

## DOM - Document Object Model (Modelo de Objetos para Documentos)

Antes de iniciar vamos criar uma estrutura HTML simples no Visual Studio Code, para que a explicação fique mais fácil em modo visual.

- 1- Crie um arquivo chamado Exercicio9.html
- 2- Inicie um HTML simples
- 3- Coloque o Título como DOM - JavaScript
- 4- De os seguintes estilos ao body: fundo - blue, cor – yellowgreen e fonte normal 18 pontos Arial
- 5- Crie um título – Entendendo o DOM
- 6- Crie um parágrafo – Postando aqui os resultados
- 7- Crie outro parágrafo – Aprendendo a usar o DOM em JavaScript e destaque o DOM com tag em HTML
- 8- Crie uma div – Clique aqui!!!
- 9- Abra área para JavaScript

O que é DOM?

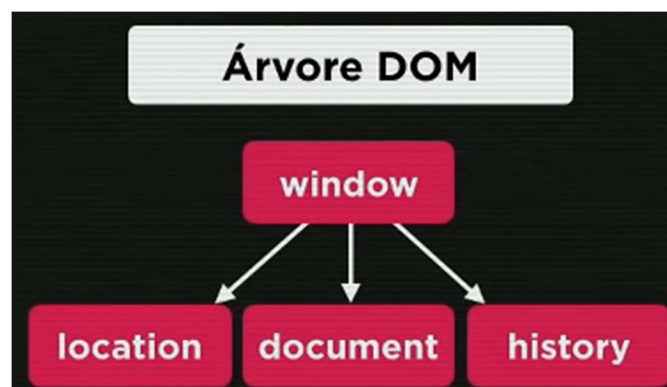
É um acrônimo para **Document Object Model**, que é o modelo de objetos para documentos, é um conjunto de objetos dentro do seu navegador que vai dar acesso aos componentes internos do seu website.

O DOM não funciona dentro do Node.js, ele está presente quando eu estou rodando JavaScript dentro navegador.

É muito importante que você saiba fazer a sua árvore DOM do seu site.

A árvore DOM começa da raiz, que dentro do navegador chamamos de **window**, tudo dentro do JavaScript está dentro de um objeto chamado window, aquela janela do seu navegador é um objeto DOM

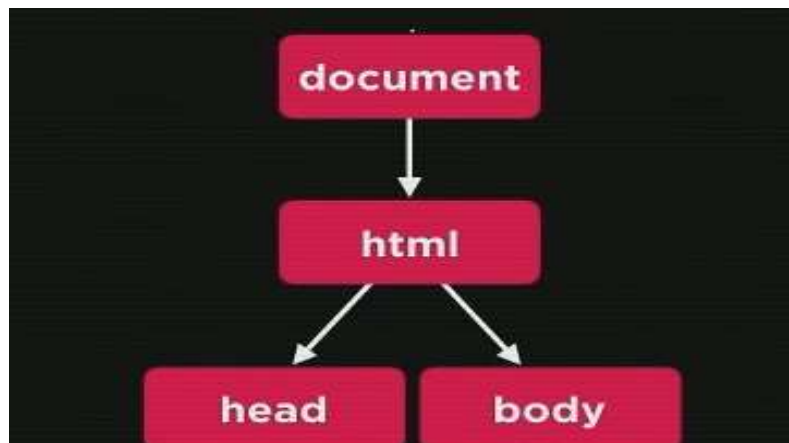
Dentro do **window** existem vários outros objetos, vamos representar apenas três exemplos:



Temos o objeto **location**, que diz qual é a localização do seu site, qual é a URL, qual é a página atual, qual foi a página anterior, o **document**, que é o documento atual e o **history**, que vai guardando de onde você veio, para onde você vai, isso facilita a navegação dentro do seu site.

Mais à frente vou demonstrar tudo o que tem dentro de windows.

