INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

AULA:

Document Object Model (DOM)

Disciplina: Programação de Sistemas para Internet

Alba Lopes, Profa.

http://docentes.ifrn.edu.br/albalopes
alba.lopes@ifrn.edu.br

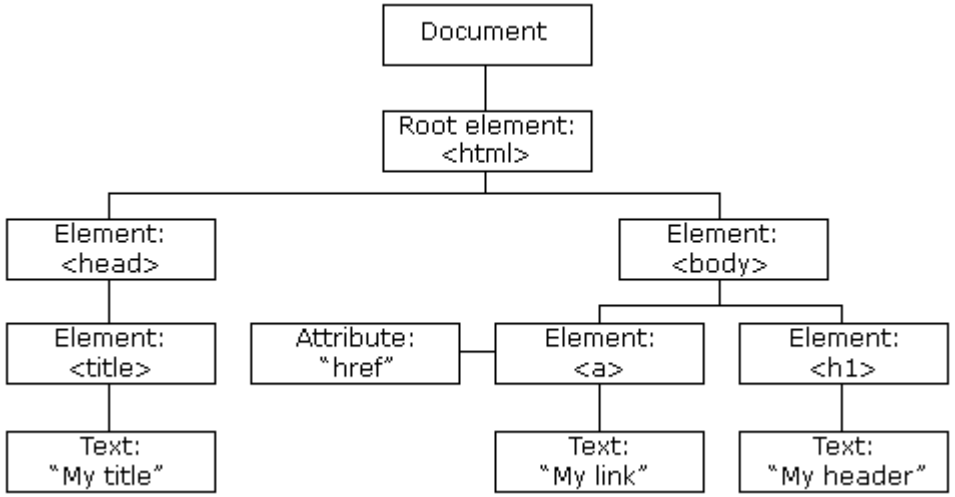
Agenda

- Document Object Model
  - Introdução
  - Encontrando elementos por tag
  - Encontrando elementos por classe
  - Encontrando elementos por seletor CSS
  - Criando elementos no documento
  - Exercícios

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Document Object Model

- Quando uma página é carregada, o navegador cria um **D**ocument **O**bject **M**odel da página.
- O modelo **HTML DOM** é construído como uma árvore de objetos (**Objects**):



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Document Object Model

- Com o modelo de objeto, JavaScript tem todo o poder necessário para criar HTML dinamicamente:
  - JavaScript pode alterar todos os elementos HTML na página
  - JavaScript pode alterar todos os atributos dos elementos HTML na página
  - JavaScript pode alterar todos os estilos CSS
  - JavaScript pode remover um elemento HMTL e seus atributos
  - JavaScript pode adicionar um elemento HMTL e seus atributos
  - JavaScript pode reagir a todos os eventos que ocorrerem em uma página
  - JavaScript pode criar novos eventos na page

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Document Object Model

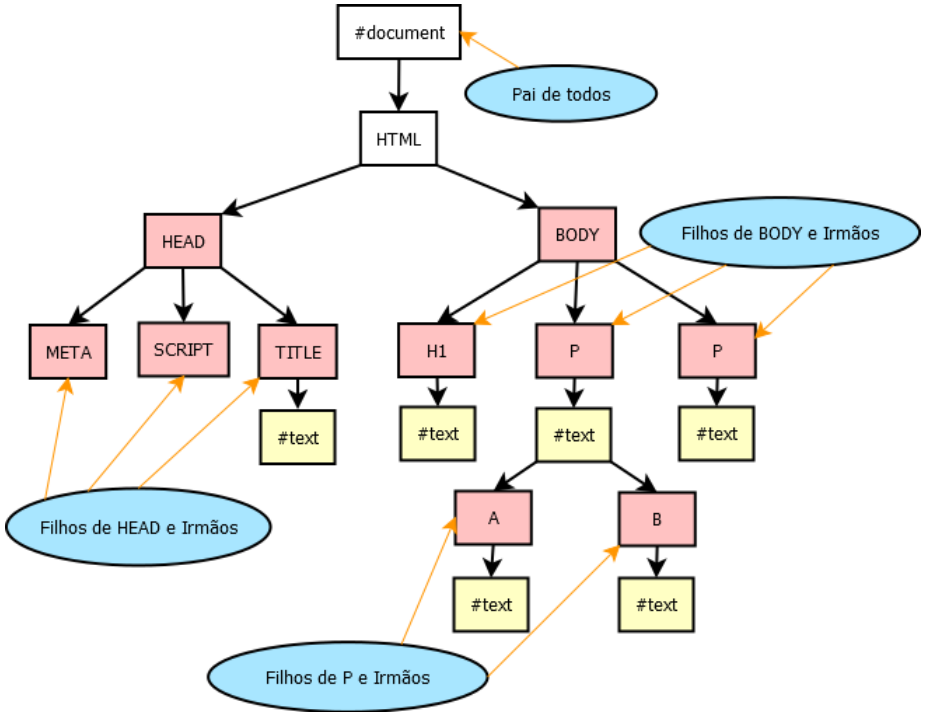
- Seja o código HTML a seguir:

