DOM

Document Object Model

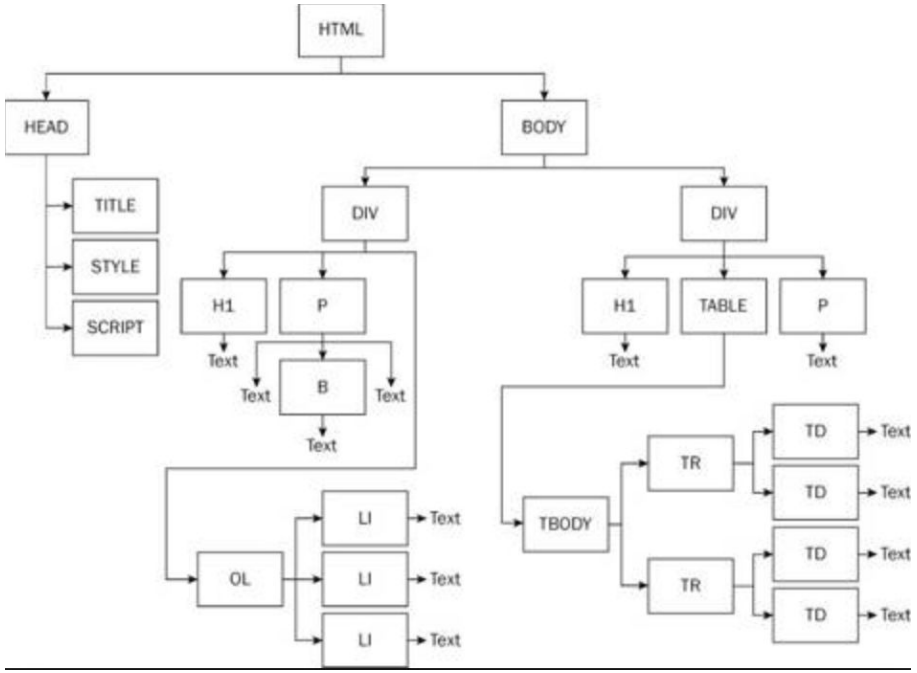
Com JavaScript

DOM – Introdução

- DOM – Document Object Model ou Modelo de Objeto de Documento, é uma interface que representa o documento HTML da página web.
- E uma interface de programaçao para manipulação de documentos HTML, XML e SVG e está disponível para várias linguagens.
- O DOM fornece essa interface através de nós e objetos para que possamos manipular toda nossa página através de métodos e propriedades, em nosso caso, JavaScript, onde veremos mais adiante.

DOM – Árvore

- A ideia lógica do DOM é como uma árvore do HTML conforme a imagem abaixo:



