DOM - Document Object Model

OBJETIVOS DA AULA DE HOJE:

→ Entendermos o conceito de DOM;

→ Entender as possibilidades de interação com nossa aplicação que o DOM nos permite fazer;

→ Aprender a acessar um elemento específico de uma página Web;

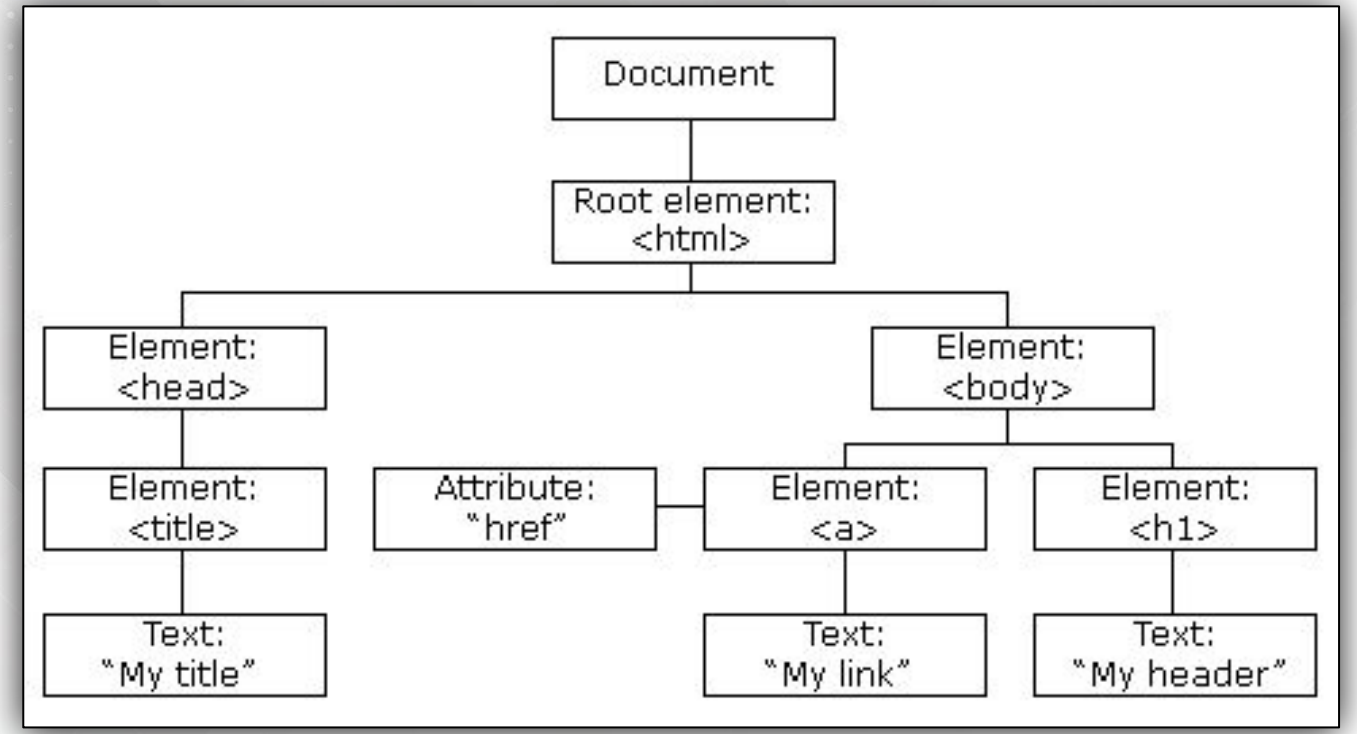
→ Aprender a adicionar, modificar e excluí-los.

DOM - Definição

*Ele é uma **representação** orientada a objetos da página da web, que pode ser modificada com uma linguagem JavaScript.*

-MDN (Introdução ao DOM)

DOM - Definição



Vamos estruturar uma árvore DOM de uma aplicação Web?

