# Tema 12 – JavaScript HTML DOM – Parte 02

## Introdução

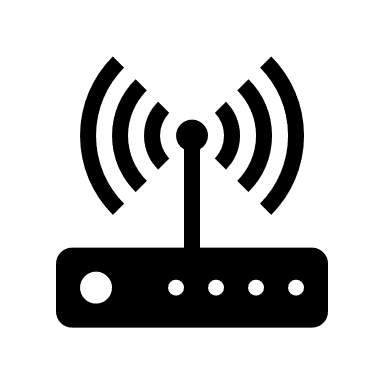
Olá, futuras mentes desenvolvedoras, sejam bem-vindos do curso de Frontend do Instituto da Oportunidade Social. Nessa aula, você aprenderá mais sobre o Document Object Model (Modelo de objeto de documentos).

## Criação do projeto inicial

Siga os passos para criar o projeto:

1. Abra o VS Code e escolha um diretório de trabalho para o seu projeto.
2. Crie um diretório para seu projeto com o nome representativo, por exemplo, 12\_JS\_DOM\_Parte\_02.
3. Crie um arquivo dentro do diretório do projeto com o nome index.html.
4. Insira o seguinte código no seu arquivo index.html.

|  |
| --- |
| <!DOCTYPE html>  <html lang="pt-br">      <head>          <meta charset="UTF-8" />          <link rel="shortcut icon" href="#" />          <title>JavaScript com HTML DOM - Parte 02</title>      </head>      <body>          <h2 id="titulo">Minha página</h2>          <ul id="items" class="list">              <li class="item">Item 1</li>              <li class="item">Item 2</li>              <li class="item" name="fitem">Item 3</li>              <li class="item" name="fitem">Item 4</li>          </ul>          <script src="./js/main.js"></script>      </body>  </html> |

1. Esse código mostra a marcação <script> sem nenhum código JavaScript entre a abertura e o fechamento da tag, apenas o atributo **src** com o valor main.js. Isso significa que o código JavaScript está em um arquivo externo. Portanto, temos que criar esse novo arquivo **main.js** dentro do diretório do projeto.
2. Vamos deixar o arquivo main.js vazio e à medida que vamos aprendendo coisas novas iremos completar a implementação do código JavaScript.
3. Abra o arquivo index.html, clique no botão **Go Live** da extensão **Live Server**. A página inicial mostrada será:

Graphical user interface, text, application

Description automatically generated

## Método getElementById

O método **getElementById** é o modo para comum de acessar um elemento HTML na página web. Esse método retorna a referência do elemento através do atributo **ID** ou **null** se a ID não for encontrada. Esse é um dos motivos do atributo id em uma página web ser único.

Observe no arquivo index.html, que colocamos o atributo **id** na marcação <h2>.

### Vamos praticar

1. No arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 1 | // Método getElementById  let titulo = document.getElementById('titulo');  // Alterando o conteúdo do elemento  titulo.innerHTML = 'Olá meus caros alunos!';  // Configurando o estilo CSS do elemento  titulo.style.textAlign = 'center';  titulo.style.backgroundColor = '#CCCCC9';  titulo.style.borderBottom = 'solid 3px #000';  titulo.style.margin = '20px'; |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

Observe que a instrução na linha 2 criou uma referência do elemento <h2> da página e, então foi possível manipular o conteúdo (linha 4) e as configurações de estilo (linhas 6 a 9) desse elemento.

## Método getElementsByClassName

O método **getElementsByClassName** retorna a coleção de todos os elementos do documento com a classe específica. Isso significa que esse método pode retornar mais de um elemento.

### Vamos praticar

1. Continuando a implementação do projeto, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 11 | // Método getElementsByClassName  let items = document.getElementsByClassName('item');  console.log(items);  console.log(items[1]);  items[1].textContent = 'Hello 2';  items[1].style.fontWeight = 'bold';  items[1].style.backgroundColor = 'yellow'; |
| 12 |
| 13 |
| 14 |
| 15 |
| 16 |
| 17 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

Na linha 12, buscamos todos os itens da lista, pois todos eles têm a classe **item** associada a eles. A instrução na linha 13 imprime no console os itens retornado pelo método e armazenados no objeto **items**:

A picture containing text

Description automatically generated

Como podemos ver o objeto **items** possui vários itens dentro dele e podemos acessar individualmente como mostra a instrução na linha 14 ,que resulta na impressão no console do segundo item do objeto:



A instruções nas linhas 15, 16 e 17 alteram o conteúdo do segundo item da lista, a espessura da fonte e a cor de fundo, respectivamente.

1. Podemos utilizar um laço de repetição para alterar a cor de fundo de todos os itens da lista, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 19 | for (let i = 0; i < items.length; i++) {      items[i].style.backgroundColor = '#f4f4f4';  } |
| 20 |
| 21 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

O laço de repetição percorre os itens dentro do objeto e modifica a cor de fundo de cada item referenciando-o pelo índice do array.

## Método getElementsByTagName

O método **getElementsByTagName** retorna a coleção de todos os elementos do documento com a tag, ou seja, com a mesma marcação HTML. Isso significa que esse método pode retornar mais de um elemento.

