DEV FULL STACK - JavaScript

Aula 05
DOM (Básico)

IN INFINITY SCHOOL

DEV FULL STACK - JavaScript

Aula 05: DOM (Básico)

Objetivos da aula:

1. Entender o que é o DOM e sua estrutura de funcionamento
2. Aprender a encontrar elementos HTML e CSS utilizando JS
3. Manipular dinamicamente o conteúdo de páginas com JS
4. Criar novos elementos da página com JS

Introdução

O que é DOM?

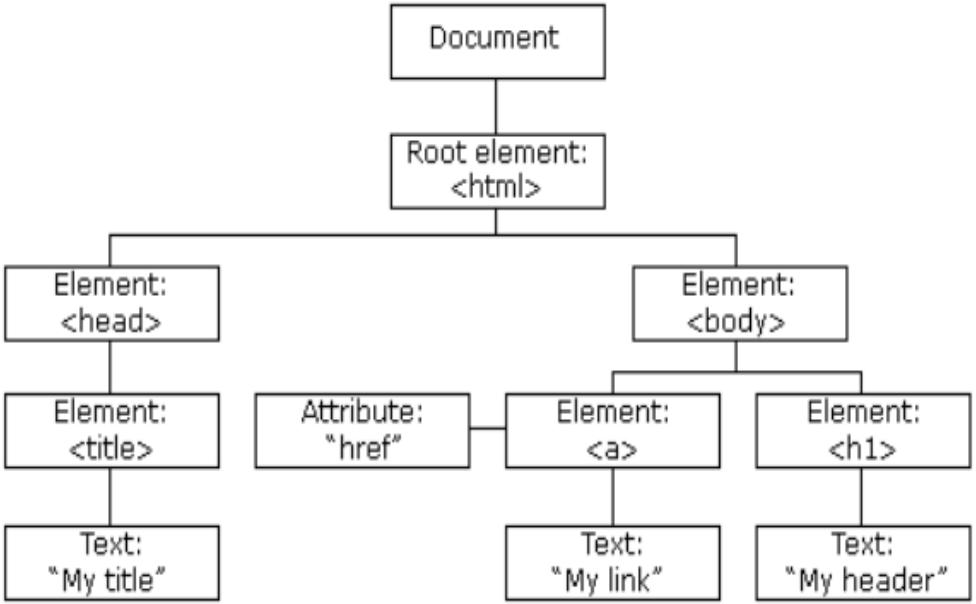
O **DOM (Document Object Model)** é a representação de dados dos objetos que compõem a estrutura e o conteúdo de um documento na Web.

Quando uma página é carregada, o navegador cria um **Document Object Model** da página contendo todos os seus elementos.

É possível utilizar o JavaScript para manipular esses elementos e suas características de forma dinâmica, durante a execução da página.

Document Object Model

O modelo HTML DOM é construído como uma **árvore de objetos**:



Documento HTML:

```
<html>
    <head>
        <title>My title</title>
    </head>
    <body>
        <h1>My header</h1>
        <a href="">My link</a>
    </body>
</html>
```

Document Object Model

O que o JavaScript consegue fazer com o DOM?

Alterar todos os **elementos** HTML na página.

Alterar todos os **atributos dos elementos** HTML na página.

Alterar todos os **estilos CSS**.

Adicionar ou remover elementos HMTL e seus atributos.