DOM - Definição

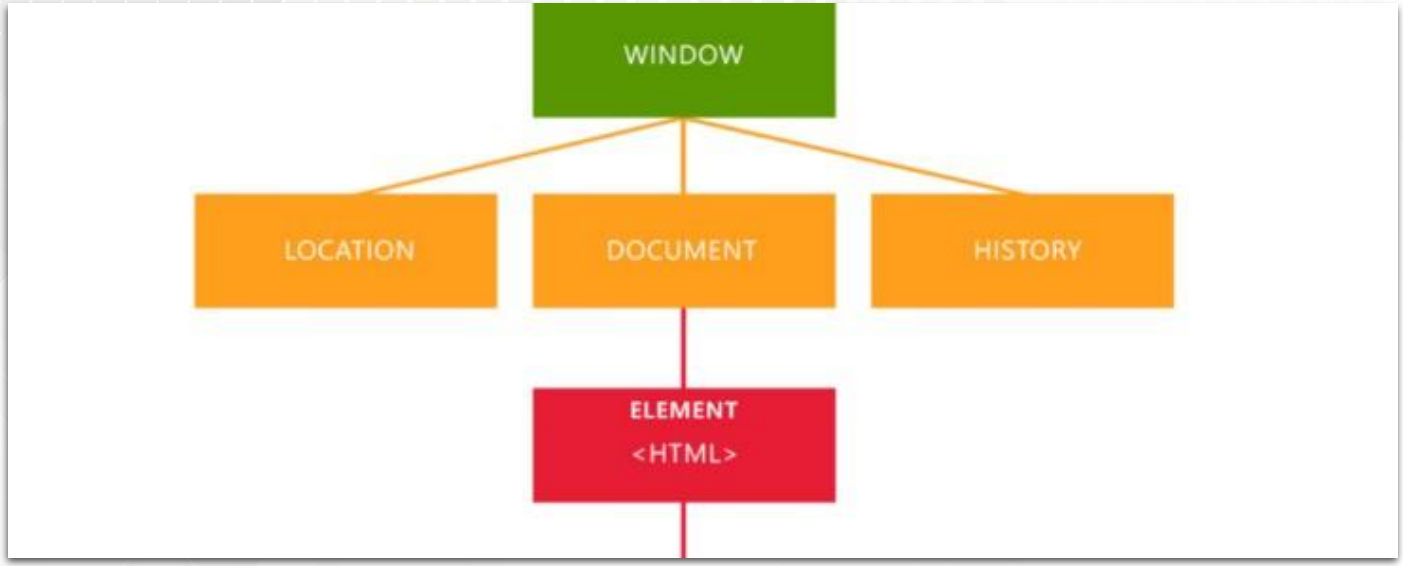
O DOM representa a forma/estrutura como o browser enxerga seu documento HTML. (ou XML)

DOM - Definição

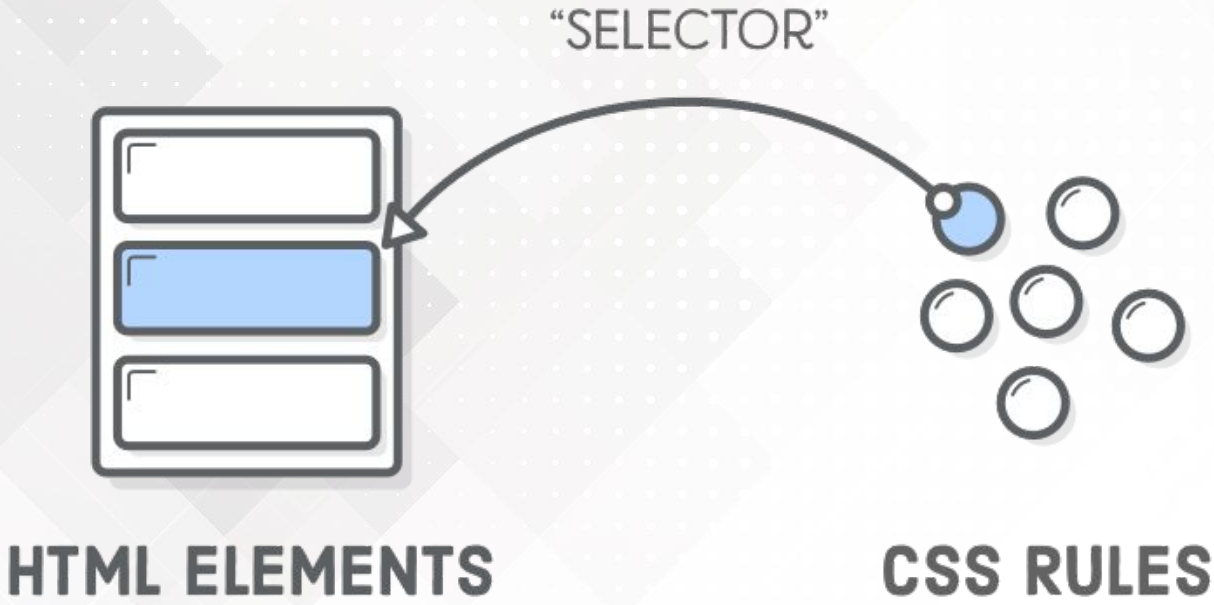
Com o DOM, o Javascript consegue:

- Modificar elementos, atributos e estilos de uma página.
- Excluir qualquer elemento e atributo.
- Adicionar novos elementos ou atributos.
- Responder a todos os eventos na página.
- Criar novos eventos na página.

DOM - Hierarquia para acessarmos o documento HTML



Seletores



Seletores

Para acessar os **elementos** de uma página, usamos seletores.

Cada seletor pode retornar apenas **um elemento** ou uma **lista de elementos**.

Seletores

O objeto document tem os seguintes seletores como método:

- **document.getElementById(ID)**
- **document.getElementsByClassName(Classe)**
- **document.getElementsByTagName(Tag)**
- **document.querySelector(cssQuery)**
- **document.querySelectorAll(cssQuery)**

Seletores

O objeto document tem os seguintes seletores como método:

- document.getElementById(ID) — único
- document.getElementsByClassName(Classe) — array
- document.getElementsByTagName(Tag) — array
- document.querySelector(cssQuery) — único
- document.querySelectorAll(cssQuery) — array

document.getElementById()

```
<script>
    let h1 = document.getElementById("titulo")
    h1.style.display = 'none';
</script>


<h1 id="titulo">Bem-vindo!</h1>
```

document.querySelector(CSSQuery)

```
<script>
    let vermelho = document.querySelector(".vermelho");
    vermelho.style.color = 'red’;
</script>
<h1 class="vermelho">Bem-vindo!</h1>
<p class="vermelho">Esse é meu site :)</p>
```

document.querySelector(CSSQuery)

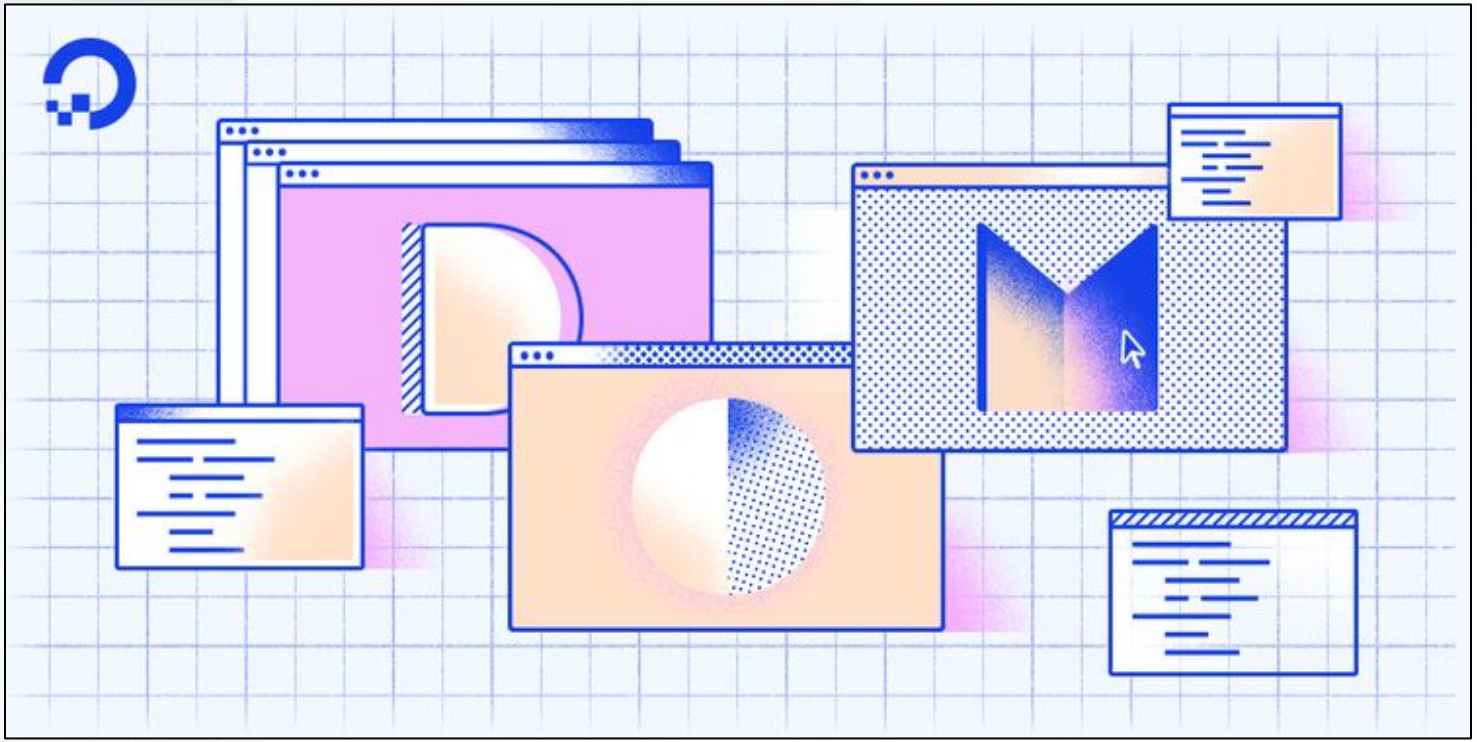
```
<script>
    let vermelho =
document.querySelector(".verm
    vermelho.style
</scri
     ass="vermelho">Bem-vindo!</h1>
<p class="vermelho">Esse é meu site :)</p>
```