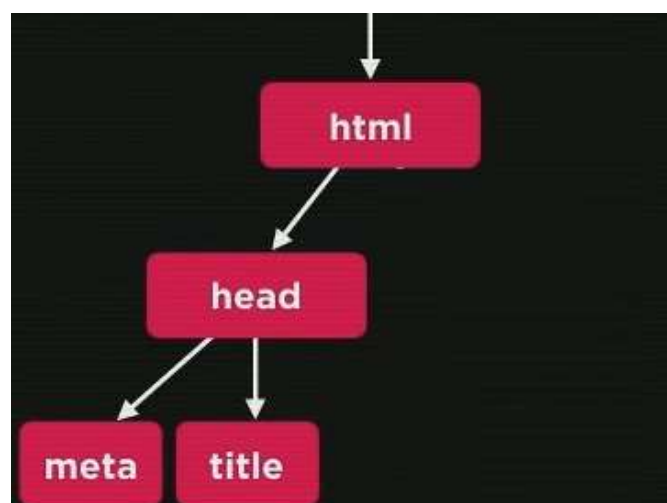
Dentro do **document**, existe um outro objeto muito importante, que é o objeto **HTML**, que é exatamente a parte HTML do site, dentro de HTML eu tenho basicamente dois objetos, o **child**(dois filhos), que são o **head** e o **body**, a parte de cabeçalho e a parte de corpo.



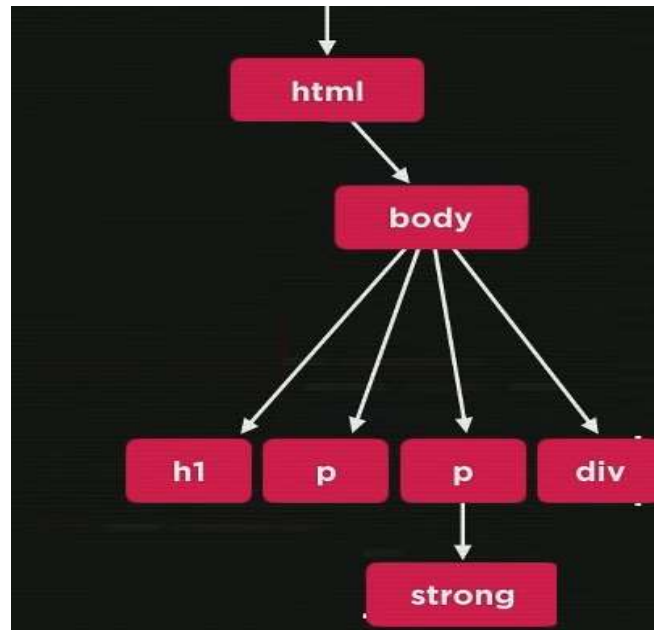
É importante dizer que, **head** e **body**, é **filho de HTML**, são **child**, já por sua vez, o **HTML** é um **parent**, é um **pai/mãe**, de **head** e **body**.

Quem está **embaixo** é **child**, quem está **acima** é **parent**.