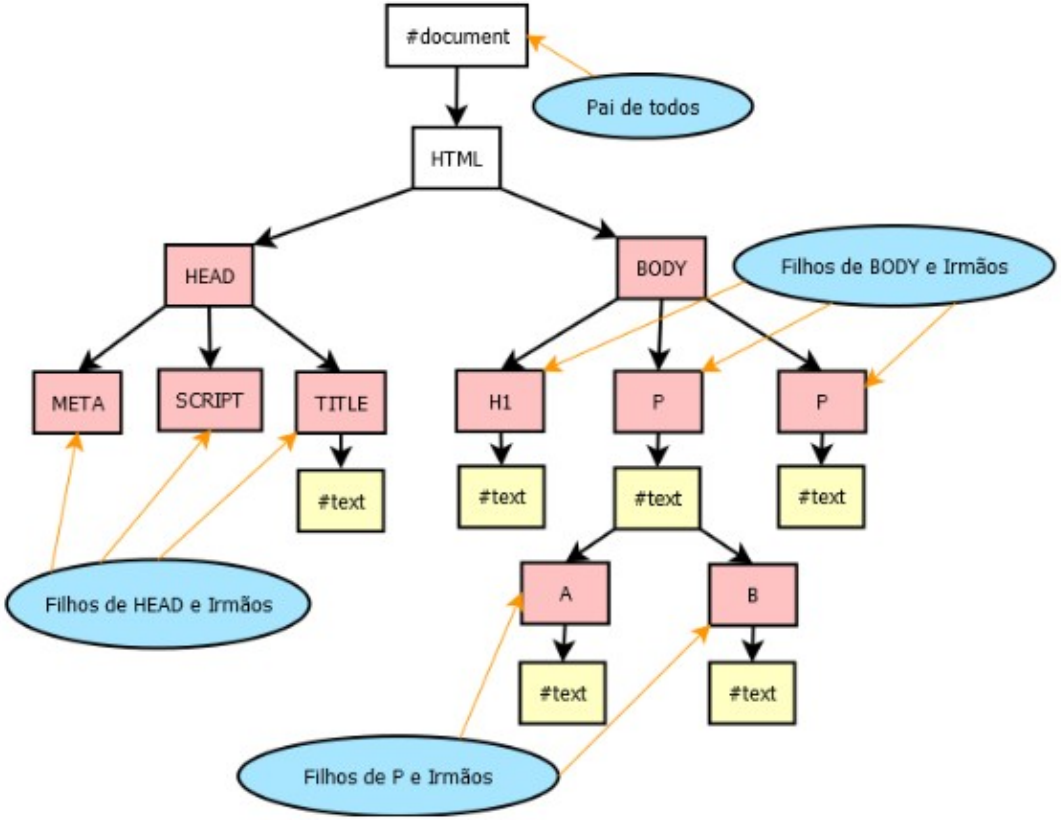
Reagir a todos os **eventos** que ocorrerem em uma página.

Criar novos eventos na pagina.

Document Object Model

Os elementos do DOM possuem relações de parentesco entre si (pais, filhos e irmãos).

Essas relações estabelecem a hierarquia entre os elementos de uma página e permitem ao JS realizar uma navegação mais organizada entre eles.



Document Object Model

Cada objeto mapeado pelo DOM possui métodos (funções) e propriedades (atributos).

Exemplo de funções de objeto do DOM que já conhecemos:

getElementById

Exemplo de propriedades de objeto do DOM que já conhecemos:

innerHTML

value

style

DOM: Encontrando Elementos

É possível encontrar elementos dentro de um documento usando funções que procuram e recuperam esses elementos por diversos critérios de busca:

- Através do ID (já visto) → **getElementById(“id”)**
- Através da tag → **getElementsByTagName(“tag”)**
- Através da classe → **getElementsByClassName(“classe”)**
- Por seletores CSS → **querySelector(“seletor”)**
- Por **coleções de objetos** → Especifica-se o **nome** da coleção

DOM: Encontrando Elementos

Através do ID do elemento (já foi visto - apenas revisando)

Função: **getElementById(“id_procurado”)**

```
1  <html>
2      <body>
3          <p id="exemplo"></p>
4          <script type="text/javascript">
5              var x = prompt("Entre com seu nome");
6              document.getElementById("exemplo").innerHTML = x;
7          </script>
8      </body>
9  </html>
```

DOM: Encontrando Elementos

Através da TAG do elemento:

Função: **getElementsByTag(“tag_procurada”)**

O parâmetro a ser passado é o **nome da tag** que se deseja buscar.

O método sempre irá retornar **um array contendo os elementos** daquela determinada tag que existirem na página.

Exemplos:

document.getElementsByTagName(“p”)

document.geteElementsByTagName(“div”)

IN INFINITY SCHOOL

Exemplo 1

Usando um script JS, vamos procurar todas as tags <p> do documento ao lado e exibir o conteúdo de cada uma delas e uma janela de alerta diferente.

