Roteiro Didático de Aula
Aula 04 – JavaScript
24/09/2020

Assuntos que serão abordados:

Entendendo o DOM (Document Object Model)
Então, o que é o DOM?
Quais as vantagens do DOM?
Manipulando o DOM
Métodos
getElementById()
getElementsByClassName()
getElementsByTagName()
querySelector()
querySelectorAll()
Events
click
select
Percorrer elementos
.childNodes
.firstChild
.nodeName
.nodeValue
.nodeType
Elements
.lastElementChild
Referências

Entendendo o DOM (Document Object Model)

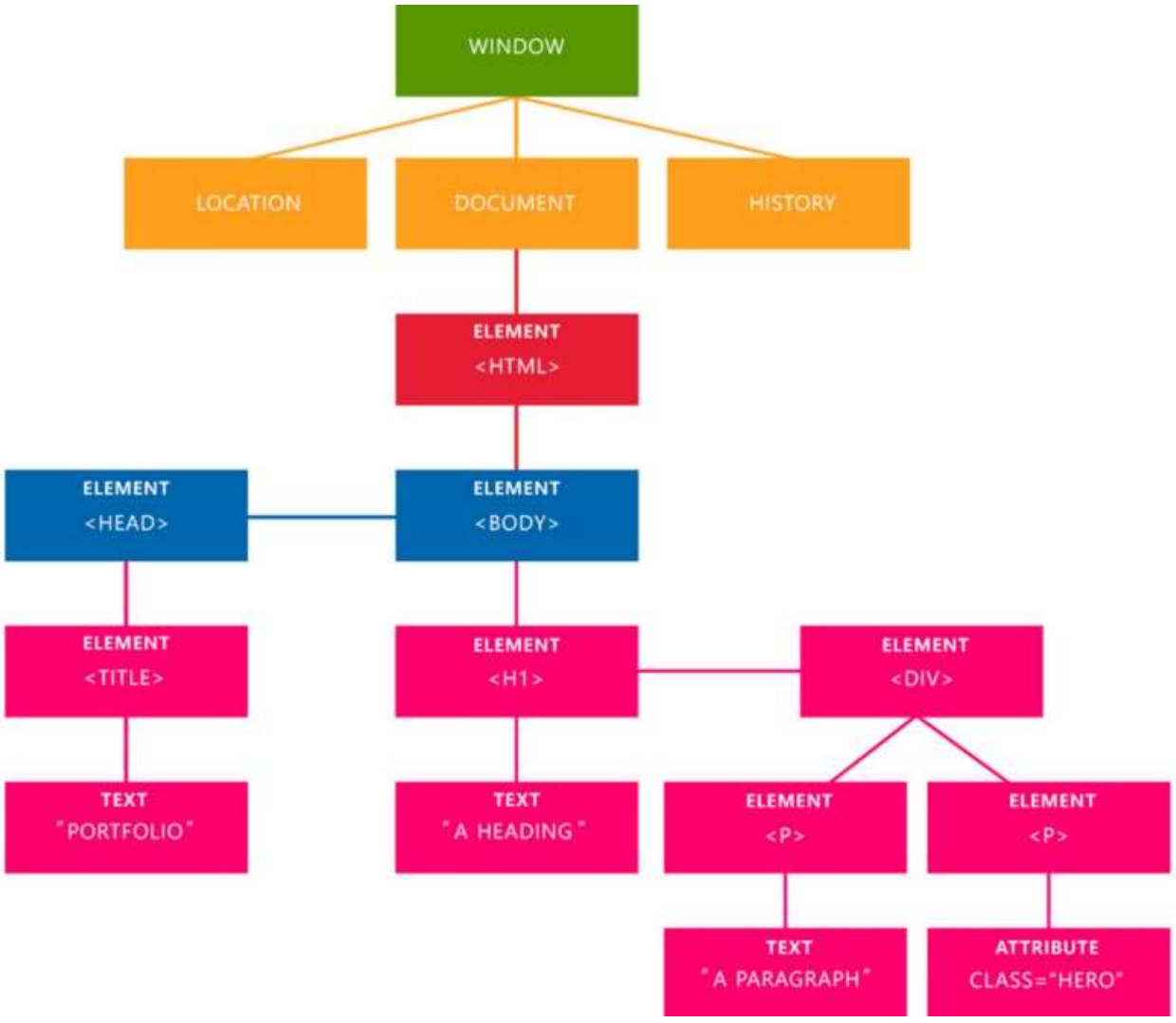
O DOM (Document Object Model) é uma interface que representa como os documentos HTML e XML são lidos pelo seu browser. Após o browser ler seu documento HTML, ele cria um objeto que faz uma representação estruturada do seu documento e define meios de como essa estrutura pode ser acessada. Nós podemos acessar e manipular o DOM com JavaScript, é a forma mais fácil e usada.

Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br

Quais as vantagens do DOM?

Com ele você tem infinitas possibilidades, você pode criar aplicações que atualizam os dados da página sem que seja necessário atualização. Pode também criar aplicações que são customizáveis pelo usuário, mudar o layout da página sem que seja necessário atualização. Arrastar, mover, excluir elementos. Ou seja, você tem infinitas possibilidades, milhares de coisas que você pode fazer manipulando o DOM, basta você usar sua criatividade.

Como ele é representado pelo browser:



A estrutura que o DOM constrói a partir da leitura do seu documento HTML.
Nessa imagem vemos a estrutura do DOM, suas marcações e como ele é montado pelo browser.
Nessa base, temos 4 pontos importantes que você vai ver bastante daqui pra frente:

**Document: **que como o nome diz, cuida de documentos HTML.

Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br

**Elements: **são todas as tags que estão em arquivos HTML ou XML se transformam em elementos da árvore DOM.

Texts: É o texto que vai entre os elementos. Todo o conteúdo das tags.

Attributes: É a junção de todos atributos para um nó específico. No caso, o attribute *class="hero"* está apontando para o elemento `<p>`.

Manipulando o DOM

Primeiramente, vamos criar um HTML como exemplo para mostrar como os métodos e os eventos funcionam.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Entendo o DOM (Document Object Model)</title>
</head>
<body>
    <div class="container">
        <h1><time>00:00:00</time></h1>
        <button id="start">Start</button>
        <button id="stop">Stop</button>
        <button id="reset">Reset</button>
    </div>
</body>
</html>
```

Métodos

O DOM possui muitos métodos, são eles que fazem a ligação entre os nodes (elementos) e os eventos. Lembrando que existem vários, para todos os tipos e você pode ver todos aqui.

getElementById()

Esse método retorna o elemento que estiver contendo o nome do ID passado. Como os IDs devem ser únicos, é um método muito útil para pegar apenas o elemento desejado.

```
var myStart = document.getElementsById('start');
```

myStart: elemento específico que se equipara com o seletor passado.
start: seletor passado, caso não houvesse nenhum ele retornaria *null*.

getElementsByClassName()

Esse método retorna um *HTMLCollection* de todos elementos que estiverem contendo o nome da classe passada.

```
var myContainer = document.getElementsByClassName('container');
```

myContainer: elemento específico que se equipara com o seletor passado.
.container: seletor passado, caso não houvesse nenhum ele retornaria *null*.

Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br

getElementsByTagName()

Na mesma maneira do método acima, ele também retorna uma *HTMLCollection* mas com uma diferença: esse método retorna todos elementos contendo a tag name passada.

```
var buttons = document.getElementsByTagName('button');
```

buttons: elemento específico que se equipara com o seletor passado.
button: tag name passada.

querySelector()

Retorna o primeiro elemento que se equipara ao seletor CSS passado. Lembrando que o seletor deve seguir a sintaxe CSS. Caso não tenha nenhum seletor, ele retornará *null*.

```
var resetButton = document.querySelector('#reset');
```

resetButton: primeiro elemento que se equipara com o seletor passado.
#reset: seletor passado, caso não houvesse nenhum ele retornaria *null*.

querySelectorAll()

Muito igual ao querySelector(), mas ele tem apenas uma diferença: retorna todos elementos que se equiparam ao seletor CSS passado. O seletor também deve seguir a sintaxe CSS, caso não haja nenhum ele retornará *null*.

```
var myButtons = document.querySelector('#buttons');
```

myButtons: elementos que se equiparam com o seletor passado.
#buttons: seletor passado, como nesse caso não há nenhum seletor com esse nome, ele retorna *null*.
Esses são apenas 3 métodos do DOM, existem vários e alguns são bastante usados, por exemplo o *createElement()* que cria um novo elemento HTML usando o nome da da tag a ser criada, o setAttribute() que você pode setar novos atributos para elementos HTML e muitos outros.

Você pode encontrar todos AQUI indo no canto esquerdo em _Methods_ .

Events

Os elementos DOM além de possuirem métodos também possuem eventos. São eles que fazem a interatividades dos elementos no documento, mas não se engane: eventos também são métodos.

click

Um dos mais usados é o click, quando o usuário clicar no elemento, ele realizará alguma ação.