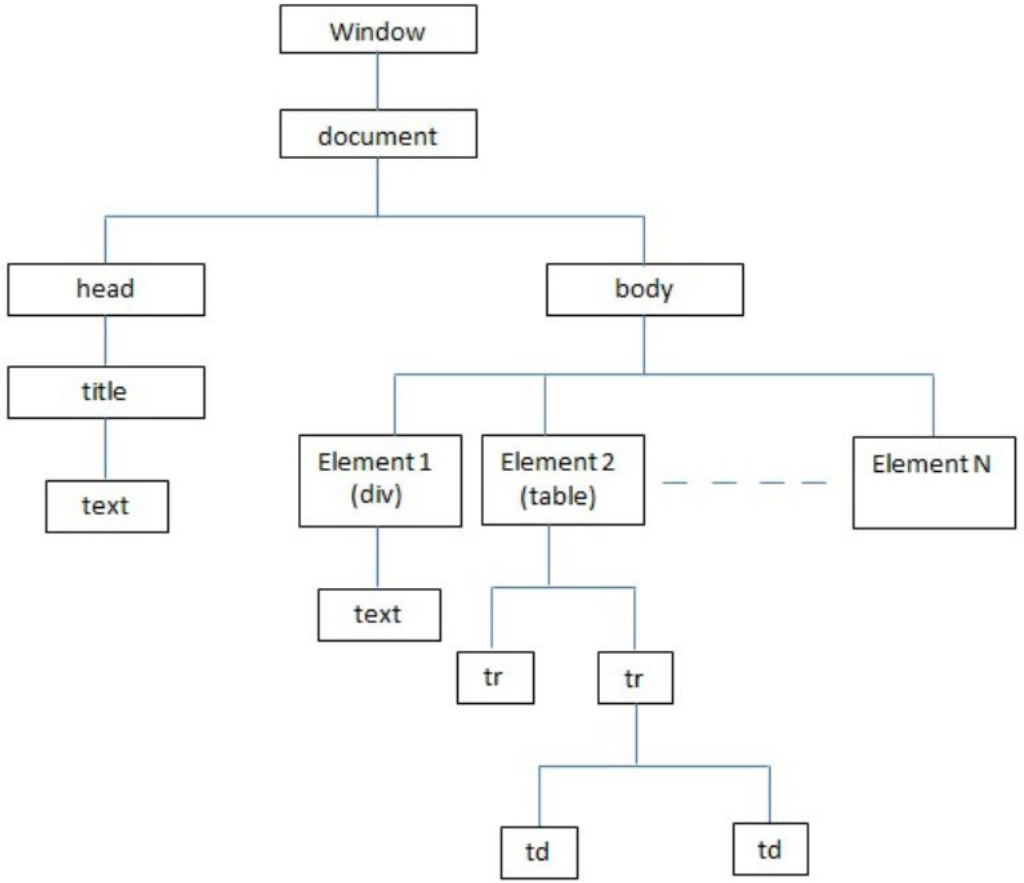
DOM – Árvore

- A hierarquia começa pelo objeto window:

- window.document.body

OU APENAS

- document.body



DOM – PRINCIPAIS MÉTODOS

- Nós veremos os principais métodos para o entendimento na criação e manipulação de elementos HTML através do DOM.
  - getElementById()
  - getElementsByClassName()
  - getElementsByTagName()
  - querySelector()
  - querySelectorAll()
  - createElement()
  - appendChild()
  - removeChild()
  - parentNode()

DOM – Dica

- Lembrando que JavaScript é casa sensitive isso quer dizer que os métodos do DOM são escritos dessa forma: **getElementById** e não getelementbyid.

- *Os nomes desses métodos foram assinados com o padrão Camel Case a partir da segunda palavra.*

DOM – getElementById()

- **getElementById()**: pega um elemento através do id e retorna o seu objeto.

```
<form class="form">
    <label class="form__label" for="nick-name">Nome</label>
    <input class="form__input" type="text" id="nome" value="Eduardo" >
    <label class="form__label" for="nick-name">Nome</label>
    <input class="form__input" type="text" id="email" value="eduardo@email.com">
</form>

<script>
    var elemento = document.getElementById('email');
    console.log(elemento);
    console.log(elemento.value);
</script>
```

```
<input class="form__input" type="text" id="email" value="eduardo@email.com">
eduardo@email.com
```

DOM – getElementById()

- Opcionalmente você pode retornar apenas uma propriedade do objeto selecionado:

```
var elemento = document.getElementById('email').value;
console.log(elemento);
```

```
eduardo@email.com
```

DOM – getElementsByClasName()

- **getElementsByClassName():** Seleciona uma **coleção** de elementos através do nome da classe. Perceba o plural "**Elements**"

```
<ul>
    <li class="corsim">item 1</li>
    <li class="cornao">item 2</li>
    <li class="corsim">item 3</li>
    <li class="cornao">item 4</li>
</ul>

<script>
    var itens = document.getElementsByClassName('cornao');
    console.log(itens);
    console.log(itens[1]);
</script>
```

```
▼HTMLCollection(2) [li.cornao, li.cornao]
  ▶0: li.cornao
  ▶1: li.cornao
    length: 2
  ▶__proto__: HTMLCollection
<li class="cornao">item 4</li>
```

DOM – getElementsByTagName()