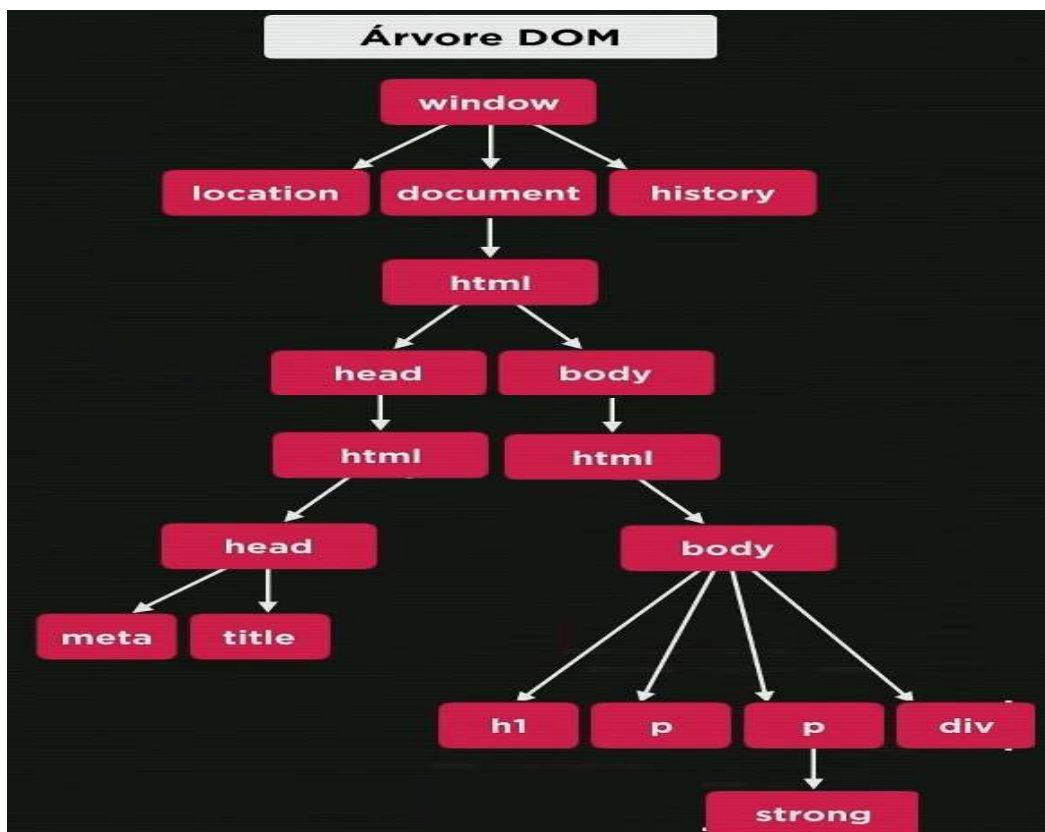
Em nosso documento, dentro de **head**, temos várias **tags**, **meta**, **title** e outras que podemos adicionar em nosso documento.



Já dentro do **body**, para esse documento que estamos em desenvolvimento, temos um **h1**, dois **parágrafos** e uma **div** e dentro do segundo parágrafo, um filho que é o **strong**.



Então com isso acabamos de criar a árvore hierárquica do site que acabamos de criar.



É importante que você consiga montar em sua cabeça essa árvore, porque vamos precisar acessar os seus componentes.

Ver exemplos no Visual Studio Code.

Assim conseguimos ter acesso a vários componentes utilizando diretamente DOM, dentro do JavaScript, assim podemos navegar dentro da árvore, da maneira que achar melhor.

E para isso existem várias maneiras de navegar entre os elementos, posso selecionar esses elementos para poder navegar dentro da minha árvore DOM. E existem vários métodos para isso.

Existem 5 métodos de acesso, podemos acessar por:

1. Por Marca (tag) – **getElementsByTagName()[]**;
2. por ID - **getElementById()**;
3. por Nome - **getElementsByName()[]**;
4. por Classe - **getElementsByClassName()[]**;
5. por seletor – **querySelector()** – **querySelectorAll()**, (por CSS, que é um recurso mais recente. Ele não está presente em todas as versões do ECMAScript, mas nas mais recentes, todos os navegadores atualizados já têm acesso a esse outro método aqui que é por seletor).

Em primeiro lugar, vamos aprender a fazer a primeira seleção, como selecionar por marca ou por tag name.

O comando está dentro de document e é o método:

### **getElementsByTagName()**

É importante que você saiba que quando você usa o **getElementsByTagName()**, você consegue selecionar mais de um objeto, porque existem vários objetos do mesmo tipo, da mesma tag.

Por exemplo, o nosso código nós temos dois parágrafos, tenho duas tags <p>, mas tenho só uma tag <body>, só uma tag <div>, só uma tag <h1>.

