento.addEventListener("click", () ⇒ {
4    ...
5  });
```

```
1  var button = document.querySelector("button");
2
3  button.addEventListener("click", () ⇒ {
4   button.style.backgroundColor = "red";
5  });
```

EXERCÍCIOS

- Trocar cor da tela
- Trocar cor do button
- Emitir um alerta quando a cor da Tela for alterada
- Fazer um pequeno formulário de usuário e senha e caso a senha digitada for “12345” mostra a mensagem “usuário autenticado”, caso não mostra a mensagem “usuário não autenticado”.

TRABALHO FINAL

Controle de Material

Cimento 50 KG

41.00

3000

Salvar

Nome	Preço	Quantidade		
Prego	R$ 50.00	3		
Luva	R$ 70.00	5		
Cimento 50 KG	R$ 41.00	3000		

TRABALHO FINAL

Utilizar neste projeto as tecnologias:

- JavaScript
- HTML
- CSS
- Bootstrap
- Fontawesome
- LocalStorage

TRABALHO FINAL

Posteriormente, transpor sua aplicação para (precisa no Node instalado):

- Vite com VanillaJS.
- https://vitejs.dev/
- Executar os seguintes comando no terminal/prompt/powershell
- npm create vite@latest controle-material -- --template vanilla
- cd controle-material
- npm install
- npm run dev