
Enxurrada de Bits

# Integração do JavaScript com o HTML

# variável do tipo objeto

- Um objeto é uma coleção de propriedades, e uma propriedade é uma associação entre um nome (ou chave) e um valor
- Cada par nome/valor deve ser separado por uma vírgula e o nome e valor, em cada caso, separados por dois pontos

Sintaxe:

```
const nome_objeto = {
    nome_membro1: valor_membro1,
    nome_membro2: valor_membro2,
};
```

# variável do tipo objeto

*Exemplo:*

```
// criando um objeto vazio
let pessoa = {}

//populando o objeto
pessoa = {
  nome: 'Ana Moreira',
  idade: 18,
  profissao: 'Estudante',
  saudacao: function(nome) {
              alert(`Olá! Eu sou ${nome}.`)
            }
}

//acessando valores do objeto
const nome = pessoa.nome;
pessoa.saudacao(nome);
```

# window

- É um objeto que possui informações e utilidades importantes sobre a janela
  - window.history - funções de manipulação do histórico da página
  - window.location - informações acerca do endereço da página
  - window.console - objeto de acesso à saída do terminal
  - window.document - acesso a estrutura html da página

Como o objeto window é o único objeto que o navegador expõe, podemos acessar suas propriedades sem usar "window"

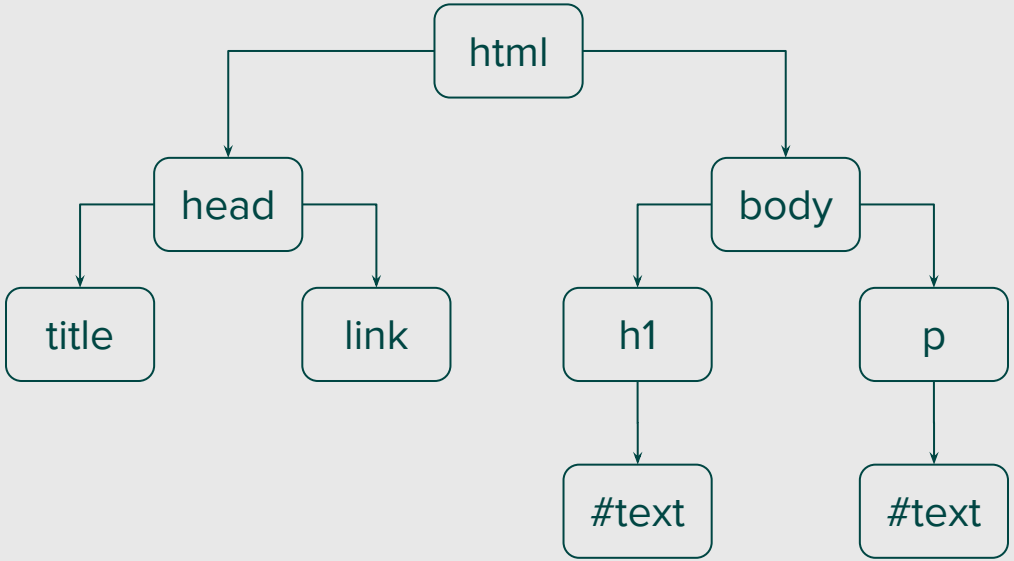
*Exemplo:*

**window.console.log('mensagem')** é o mesmo que **console.log('mensagem')**

Vamos falar agora sobre window.document, ou apenas document

# document

- O objeto document dá acesso ao **Document Object Model** (DOM)
- O DOM é uma representação da estrutura dos elementos html na forma de árvore:



# document

Com o document é possível...

# document

- Recuperar nós específicos da árvore (elemento.querySelector)
  - document.querySelector(seletor);

```
<!DOCTYPE html>
<html>
    <head>
      <title>Trabalhando com elementos</title>
    </head>
    <body>
      <div id="meu_id" class="meu_class">Exemplo de querySelectos</div>
      <script>
        // pegando um elemento no html com querySelector
        const div = document.querySelector("div");
        const div_id = document.querySelector("#meu_id");
        const div_class = document.querySelector(".meu_class");
      </script>
    </body>
</html>
```

Mais sobre querySelector...

# document.querySelector(seletor)

- Pega um elemento html por meio de um seletor
- Retorna o primeiro elemento que corresponde aos seletores CSS especificados na chamada do método
- Dentro dos parênteses podemos colocar qualquer um dos seletores CSS (nome da tag, id, class, etc)
- Caso não seja encontrado um elemento correspondente, é retornado *null* (um dos valores primitivos do JavaScript que representa um valor nulo ou "vazio" )

# document.querySelector(seletor)

*Exemplos:*

- const x = document.querySelector("p");
  - x é o primeiro elemento <p> do documento HTML
- const y = document.querySelector("div > p");
  - y é o primeiro elemento <p> que estiver logo dentro de uma div
- const z = document.querySelector(".exemplo");
  - z é o primeiro elemento que tiver a class “exemplo”
- const w = document.querySelector("#exemplo");
  - w é o elemento que tiver o id “exemplo”

# document

- Criar um elemento HTML (elemento.createElement)
  - document.createElement(nome-da-tag);

```
// cria um novo elemento
const nova_div = document.createElement("div");
const novo_p = document.createElement("p");
const novo_h1 = document.createElement("h1");
const nova_img = document.createElement("img");
```

- Criar um nó de texto para adicionar a um elemento HTML (elemento.createTextNode)
  - document.createTextNode(“algum texto”);

```
// cria um nó de texto
const conteudoNovo = document.createTextNode('Olá, mundo!');
```