### Vamos praticar

1. Continuando a implementação do projeto, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 23 | // Método getElementsByTagName  let li = document.getElementsByTagName('li');  console.log(li);  for (let i = 0; i < li.length; i++) {      if (i % 2) li[i].style.backgroundColor = '#f4f4f4';      else li[i].style.backgroundColor = '#fff';  } |
| 24 |
| 25 |
| 26 |
| 27 |
| 28 |
| 29 |
| 30 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

No nosso exemplo, o método **getElementsByClassName** e **getElementsByTagName** permitiram manipular os mesmos elementos. Isso foi intencional para mostrar que existem diversas maneiras de acessar um elemento no DOM.

Nas linhas 27 a 30, alteramos as cores de fundo dos elementos <li>, os elementos com índices pares são configurados com fundo branco (#fff) e os elementos com índices ímpares são configurados com fundo gelo (#f4f4f4). Lembre-se, o primeiro elemento do array possui índice zero, portanto com quarto elemento na lista temos: índice 0, índice 1, índice 2 e índice 3.

## Método getElementsByName

O método **getElementsByName** retorna a coleção de todos os elementos do documento com o nome específico, ou seja, esse método verifica o atributo HTML **name**. Isso significa que esse método pode retornar mais de um elemento.

### Vamos praticar

1. Continuando a implementação do projeto, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 32 | // Método getElementsByName  let nome = document.getElementsByName('fitem');  console.clear();  console.log(nome);  nome[0].textContent = 'Olá pessoas';  nome[0].style.backgroundColor = 'yellow';  nome[1].textContent = 'Tudo bem?';  nome[1].style.backgroundColor = '#BAC1FB'; |
| 33 |
| 34 |
| 35 |
| 36 |
| 37 |
| 38 |
| 39 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

O método **getElementsByName** retornou os dois últimos itens da lista, pois eles têm o nome igual a fitem:

|  |
| --- |
| <ul id="items" class="list">      <li class="item">Item 1</li>      <li class="item">Item 2</li>      <li class="item" name="fitem">Item 3</li>     <li class="item" name="fitem">Item 4</li>  </ul> |

E então foi possível acessar as propriedades **textContent** e **style** para alterar o conteúdo e o estilo desses elementos. A instrução nome[0].textContent = 'Olá pessoas' altera o conteúdo do terceiro item da lista e nome[0].style.backgroundColor = 'yellow' altera a cor de fundo para amarelo. A instrução nome[1].textContent = 'Tudo bem?' altera o conteúdo do quarto item da lista e nome[1].style.backgroundColor = '#BAC1FB' altera a cor de fundo para azul claro.

## Remover um elemento no DOM

O método **remove()** permite remover um elemento do DOM.

### Vamos praticar

1. Continuando a implementação do projeto, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 41 | // Remove elemento do DOM  let item2 = document.getElementById('item2');  item2.remove(); |
| 42 |
| 43 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

Observe que o item2 não aparece mais na página, pois ele foi removido com o método **remove()**.

Vamos retornar esse elemento novamente.

1. Continuando a implementação do projeto, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 45 | // Retornando o elemento  let lista = document.getElementById('items');  let item1 = document.getElementById('item1');  lista.insertBefore(item2, item1.nextSibling); |
| 46 |
| 47 |
| 48 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

Observe que o item2 foi devolvido para o seu lugar. A instrução na linha 46 busca o elemento pai dos itens da lista e a instrução na linha 47 busca o elemento filho, que queremos ter como referência para inserir o item2. Nesse caso, queremos inserir o item 2, após o item 1.

A instrução na linha 48 insere o item 2 dentro da lista pelo método **insertBefore** e a propriedade **nextSibling** retorna o próximo elemento irmão (que tem o mesmo pai). Desse modo, o item 2 pôde ser inserido após o seu irmão item 1.

## Alterar estilo da lista

Vamos alterar o estilo da lista não ordenada. Siga os passos para fazer isso:

1. Continuando a implementação do projeto, no arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 50 | // Alterando o estilo da lista  let ul = document.getElementById('items');  ul.style.listStyle = 'none'; |
| 51 |
| 52 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

A propriedade **listStyle** configura o estilo da lista e no exemplo para não mostrar o marcador da lista (o bullet point).

1. No arquivo **main.js**, insira o seguinte código.

|  |  |
| --- | --- |
| 54 | // Marcadores numéricos  ul.style.listStyle = 'decimal inside'; |
| 55 |

1. Veja o resultado apresentado no navegador.

Graphical user interface, text, application

Description automatically generated

Agora é exibido os marcadores numéricos na lista.

## Para aprender mais

Procure sempre aprender e estudar mais. Seguem alguns links para você estudar e aprender mais:

getElementById:

<https://developer.mozilla.org/pt-BR/docs/Web/API/Document/getElementById>

<https://www.w3schools.com/js/js_htmldom_methods.asp>

getElementsByClassName:

<https://developer.mozilla.org/pt-BR/docs/Web/API/Document/getElementsByClassName>

<https://www.w3schools.com/Jsref/met_document_getelementsbyclassname.asp>

getElementsByTagName:

<https://developer.mozilla.org/pt-BR/docs/Web/API/Document/getElementsByTagName>

<https://www.w3schools.com/Jsref/met_document_getelementsbytagname.asp>

getElementsByName:

<https://developer.mozilla.org/pt-BR/docs/Web/API/Document/getElementsByName>

<https://www.w3schools.com/Jsref/met_doc_getelementsbyname.asp>

Element.remove():

<https://developer.mozilla.org/en-US/docs/Web/API/Element/remove>