- **getElementsByTagName():** retorna os elementos, ou seja, uma coleção, através do nome de uma tag:

```html
<form class="form">
    <label class="form__label" for="nick-name">Nome</label>
    <input class="form__input" type="text" id="nick-name" value="Eduardo" >
    <label class="form__label" for="email">Nome</label>
    <input class="form__input" type="text" id="email" value="eduardo@email.com">
</form>
```

```js
var elementos = document.getElementsByTagName('input');

console.log(elementos);
```

```
▼ HTMLCollection(2) [input#nick-name.form__input, input#email.form__input, nick-name: input#nick-name.form__input, email: input#email.form__input]
  ▶ 0: input#nick-name.form__input
  ▶ 1: input#email.form__input
    length: 2
  ▶ email: input#email.form__input
  ▶ nick-name: input#nick-name.form__input
  ▶ __proto__: HTMLCollection
```

DOM – getElementsByTagName()

```
var elementos = document.getElementsByTagName('input');

for (var posicao = 0; posicao < elementos.length; posicao++) {
    console.log("valor do input "+ elementos[posicao].id.toUpperCase());
    console.log("    " + elementos[posicao].value);
}
```

```
valor do input NOME
    Eduardo
valor do input EMAIL
    eduardo@email.com
```

DOM – querySelector()

- **Mais Exemplos:**

```
var elemento = document.querySelector('.form__label');
console.log(elemento);
```

```
<label class="form__label" for="nick-name">Nome</label>
```

```
var elemento = document.querySelector('#email');
console.log(elemento);
```

```
<input class="form__input" type="text" id="email" value="eduardo@email.com">
```

DOM – querySelector()

- **querySelector()**: retorna **um elemento**, da mesma forma que o getElementById() porém você você utiliza um seletor CSS ao invés do *nome do ID*.

```
<form class="form">
    <label class="form__label" for="nick-name">Nome</label>
    <input class="form__input" type="text" id="nick-name" value="Eduardo" >
    <label class="form__label" for="email">Nome</label>
    <input class="form__input" type="text" id="email" value="eduardo@email.com">
</form>
```

```
var elemento = document.querySelector('input');
console.log(elemento);
```

```
<input class="form__input" type="text" id="nome" value="Eduardo">
```

querySelector() - Dica

- Resumindo, você pode pegar um elemento utilizando os seletores CSS disponíveis. querySelector.

```
var elemento = document.querySelector('[id*=name]');
console.log(elemento);
```

```
<input class="form__input" type="text" id="nick-name" value="Eduardo">
```

```
var elemento = document.querySelector('input:last-child');
console.log(elemento);
```

```
<input class="form__input" type="text" id="email" value="eduardo@email.com">
```

```
var elemento = document.querySelector('.form__input');
console.log(elemento);
```

```
<input class="form__input" type="text" id="nick-name" value="Eduardo">
```

```
var elemento = document.querySelector('.form__input:last-child');
console.log(elemento);
```

```
<input class="form__input" type="text" id="email" value="eduardo@email.com">
```

DOM – querySelectorAll()

- **querySelectorAll()**: funciona da mesma forma que o querySelector porém retorna **um Array**.

```
<form class="form">
    <label class="form__label" for="nick-name">Nome</label>
    <input class="form__input" type="text" id="nick-name" value="Eduardo" >
    <label class="form__label" for="email">Nome</label>
    <input class="form__input" type="text" id="email" value="eduardo@email.com">
</form>
```

```
var elementos = document.querySelectorAll('.form__input');

//imprime os values de cada input no console
for (var i = 0; i < elementos.length; i++) {
    console.log(elementos[i].value);
}
```

```
Eduardo
eduardo@email.com
```

DOM – createElement e appendChild

- Esses dois métodos da interface DOM servem para criar elementos e adicioná-los ao DOM.
-
- É como se fossemos utilizar o innerHTML, porém aqui a ideia é criar cada elemento como um objeto separadamente.
-
- Imagine que você precisa criar um elemento div qualquer e adicioná-lo em sua página. Com innerHTML fizemos algo como na imagem abaxo:

```
var div = '<div>';
document.body.innerHTML = div;
```

DOM – createElement e appendChild

- Com create element você cria um fragmento do elemento em memória e logo em seguida utiliza o método appendChild para inserir o elemento de fato no HTML. Veja:

```
var div = document.createEleen('div');
document.body.appendChild(div);
```