E como vou selecionar os meus objetos aqui? Veja no Visual Studio Code:

Vamos criar uma variável e essa variável vai se chamar p1, que vai ser meu primeiro parágrafo, e para selecionar os parágrafos, vamos utilizar window.document ou simplesmente só document, ai digitar getE e ele já mostra as opções, seleciono o **getElementsByTagName()**, dentro dos parênteses, entre aspas simples ou duplas, colocar a tag que eu quero selecionar.

```
var p1 = window.document.getElementsByTagName('p')
```

Mas ele não vai pegar um elemento só, note que o comando está no plural **Elements**, e tenho dois parágrafos, para selecionar o primeiro é preciso adicionar depois desse parênteses sem dar espaço, abre e fecha colchete dentro colocar um número, que neste caso será zero, que representa o primeiro parágrafo.

```
var p1 = window.document.getElementsByTagName('p')[0]
```

Se quiser selecionar o segundo, eu vou usar no lugar de zero, vou colocar 1 e assim sucessivamente.

Vamos escrever na tela o utilizando o innerText, ele pega texto que está dentro do primeiro parágrafo.

```
window.document.write('Esse é o texto: ' + p1.innerText)
```

Assim ele replica o mesmo parágrafo original

Outro por exemplo, vamos modificar o estilo do meu p1, vamos modificar para color black.

```
p1.style.color = 'black'
```

Podemos criar uma variável para o corpo do site e acessamos com window.document.body para modificar a cor de fundo a qualquer momento.

```
var corpo = window.document.body  
corpo.style.background = 'green'
```

Assim podemos modificar partes diferentes.

Veja neste outro exemplo:

```
document.write(p1.innerText)
```

Veja, quando digito inner temos innerHTML e innerText. Se escolho innerText ele vai mostrar o resultado, Aprendendo a usar o DOM em JavaScript.

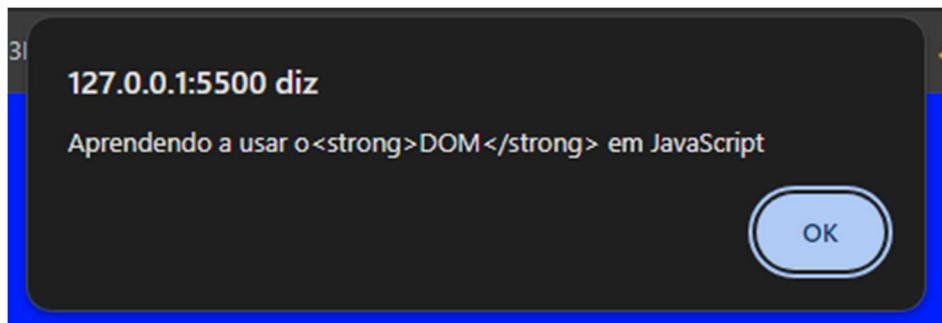
Clique aqui!!!  
Aprendendo a usar oDOM em JavaScript

Só que dentro do parágrafo não é só aprendendo a usar o DOM em JavaScript, o **DOM** está em negrito e ele não puxou o negrito, mas se quiser eu posso ver qual é o comando inteiro, em vez de innerText, digitar innerHTML assim ele vem já formatado.

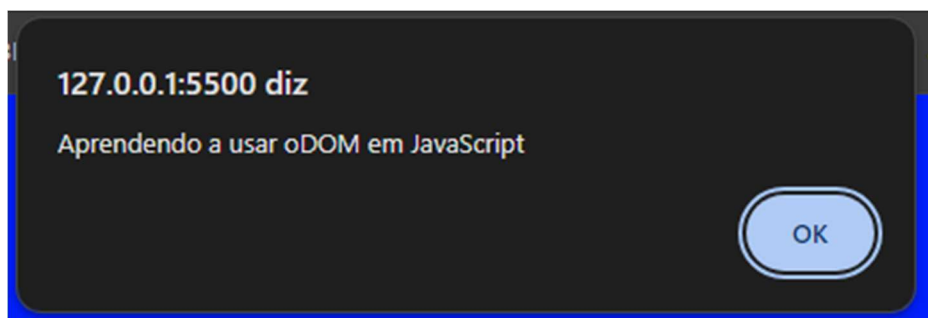
```
document.write(p1.innerHTML)
```

## Clique aqui!!! Aprendendo a usar oDOM em JavaScript

Se transformar em um alert, por exemplo, window.alert, ele já mostra, só que o DOM veio com as tags.