# document

- Adicionar um elemento ao final da lista de filhos de um pai (elemento.appendChild)
  - document.body.appendChild(filho);

```
// adiciona o texto criado na div criada
nova_div.appendChild(conteudoNovo);
```

- Acessar e alterar o conteúdo interno dos elementos (elemento.innerHTML)
  - document.body.innerHTML = '<tag></tag>';

```
// adiciona um conteúdo na div criada
nova_div.innerHTML = '<img src="images/desenho.png">';
```

Obs.: Este método apaga completamente o conteúdo de um elemento e substitui pelo o que você está mandando

# document

- Adicionar um elemento em uma posição específica do pai (elemento.insertAdjacentHTML)
  - document.body.insertAdjacentHTML(posicao, texto);

**posicao** é a posição em relação ao elemento, e deve ser um dos seguintes tipos:
  - 'beforebegin' - antes do elemento
  - 'afterbegin' - dentro do elemento e antes de seu primeiro filho
  - 'beforeend' - dentro do elemento e após seu último filho
  - 'afterend' - após o elemento.

**texto** é a string a ser analisada como HTML e inserido na árvore

Mais sobre insertAdjacentHTML...

# document.insertAdjacentHTML(posicao, texto)

```
<!DOCTYPE html>
<html>
    <head>
    <title>Trabalhando com elementos</title>
    </head>
    <body>
      <!-- beforebegin -->
      <div>
        <!-- afterbegin -->
        <p> Exemplo de insertAdjacentHTML</p>
        <!-- beforeend -->
      </div>
      <!-- afterend -->
      <script>
        const div = document.querySelector('div');
        div.insertAdjacentHTML('beforebegin', '<div>Exemplo 1</div>');
        div.insertAdjacentHTML('afterbegin', '<div>Exemplo 2</div>');
        div.insertAdjacentHTML('beforeend', '<div>Exemplo 3</div>');
        div.insertAdjacentHTML('afterend', '<div>Exemplo 4</div>');
      </script>
    </body>
</html>
```

# document

- Acessar e alterar o valor de qualquer atributo HTML do elemento (elemento.atributo)
  - botao.id
  - a.href
  - imagem.src
  - input.name

- Adicionar ou remover classes (elemento.classList.funcao)
  - h1.classList.add(“class”)
  - p.classList.remove(“class”)

Obs.: Lembrando que para pegar estes elementos é só utilizar o querySelector

# Criando elementos HTML e colocando na página

*Exemplo:*

- HTML antes do JavaScript

```
<!DOCTYPE html>
<html>
    <head>
      <title>Trabalhando com elementos</title>
    </head>
    <body>
      <div>
        <p class="paragrafo-antigo"> Exemplo geral </p>
      </div>
      <script src="arquivo.js"></script>
    </body>
</html>
```

# Criando elementos HTML e colocando na página

*Exemplo:*

- JavaScript

```
// pegando a div existente no html, alterando seu id e colocando um elemento h1 como seu primeiro filho
const antiga_div = document.querySelector('div');
antiga_div.id = 'div_id';
antiga_div.insertAdjacentHTML('afterbegin', '<h1 class="h1_class">Criando um h1 com insertAdjacentHTML</h1>');

// criando um elemento p e um nó de texto e adicionando o texto no final da tag p
const novo_p = document.createElement('p');
const texto = document.createTextNode('Criando um texto com createTextNode');
novo_p.appendChild(texto);
// adicionando uma classe a p e, depois, o inserindo no final da tag body
novo_p.classList.add('paragrafo-novo');
document.body.appendChild(novo_p);

// pegando o p já existente no html e removendo sua classe
const antigo_p = document.querySelector('.paragrafo-antigo');
antigo_p.classList.remove('paragrafo-antigo');
```

# Criando elementos HTML e colocando na página

*Exemplo:*

- HTML depois do JavaScript

```
<!DOCTYPE html>
<html>
    <head>
      <title>Trabalhando com elementos</title>
    </head>
    <body>
      <div id="div_id">
        <h1 class="h1_class">Criando um h1 com insertAdjacentHTML</h1>
        <p class> Exemplo geral </p>
      </div>
      <script src="arquivo.js"></script>
      <p class="paragrafo-novo">Criando um texto com createTextNode</p>
    </body>
</html>
```

# Criando elementos HTML e colocando na página

*Exemplo:*

- Resultado


Mas não para por aí!
Apesar de termos adicionados vários elementos legais na nossa página, observe que ela ainda está sem graça e, para melhorar isso, basta utilizarmos os atributos de id e class que adicionamos aos elementos para estilizá-los no CSS 🤗

# Dúvidas?

# Referências

Material da disciplina de Programação Web do prof. Flávio Coutinho:
https://fegemo.github.io/cefet-web/classes/js2/

Documentação da Mozilla Developer Network:
https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Manipulating_documents

Documentação da W3Schools:
https://www.w3schools.com/js/js_htmldom_html.asp

# Leiam

Conceitos iniciais de Programação Orientada a Objetos (POO)
https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript
(video aula)
https://www.codigofluente.com.br/11-javascript-programacao-orientada-a-objetos-parte-01/

# Colocando em Prática

- Você irá fazer um site utilizando unicamente JavaScript
- Para isso, você deve:
  - Criar um arquivo HTML apenas com as divisões semânticas (header, main, etc)
  - Você irá pegar o header com o querySelector e colocar como filho dele um título “h1” que você irá criar no javascript
  - Você irá pegar o main com o querySelector e colocar como filho dele dois divs: cada um dos divs deve ter como filhos um título “h2” e um parágrafo “p”
- Obs: Você deve estilizar o seu site!!! Para isso, lembre de como adicionar ids e classes no javascript e defina a estilização no arquivo .CSS