Aula05_script1.js

Aula05 > Aula05_script1.js > recuperarPs

```
1  function recuperarPs(){
2      var paragrafos = document.getElementsByTagName("p");
3      var i;
4      for (i=0; i<paragrafos.length; i++){
5          alert(paragrafos[i].innerHTML);
6      }
7  }
```

Aula05_pagina_exemplo.html

Aula05 > Aula05_pagina_exemplo.html > html > body > img#img01.imagem

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>DOM - Página de Exemplo</title>
6      </head>
7      <body>
8          <h1 id="cb01" class="cabecalho">
9              Este é o Cabeçalho da Página
10         </h1>
11         <p id="prgf01" class="paragrafo">
12             Este é o primeiro parágrafo da página.
13         </p>
14         <p id="prgf02" class="paragrafo">
15             Este é o segundo parágrafo da página.<br>
16             <a href="https://infinityschool.com.br/">
17                 Infinity School
18             </a>
19         </p>
20         <img src="logo_infinty.png"
21             id="img01" class="imagem"><br>
22         <button onclick="recuperarPs()">Recuperar </button>
23
24         <script src="Aula05_script1.js"></script>
25     </body>
26 </html>
```

IN INFINITY SCHOOL

Exemplo 2

Agora vamos procurar todas as tags <p> do documento e alterar a cor de fundo de cada parágrafo toda vez que o botão for pressionado.

Aula05_script1.js

Aula05 > Aula05_script1.js > recuperarPs

```
1  function recuperarPs(){
2      var paragrafos = document.getElementsByTagName("p");
3      var i;
4      for (i=0; i<paragrafos.length; i++){
5          alert(paragrafos[i].innerHTML);
6      }
7  }
```

Aula05_pagina_exemplo.html

Aula05 > Aula05_pagina_exemplo.html > html > body > img#img01.imagem

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>DOM - Página de Exemplo</title>
6      </head>
7      <body>
8          <h1 id="cb01" class="cabecalho">
9              Este é o Cabeçalho da Página
10         </h1>
11         <p id="prgf01" class="paragrafo">
12             Este é o primeiro parágrafo da página.
13         </p>
14         <p id="prgf02" class="paragrafo">
15             Este é o segundo parágrafo da página.<br>
16             <a href="https://infinityschool.com.br/">
17                 Infinity School
18             </a>
19         </p>
20         <img src="logo_infinty.png"
21             id="img01" class="imagem"><br>
22         <button onclick="recuperarPs()">Recuperar </button>
23
24         <script src="Aula05_script1.js"></script>
25     </body>
26 </html>
```

IN INFINITY SCHOOL

ATIVIDADE 1

Utilizando como base o código HTML ao lado, crie uma página de teste e um script JS que seja capaz de **alterar a cor das letras de todos os textos da classe "mudacor" para vermelho e transformá-los em negrito**.

DICA: Agora você não está procurando uma TAG e sim uma CLASSE, logo, deve usar outra função para a tarefa: getElementsByClassName()

```
<html>
  <head>
    <meta charset="UTF-8">
    <title>DOM – Atividade 1</title>
  </head>
  <body>
    <p id="prgf01" class="mudacor">
      Este é o primeiro parágrafo da página.</p>
    <p id="prgf02" class="naomuda">
      Este é o segundo parágrafo da página.</p>
    <p id="prgf03" class="mudacor">
      Este é o terceiro parágrafo da página.</p>
    <p id="prgf04" class="mudacor">
      Este é o quarto parágrafo da página.</p>
    <p id="prgf05" class="naomuda">
      Este é o quinto parágrafo da página.</p>
    <button onclick="mudarPs()">
        Mudar Cor</button>
    <script src="scriptAtiv01.js"></script>
  </body>
</html>
```

IN INFINITY SCHOOL

Exemplo 3

Desta vez, vamos mudar a classe de parágrafos da classe “normal” para “az” ou “vm” (dependendo do botão apertado) e isso vai mudar a formatação deles.