```
myStart.addEventListener('click', function(event) {

  // Faça algo aqui.

}, false);
```

Os parâmetros do **addEventListener()** são:

1. O tipo de evento que você deseja (nesse exemplo eu usei o *'click'*).
2. A função de callback
3. O *useCapture* faz o evento se atrelar ao pai até chegar ao filho. Ele por padrão é *false*, mas caso você coloque-o como *true*, ele vai fazer o caminho contrário atrelando o elemento. Você quase sempre vai usá-lo como *false*.

Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br

select

Esse evento serve para quando você quiser disparar algo quando o elemento for selecionado. Nesse caso apenas disparamos um simples *alert*.

```
myStart.addEventListener('select', function(event) {

    alert('Elemento selecionado!');

}, false);
```

Esses são uns dos mais usados, temos outros vários eventos que você pode usar, por exemplo, eventos de *drag & drop*, quando o usuário começar a arrastar um elemento você pode realizar uma ação, e quando ele soltá-lo você pode realizar outra.

Percorrer elementos

Você consegue percorrer elementos pelo DOM, usando algumas propriedades que vamos ver agora. É possível retornar com elas: elementos, comentários, textos.

.childNodes

Essa propriedade retorna uma *nodeList* de filhos do elemento passado. Ela vai retornar textos, comentários e quebras de linhas também. Portanto muito cuidado ao usá-la

```
var container = document.querySelector('.container');

var getContainerChilds = container.childNodes;
```

.firstChild

Essa propriedade retorna o primeiro filho do elemento passado.

```
var container = document.querySelector('.container');

var getFirstChild = container.firstChild;
```

.nodeName

Essa propriedade retorna o name do elemento passado. Como no caso passamos uma div, ele irá nos retornar "_div_".

```
var container = document.querySelector('.container');

var getName = container.nodeName;
```

.nodeValue

Específico para textos e comentários, ele retorna o conteúdo do nó de texto ou comentário. Como passamos ele para uma *div*, ele irá nos retornar *null*.

Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br

```
var container = document.querySelector('.container')

var getValue = container.nodeValue;
```

.nodeType

Essa propriedade nos retorna o tipo do elemento passado. Nesse caso ele retornaria '*1*'.

```
var container = document.querySelector('.container')

var getValue = container.nodeType;
```

Mas o que significa esse '_1_'? Ele simplesmente é o *nodeType* do elemento, no caso ele é um _ELEMENT_NODE _e retorna null, caso fosse um atributo por exemplo ele era um nodeType '2' e retornaria o valor do atributo.

Node type		nodeName returns	nodeValue returns
1	Element	element name	null
2	Attr	attribute name	attribute value
3	Text	#text	content of node
4	CDATASection	#cdata-section	content of node
5	EntityReference	entity reference name	null
6	Entity	entity name	null
7	ProcessingInstruction	target	content of node
8	Comment	#comment	comment text

Você pode ler mais sobre nodeTypes aqui.

Elements

Essas propriedades, ao contrário das outras acima retornam somente elementos. São mais recomendadas pois podem causar menos confusão e são mais simples de serem entendidas.

.parentNode

Essa propriedade retorna o parente do nó passado.

```
var container = document.querySelector('.container')

var getParent = container.parentNode;
```

.firstElementChild

Essa propriedade retorna o primeiro elemento-filho do elemento especificado.

```
var container = document.querySelector('.container')

var getValue = container.firstElementChild;
```


Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br

.lastElementChild

Semelhante a propriedade anterior, mas essa retorna o último elemento-filho do elemento especificado.

```
var container = document.querySelector('.container')

var getValue = container.lastElementChild;
```

Referências

Lima, M. (s.d.). *Guia do ES6*. Fonte: Medium: https://medium.com/@matheusml/o-guia-do-es6-tudo-que-voc%C3%AA-precisa-saber-8c287876325f

MDN. (s.d.). *https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Fun%C3%A7%C3%B5es*. Fonte: MDN web docs.

Prof. Caique Zaneti Kirilo & Prof. Priscila Medeiro
ESCOLA SENAI DE INFORMÁTICA - Al. Barão de Limeira, 539 - Santa Cecília - São Paulo/SP - CEP 01202-001 Telefone: (11) 3273-5000 |E-mail:informatica@sp.senai.br