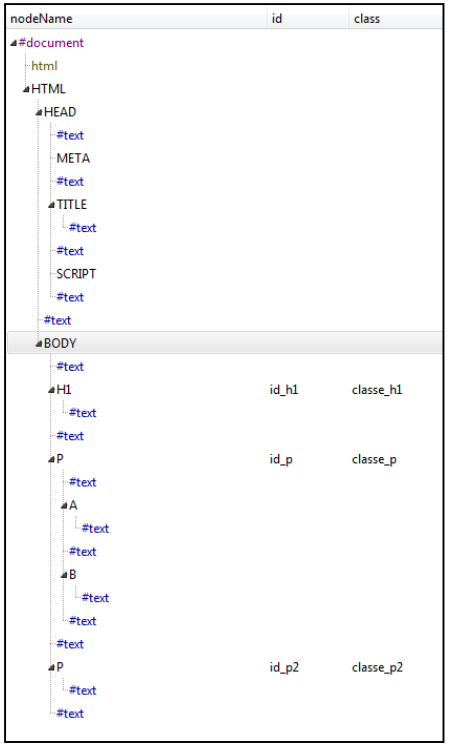
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>DOM</title>
        <script src="meuscript.js"></script>
    </head>

    <body>
        <h1 id="id_h1" class="classe_h1">Sou um cabeçalho!</h1>
        <p id="id_p1" class="classe_p">
            Um texto qualquer dentro de uma tag de parágrafo. Aqui também
            temos outras tags, como <a href="#">um link<a>, ou um texto
            <b>em negrito</b>.
        </p>
        <img src="#" id="id_imagem" class="classe_imagem">
        <p id="id_p2" class="classe_p">
            Este é outro parágrafo.
        </p>
    </body>
</html>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Document Object Model

- Existem elementos pai (**parent**), filhos (**childs**) e irmãos (**siblings**). Estes elementos são caracterizados na forma como estão na árvore, veja o exemplo na imagem abaixo:





Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Document Object Model

- Enquanto objeto, possui métodos (funções) e propriedades (atributos)
  - Funções já conhecidas do objeto **document**:
    - **getElementById**
    - **write**
    - **addEventListener**
  - Propriedades já conhecidas do objeto **document:**
    - **innerHTML**
    - **value**
    - **style**

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos na página

- Através do ID (como já vimos)
- Através da tag
- Através da classe
- Por seletores CSS
- Por coleções de objetos

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da tag

- Para localizar um elemento pelo nome da tag,utiliza-se o método **getElementsByTagName**.
- 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 houverem na página.

- **getElementsByTagName**
  - Para parágrafos, passa-se o parâmetro **p**
    - **document.getElementsByTagName("p")**
  - Para divs, passa-se o parâmetro **div**
    - **document.geteElementsByTagName("div")**
  - E assim por diante

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da tag

- Exemplo 1: *recuperar* e *exibir conteúdo de todos os parágrafos da página*

HTML

```
<!DOCTYPE html>
<html>
     <head>
          <meta charset="UTF-8">
          <title>DOM</title>
          <script src="meuscript.js"></script>
     </head>

     <body>
          <h1 id="id_h1" class="classe_h1">Mundo Mágico do Javascript</h1>
          <p id="corvinal" class="classe_p">Corvinal</p>
          <p id="grifinoria" class="classe_p">Grifinória</p>
          <p id="sonserina" class="classe_p">Sonserina</p>
          <p id="lufalufa" class="classe_p">Lufa-Lufa</p>
          <button type="button" id="bdefinir" onclick="recuperarPs()">Recuperar </button>
     </body>
</html>
```

JAVASCRIPT

meuscript.js

```
function recuperarPs(){
     var paragrafos = document.getElementsByTagName("p");
     var i;
     while (i=0; i<paragrafos.length; i++){
          alert(paragrafos[i].innerHTML);
     }
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da tag

- Exemplo 2: *alterar propriedade dos parágrafos*

HTML