Em vez de utilizar a innerHTML, basta usar o innerText.



O innerText pega o texto sem as filhas, sem a formatação, ele pega simplesmente o texto o innerHTML pega o HTML inteiro, inclusive com as tags filhas.

E assim conseguimos fazer o acesso a todos os componentes utilizando **getElementsByTagName** mas essa não é única técnica.

Também podemos selecionar objetos quando o site é um pouco mais evoluído, **por ID**, e não precisamos ficar selecionando elemento por elemento.

Podemos identificar um parágrafo ou a div por um ID e usar o **getElementById()**.

Podemos utilizar o **nome** do objeto se tenho uma propriedade **name**, então vamos utilizar o **getElementsByTagName()**, vejam que o **elements** está no plural, então é preciso usar colchete quando temos mais de um objeto.

Podemos usar como conjunto por **classe**, vamos utilizar o **getElementsByClassName()**.

Vamos à alguns exemplos:

Vamos pegar a <div>, em vez de utilizar o getElementByTag, que vai ser muito genérico, às vezes teremos muitas tags <div>, isso vai nos prejudicar, criaremos um id para essa <div>, vou chamar de msg. Assim temos um id mensagem.

```
<div id="msg">Clique aqui!!!</div>
```

Vamos selecionar por id, getElementById.

```
var d = window.document.getElementById('msg')
```

Vamos fazer o background, fica verde.


```
d.style.background = 'green'
```



Clique aqui!!!

Vamos modificar o conteúdo como innerText ou innerHTML

```
d.innerText = 'Estou aguardando...'
```



Estou aguardando...

Originalmente a <div> está clique aqui!!!, mas madamos modificar por JavaScript usando DOM com o innerText, e ele escreveu estou aguardando.

Podemos fazer também, em vez de selecionar por id, selecionar por name. Fazemos a mesma coisa, na <div> o nome vai ser msg e como temos no plural elements temos que colocar o primeiro elemento no colchete.

```
<div name="msg">Clique aqui!!!</div>  
var d = window.document.getElementsByName('msg')[0]
```

```
d.innerText = 'Estou aguardando...'
```

Estou aguardando...

Vimos que conseguimos fazer a seleção por **nome**, por **id**, podemos fazer também por **classe**, que basta trocar nome por class, é tudo o mesmo princípio, a forma de acessar é a que você prefere, dependendo da situação pode ser uma, dependendo da situação pode ser outra.

```
<div class="msg">Clique aqui!!!</div>  
var d = window.document.getElementsByClassName('msg')[0]  
d.innerText = 'Estou aguardando...'
```

Estou aguardando...

E temos uma forma nova de se fazer, que é utilizando por seletor, essa forma nova é até recomendável pela maioria dos manuais, que é utilizando o **querySelector()**, e o **querySelectorAll()** sendo o plural.

Vamos aprender a utilizar o **querySelector** no lugar de utilizar qualquer uma dessas outras formas.

Vamos voltar a div para id.

```
<div id="msg">Clique aqui!!!</div>
```

E utilizar o **querySelector**.

```
var d = window.document.querySelector('div#msg')  
d.style.backgroundColor = "black"
```

Clique aqui!!!

Dentro dos parenteses, vamos usar a síntese do CSS, a **<div>** que tenho id msg.

Toda **<div>** é representada por uma hashtag (#), toda classe é representada por um ponto(.).

```
<div class="msg">Clique aqui!!!</div>  
var d = window.document.querySelector('div.msg')  
d.style.backgroundColor = "red"
```



[Clique aqui!!!](#)

Lembrando, o `querySelector` é um método mais recente. Então, navegadores mais antigos, que rodam ECMAScript, versões mais antigas, não vão ter suporte a ele.