Aula05_pagina_exemplo3.html

Aula05 > Aula05_pagina_exemplo3.html > html > body > script

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>DOM - Exemplo 3</title>
        <style>
            .normal { color: black}
            .az { color: blue}
            .vm { color: red}
        </style>
    </head>
    <body>
        <p class="normal">Este é um parágrafo 1.</p>
        <p class="normal">Este é um parágrafo 2.</p>
        <p class="normal">Este é um parágrafo 3.</p>
        <p class="normal">Este é um parágrafo 4.</p>
        <button onclick="mudarPs('az')">AZUL</button>
        <button onclick="mudarPs('vm')">VERMMELHO</button>
        <script src="Aula05_script3.js"></script>
    </body>
</html>
```

Aula05_script3.js

Aula05 > Aula05_script3.js > mudarPs

```
function mudarPs(clsNova){
    var paragrafos = document.getElementsByClassName("normal");
    var i;
    while (paragrafos.length>0){
        paragrafos[0].className = clsNova;
    }
}
```

Notou algo diferente na lógica do script JS?

Exemplo 3

Explicando a diferença:

Desta vez optamos por mudar a lógica de repetição (de **for** para **while**) porque quando alteramos a classe de um dos itens do array **paragrafos**, esse item deixa de fazer parte do array, pois ele não é mais parte da resposta da seleção de elementos da classe “normal”.

Se usássemos um for, ele se perderia na varredura do array porque os limites do loop não mudariam automaticamente com a mudança no array. Foi preciso mudar a estratégia de loop para garantir que o mesmo segue executando corretamente sem sofrer influência da mudança no conteúdo do Array.

Usando **while** e mudando sempre apenas a primeira ocorrência do array, o loop continua enquanto houverem elementos da classe “normal”.

Aula05_script3.js

Aula05 > Aula05_script3.js > mudarPs

```
1  function mudarPs(clsNova){
2      var paragrafos = document.getElementsByClassName("normal");
3      var i;
4      while (paragrafos.length>0){
5          paragrafos[0].className = clsNova;
6      }
7  }
```

IN INFINITY SCHOOL

ATIVIDADE 2 - Desafio

Você consegue alterar o script JS do E**xemplo 3** para que a página consiga alternar entre letras vermelhas e azuis ao apertar dos botões quantas vezes o usuário desejar sem que a página precise ser recarregada a cada mudança?

DICA: Primeiramente tente identificar o motivo do script JS original não conseguir fazer isso. Em seguida, discuta com seus colegas que solução poderia contornar o problema.

Usando Seletores CSS

Para encontrar elementos usando seletores CSS:

Função: **querySelector(“seletor”)**

Este método retorna o **primeiro element** que combine com o padrão.

É possível especificar um ou mais seletores CSS, mas para múltiplos seletores, deve-se usar vírgulas para separação.

Exemplos:

document.querySelector(“p.oculto”)

Veja lista de seletores CSS:

http://www.w3schools.com/cssref/css_selectors.asp

Usando Seletores CSS

E se eu quiser usar seletores CSS para selecionar TODOS os elementos que seguem um padrão?

Função: **querySelectorAll(“seletor”)**

Este método retorna o **um array contendo TODOS os elements** que combine com o padrão.

Lembre-se:

Caso seja especificado em seletor que não encontre nenhum elemento, não ocorrerá erro no script. Nesses casos, tanto o **querySelector** quanto o **querySelectorAll** apenas retornarão **null**.

IN INFINITY SCHOOL

Usando Seletores CSS

Exemplo:

Aula05_script4.js

Aula05 > Aula05_script4.js > mudarPs

```
function mudarPs(){
    var prg = document.querySelector("p");
    prg.className = ("nao_muda");
    var variosParag =
        document.querySelectorAll("p.normal");
    var i;
    for (i=0; i<variosParag.length; i++){
        variosParag[i].style =
            ("color:red; font-weight:bold;");
    }
}
```

Aula05_pagina_exemplo4.html

Aula05 > Aula05_pagina_exemplo4.html > ...

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>DOM - Exemplo 4</title>
        <style>
            .normal { color: black;}
            p:hover { background-color: #ffff00;}
        </style>
    </head>
    <body>
        <p class="normal">Este é um parágrafo 1.</p>
        <p class="normal">Este é um parágrafo 2.</p>
        <p class="normal">Este é um parágrafo 3.</p>
        <p class="normal">Este é um parágrafo 4.</p>
        <button onclick="mudarPs()">MUDAR</button>
        <script src="Aula05_script4.js"></script>
    </body>
</html>
```

Criando Novos Elementos

Há mais de uma função que permite a inclusão de novos elementos em uma página:

Função: **document.write(“algo”);**

Este método escreve “algo” na página. Pode ser uma tag, um texto ou qualquer outra coisa. O comportamento depende do documento estar aberto ou fechado.