```
<!DOCTYPE html>
<html>
      <head>
            <meta charset="UTF-8">
            <title>DOM</title>
            <script src="meuscript.js"></script>
      </head>

      <body>
            <h1 id="id_h1" class="classe_h1">Mundo Mágico do Javascript</h1>
            <p id="corvinal" class="classe_p">Corvinal</p>
            <p id="grifinoria" class="classe_p">Grifinória</p>
            <p id="sonserina" class="classe_p">Sonserina</p>
            <p id="lufalufa" class="classe_p">Lufa-Lufa</p>
            <button type="button" id="bdefinir" onclick="definirCores()">Definir Cores</button>
      </body>
</html>
```

JAVASCRIPT

meuscript.js

```
function definirCores(){
      var paragrafos = document.getElementsByTagName("p");
      var cores = ["blue", "red", "green", "yellow"];
      var i;
      for (i=0; i<paragrafos.length; i++){
            paragrafos[i].style.backgroundColor = cores[i];
      }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da classe

- Para localizar um elemento utilizando o nome da classe que deseja, utiliza-se o método **getElementsByClassName**.
- O parâmetro a ser passado é o nome da classe que se deseja buscar
- O método sempre irá retornar um array contendo todos os elementos daquela determinada classe que houverem na página.
- **getElementsByClassName**
  - Por exemplo, seja uma classe chamada **oculto**, definida no CSS que faz com que os objetos fiquem ocultos (display: none)
  - Deseja-se buscar todos esses elementos e transformá-los em visíveis.

    getElementsByClassName('oculto')

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da classe

- Exemplo 1: *alterando o estilo do elemento*

HTML

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="UTF-8">
		<title>DOM</title>
		<link rel="stylesheet" href="estilo.css"/>
		<script src="meuscript.js"></script>
	</head>

	<body>
		<h1 id="id_h1" class="classe_h1">Mundo Mágico do Javascript</h1>
		<p id="corvinal" class="oculto">Corvinal</p>
		<p id="grifinoria" class="oculto">Grifinória</p>
		<p id="sonserina" class="oculto">Sonserina</p>
		<p id="lufalufa" class="oculto">Lufa-Lufa</p>
		<button type="button" id="bdefinir" onclick="aparecer()">Aparecium </button>
	</body>
</html>
```

CSS

estilo.css

```
.oculto{
	display: block;
}
```

JAVASCRIPT

meuscript.js

```
function aparecer(){
	var paragrafos = document.getElementsByClassName("oculto");
	var i;
	for (i=0; i<paragrafos.length; i++){
		paragrafos[i].style.display = "block";
	}
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da classe

- Exemplo 2: *removendo a classe do elemento (forma 1)*

meuscript.js

JAVASCRIPT

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

Como a classe é removida do elemento, ele não fará mais parte do array de elementos daquela clase. Logo, a cada elemento que possui sua classe removida, o número de elementos no array paragrafos diminui.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da classe

- Exemplo 2: *removendo a classe do elemento (forma 2)*

HTML

```
<!DOCTYPE html>
<html>
     <head>
          <meta charset="UTF-8">
          <title>DOM</title>
          <link rel="stylesheet" href="estilo.css"/>
          <script src="meuscript.js"></script>
     </head>
     <body>
          <h1 id="id_h1" class="classe_h1">Mundo Mágico do Javascript</h1>
          <p id="corvinal" class="oculto">Corvinal</p>
          <p id="grifinoria" class="oculto">Grifinória</p>
          <p id="sonserina" class="oculto">Sonserina</p>
          <p id="lufalufa" class="oculto">Lufa-Lufa</p>
          <button type="button" id="bdefinir" onclick="aparecer()">Aparecium </button>
     </body>
</html>
```

JAVASCRIPT

```
function aparecer(){
     var paragrafos = document.getElementsByClassName("oculto");
     var i;
     while (paragrafos.length > 0){
          paragrafos[i].classList.remove("oculto");
     }
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através da classe

- Exemplo 3: *adicionando uma classe a um elemento*

HTML

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="UTF-8">
		<title>DOM</title>
		<link rel="stylesheet" href="estilo.css"/>
		<script src="meuscript.js"></script>
	</head>
	<body>
		<h1 id="id_h1" class="classe_h1">Mundo Mágico do Javascript</h1>
		<p id="corvinal" class="oculto">Corvinal</p>
		<p id="grifinoria" class="oculto">Grifinória</p>
		<p id="sonserina" class="oculto">Sonserina</p>
		<p id="lufalufa" class="oculto">Lufa-Lufa</p>
		<button type="button" id="bdefinir" onclick="aparecer()">Aparecium </button>
	</body>
</html>
```