CUIDADO = Traz o primeiro elemento que encontrar

document.querySelectorAll(CSSQuery)

```
<script>
    let vermelhos =
    document.querySelectorAll(".vermelho")';
    for (let i = 0; i < vermelhos.length; i++) {
        vermelhos[i].style.color = 'red'
    }
</script>
<h1 class="vermelho">Bem-vindo!</h1>
<p class="vermelho">Esse é meu site :)</p>
```

Modificar elementos



DigitalHouse >
Coding School

Atributos

Assim que um elemento é selecionado, é possível acessar os atributos dele usando a propriedade **attributes**.

Retorna um mapa (é como um array) que tem os nomes e valores dos atributos deste elemento.

elemento.attributes;

getAttribute() / setAttribute()

O método **getAttribute** aceita uma string como parâmetro com o nome do atributo que queremos obter. Retorna o valor do atributo. Caso ele não seja encontrado, retorna null.

```
elemento.getAttribute("href");
```

O método **setAttribute** permite adicionar um novo atributo ou modificar um existente.

```
elemento.setAttribute("class","vermelho"):
```

hasAttribute() / removeAttribute()

O método **hasAttribute** aceita uma string como parâmetro com o nome do atributo que queremos saber se existe no elemento.

Retorna um valor booleano.

elemento.hasAttribute(nomeAtributoEmString);

Com o método **removeAttribute,** podemos remover um atributo existente.

elemento.removeAttribute(nomeAtributo):