Função: **document.createElement(elemento);**

Este método cria um elemento de documento que pode ser inserido na página.

Importante: O elemento criado não é colocado na página. Apenas está disponível para ser manipulado pelo JS. Para incluir o elemento na página é preciso usar uma função **appendChild**..

Criando Novos Elementos

Exemplo:

Aula05_pagina_exemplo6.html

Aula05 > Aula05_pagina_exemplo6.html > ...

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>DOM - Exemplo 6</title>
6      </head>
7      <body>
8          <div id="bloco">
9              <p>Este é um parágrafo normal que
10             já existe na página originalmente.</p>
11         </div>
12         <button onclick="mudarPs()">MUDAR</button>
13         <script src="Aula05_script6.js"></script>
14     </body>
15 </html>
```

Aula05_script6.js

Aula05 > Aula05_script6.js > mudarPs

```
1  function mudarPs(){
2      novoP = document.createElement("p");
3      novoP.innerHTML = "Este é um novo paragrafo "+
4                        "criado pelo botão.";
5      document.getElementById("bloco").appendChild(novoP);
6  }
```

Criando Novos Elementos

Um exemplo mais complexo:

Página:

```
<html>
      <body>
      <p>Este é um parágrafo
            que já existe na
            página original.</p>
      <button
            onclick="novoParag()">
           NOVO</button>
      <script
            src="meuScript.js">
      </script>
   </body>
</html>
```

meuScrip.js:

```
<script>
function novoParag() {
     var parag =
           document.createElement("p");
     var texto =
           document.createTextNode(
           "Este é um novo paragrafo.");
     parag.appendChild(texto);
     document.getElementById("local").append
     Child(parag);
     }
     </script>
```

IN INFINITY SCHOOL

ATIVIDADE 3

Utilizando o exemplo de inserção com a função **createElement** como base, crie uma página com formulário contendo um campo de texto e um botão.

Ao ser apertado, o botão dispara uma rotina em JS que insere um novo parágrafo na página contendo o que foi informado no formulário.

Importante:

Lembre-se que o funcionamento de seu script vai depender da página estar ou não totalmente carregada.

Usando Coleções

Javascript possibilita localizar coleções de objetos. Por exemplo:

document.anchors : todos os elementos <a> do documento que possuem o atributo **name**

document.body: o elemento **<body>**

document.documentElement : o próprio elemento **<html>**

document.forms : todos os formulários existentes na página (array de objetos)

document.head : o elemento **<head>**

document.images : todas as imagens **<img>** existentes na página

document.links : todos os elementos **<a>** e **<area>** que possuem o atributo **href** setado

document.scripts : todos os elementos **<script>**

IN INFINITY SCHOOL

Usando Coleções

Exemplo:

Aula05_pagina_exemplo7.html

Aula05 > Aula05_pagina_exemplo7.html > ...

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>DOM - Exemplo 7</title>
6      </head>
7      <body>
8          <p>O botão vai acrescentar um novo link:</p>
9          <a name="gg" href="http://google.com">
10             Google</a><br>
11         <a name="ms" href="http://microsoft.com">
12             Microsoft</a><br>
13         <a name="fb" href="http://facebook.com">
               Facebook</a><br><br>
           <button onclick="maisUmLink()">
               INCLUIR LINK</button><br>
           <script src="Aula05_script7.js"></script>
       </body>
   </html>
```

Aula05_script7.js

Aula05 > Aula05_script7.js > maisUmLink

```
1  function maisUmLink(){
2      var novoElem = document.createElement("a");
3      novoElem.innerHTML = "Infinity";
4      novoElem.href = "https://infinityschool.com.br/"
5      document.body.appendChild(novoElem);
6  }
```

Referências

Aqui estão alguns links interessantes para leitura complementar:

Uma introdução ao DOM (Documento Object Model):

https://developer.mozilla.org/pt-BR/docs/Web/API/Document_Object_Model/Introduction

Diversos exemplos de manipulação do DOM com JavaScript:

http://www.w3big.com/pt/js/js-ex-dom.html