CSS

estilo.css

```
.oculto{
	display: block;
}

.corvinal{
	background-color: blue;
}
.grifinoria{
	background-color: red;
}
.lufalufa{
	background-color: yellow;
}
.sonserina{
	background-color: green;
}
```

JAVASCRIPT

```
function aparecer(){
	var paragrafos = document.getElementsByClassName("p");
	var i;
	while (paragrafos.length > 0){
		paragrafos[i].classList.add(paragrafos[i].id);
		paragrafos[i].classList.remove("oculto");
	}
}
```

Nesse exemplo, faça o add antes do remove. Pois após o remove, o elemento não estará mais no array parágrafos.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através de seletores CSS

- Para localizar um elemento utilizando um seletor CSS utiliza-se o método **querySelector**.
- O método retorna **o primeiro** element que combine com o padrão.
- Deve-se especificar um ou mais seletores CSS
- Para múltiplos seletores, separe através de vírgulas.
- Exemplos:
  - document.querySelector(“p”)
  - document.querySelector(“p.oculto”)
  - document.querySelector(“[type=text]”)
- Veja lista de seletores CSS: http://www.w3schools.com/cssref/css_selectors.asp

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através de seletores CSS

- Exemplo 1: *encontrando elemento que possua atributo type = button*

HTML

```
<!DOCTYPE html>
<html>
     <head>
          <meta charset="UTF-8">
          <title>DOM</title>
          <script src="meuscript.js"></script>

     </head>
     <body>
          <h1 id="id_h1" class="classe_h1">Mundo Mágico do Javascript</h1>
          <p id="corvinal" class="oculto">Corvinal</p>
          <p id="grifinoria" class="oculto">Grifinória</p>
          <p id="sonserina" class="oculto">Sonserina</p>
          <p id="lufalufa" class="oculto">Lufa-Lufa</p>
          <button type="button" id="bdefinir" onclick="aparecer()">Aparecium </button>
          <button type="button" id="bdefinir" onclick="aparecer()"> </button>
     </body>
</html>
```

JAVASCRIPT

```
function encontrar(){
     var botao = document.querySelector("[type=button]");
     botão.style.color = "red";
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através de seletores CSS

- Para localizar um elemento utilizando um seletor CSS utiliza-se o método **querySelectorAll**.
- O método um array com **todos** os elements que combinam com o padrão.
- Deve-se especificar um ou mais seletores CSS
- Para múltiplos seletores, separe através de vírgulas.
- Exemplos:
  - document.querySelector(“p”)
  - document.querySelector(“p.oculto”)
  - document.querySelector(“[type=text]”)
- Veja lista de seletores CSS: http://www.w3schools.com/cssref/css_selectors.asp

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando elementos no documento

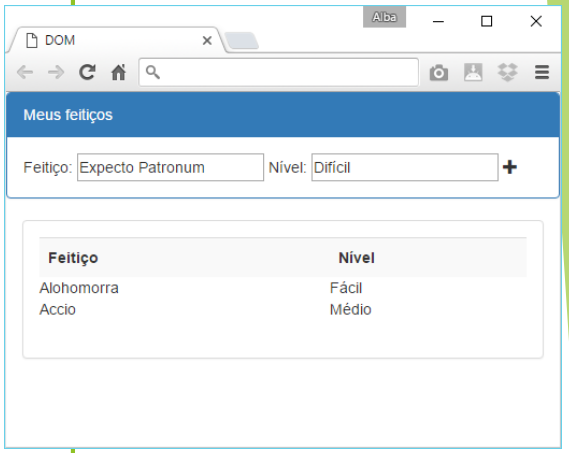
- Há ainda funções para criar e remover elementos da página, através dos métodos a seguir.
  - **document.createElement**(element)
    - cria um elemento HTML. Deve-se passar por parâmetro a tag que se deseja criar
    - o método retorna o elemento criado
    - Exemplo: **document.createElement("p") ,** cria um elemento do tipo parágrafo
  - **element.removeChild**(elementToRemove)
    - Remove o filho de um elemento
  - **element.appendChild**(newElement)
    - Adiciona um novo elemento como último filho do elemento da chamada do método
  - **document.replaceChild**(newElement, oldElement)
    - Substitui um elemento por outro
- Para outras funções: http://www.w3schools.com/js/js_htmldom_document.asp

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando elementos no documento

HTML