Estilos

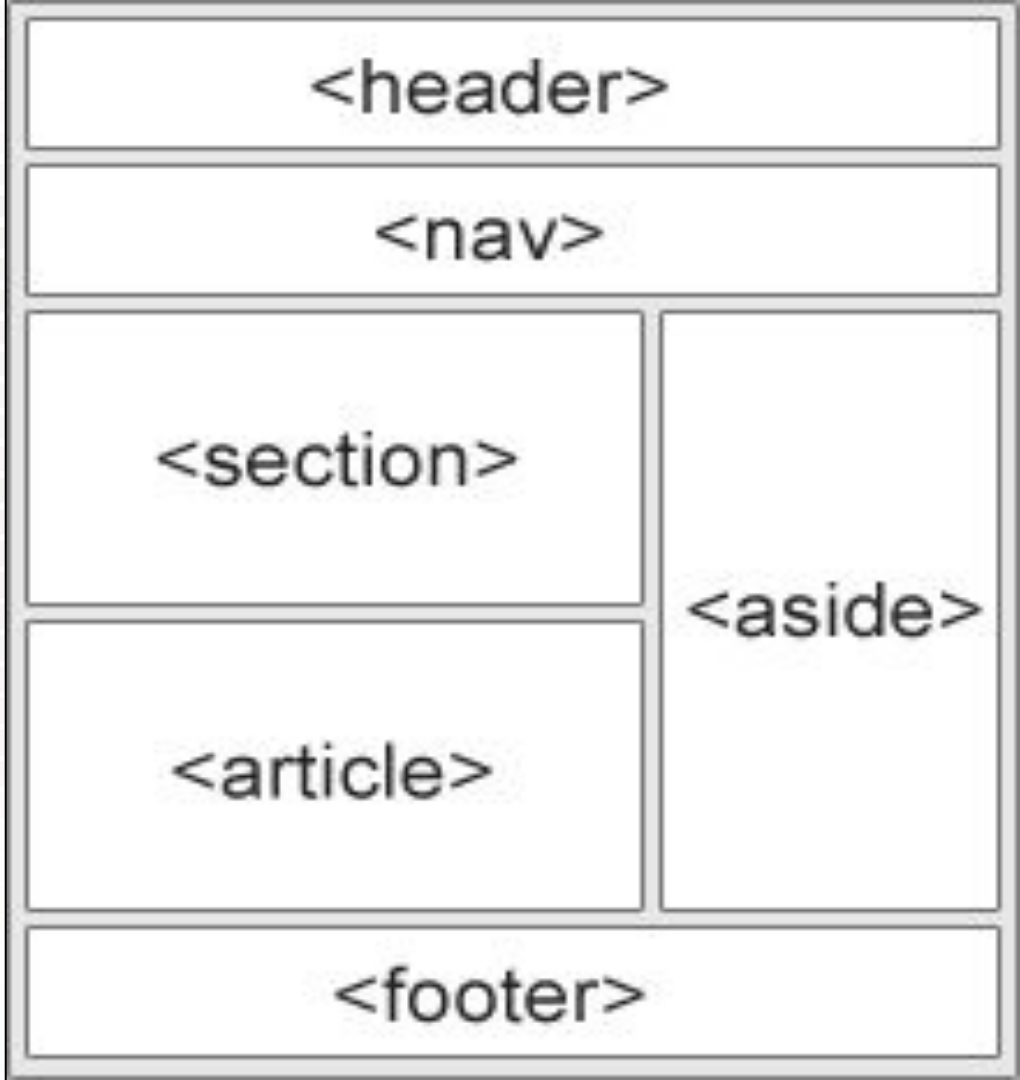
- Os elementos HTML têm uma propriedade chamada ***style***, que retorna um objeto literal que representa os estilos desse objeto.
- É possível adicionar ou modificar seus atributos.
- Os nomes das propriedades CSS em JavaScript são escritos no seguinte formato: **nomeDePropiedadeDeCss**.

```
elemento.style.color = "red";     // configuramos a cor vermelha

document.getElementById("myElement").style.cssText = "display: block;
position: absolute";              //configurando mais de uma propriedade
```

Link: Lista de propriedades

Elemento



Criando elementos - createElement

O método **createElement** permite criar novos elementos HTML.

createElement aceita como parâmetro strings com nomes de tags HTML (a, div, span, li, ul, etc).

```
let btn = document.createElement("BUTTON");
```

O método **createTextNode** permite criar novos textos HTML.

```
let texto = document.createTextNode("Olá, sou um texto");
```

Inserindo elementos - appendChild

```
let li = document.createElement("LI");

let textoLi = document.createTextNode("Item lista");

li.appendChild(textoLi);

document.getElementById("minhaLista").appendChild(li);

<div id="minhaLista"></div>
```

→ **appendChild** permite inserir um nó dentro de outro.

textContext / innerHTML

→ **textContent** permite ler ou escrever conteúdo como **texto**.

```
elemento.textContent = "texto";

elemento.textContent;      // texto
```

→ **innerHTML** permite escrever conteúdo em um elemento.

```
<div id="cabeçalho"></div>

let elemento = document.getElementById("cabeçalho");

elemento.innerHTML = "<h1>Meu elemento HTML</h1>";
```

Removendo elementos - removeChild

O objeto nó tem um método chamado **removeChild** que permite remover nós filhos.

Para poder remover um nó, primeiro precisamos selecioná-lo.

```
let elemento = document.getElementById(ID);
let elementoFilho = elemento.children.item(nroItem);
elemento.removeChild(elementoFilho);
```

Até a próxima aula!

DigitalHouse >
Coding School