```
<div id="formulario" class="panel panel-primary">
      <div class="panel-heading">Meus feitiços</div>
            <div class="panel-body">
                  Feitiço: <input type="text"/>
                  Nível: <input type="text"/>
                  <span class="glyphicon glyphicon-plus" onclick="novoItem()">
                  </span>
            </div>
      </div>
</div>

<div class="container col-md-3">
      <div class="panel panel-default ">
            <div class="panel-body">
                  <table class="table table-striped" id="tabela">
                        <tr><th>Feitiço</th><th>Nível</th></tr>
                  </table>
            </div>
      </div>
</div>
```



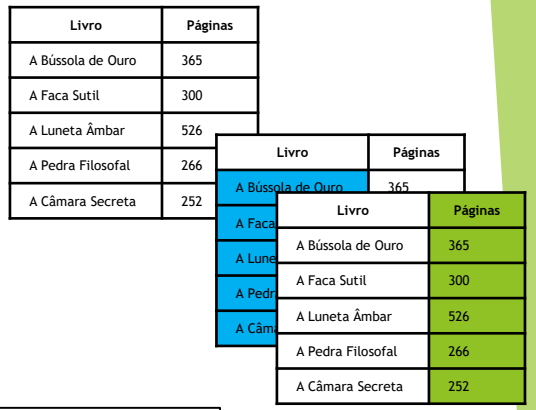
JAVASCRIPT

```
function novoItem(){
      var campos = document.getElementsByTagName("input");
      var i;
      var novaLinha = document.createElement("tr");
      for (i=0; i<campos.length; i++){
            var novaColuna = document.createElement("td");
            novaColuna.innerHTML = campos[i].value;
            novaLinha.appendChild(novaColuna);
      }
      var tabela = document.getElementById("tabela");
      tabela.appendChild(novaLinha);
}
```

Exercícios

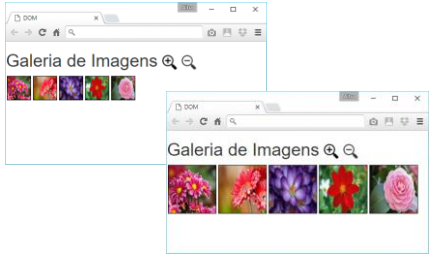
- Utilize o método **getElementsByTagName** e/ou **getElementsByClass,** além do que você já aprendeu de Javascript para resolver os problemas a seguir:

1. Crie uma tabela contendo o título de alguns livros em uma coluna e a quantidade de páginas em outra, como mostrado ao lado. Ao clicar na coluna com o nome “Livro”, altere a cor do plano de fundo das células dessa coluna para azul. Ao clicar na coluna Páginas, altere as cores das células dessa coluna para verde. Uma coluna não pode ficar pintada quando a outra estiver.



2. Crie um formulário com 10 campos de texto e um botão. Nesses campos de texto, o usuário deve digitar os itens de uma lista de compras. Ao clicar no botão, os itens devem ser exibidos em uma lista ordenada(<ol>), logo abaixo dos formulário. Após clicar no botão, limpe os campo de texto (ex: *atribua vazio à propriedade* ***value***).

3. Crie 3 classes CSS para imagens. As classes devem definir tamanhos diferentes para as imagens (ex: uma classe para imagens 50x50, outra classe para imagens 100x100 e outra para 150x150). Crie uma página contendo 10 imagens quaisquer e uma imagem de uma lupa pra zoom in (+) e outra para zoom out (-). Ao clicar na imagem da lupa zoom in, altere a classe das imagens para a classe que possua um maior tamanho de imagem. Ao clicar na lupa zoom out, altere a classe para uma que possua o tamanho de imagem menor.



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Encontrando elementos através de coleções

- Javascript possibilita localizar coleções de objetos dos seguintes elementos:
  - 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.embeds : todos os elementos incorporados **<embed>**
  - 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**>
  - document.title **:** o elemento <title>

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

REFERÊNCIAS

- [1] W3C School. JavaScript Tutorial. Disponível em: http://www.w3schools.com/js/
- http://www.w3schools.com/js/js_htmldom_elements.asp
- [2] MORISSON,Michael. Java Script – Use a Cabeça. Ed. 2. Rio de Janeiro: Altabooks
- [3] Manzano, José; Toledo, Suely. Guia de Orientação e Desenvolvimento de Sites – HTML, XHTML, CSS e JavaScript / Jscript. 2a. Edição

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br