JavaScript – HTML DOM

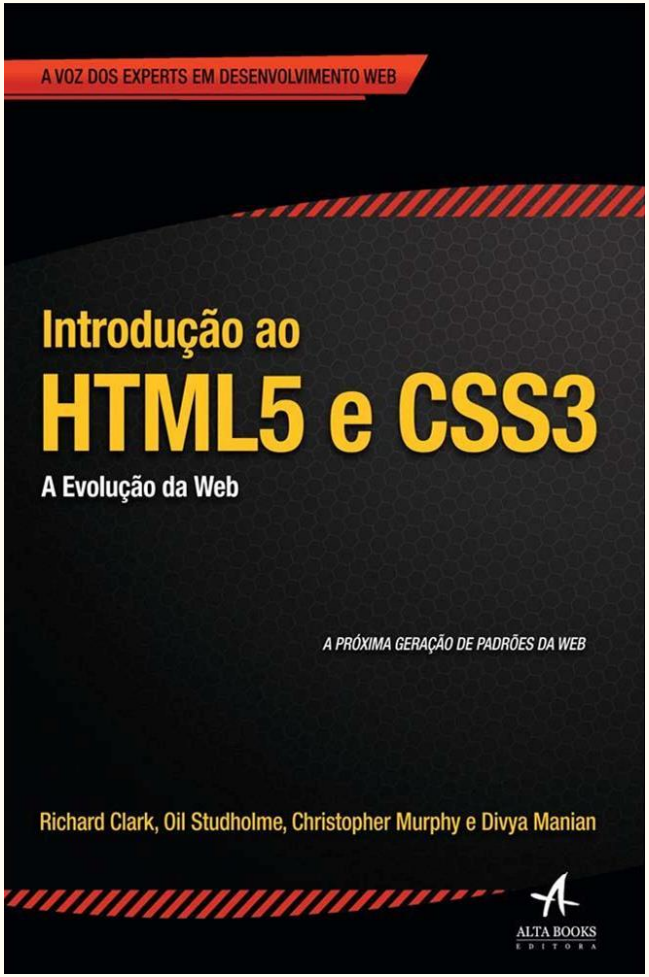
—

Prof. Victor Farias

V 1.2

Referências

w3schools.com



Introdução

JavaScript

- Linguagem de Programação padrão para páginas WEB
- JavaScript define o comportamento da página
- JavaScript tem várias utilidades:
  - Mudar conteúdo de um elemento
  - Mudar atributos HTML
  - Mudar estilos CSS
  - Esconder elementos
  - Mostrar elementos
  - Adicionar elementos

Tag <script>

- Código JavaScript deve ser colocado dentro de um elemento **script**
- Pode ser colocado no <head> ou no <body>
- É preferível colocar os scripts no final da página, pois carregamento e compilação podem atrasar a renderização da página

```
<script>

document.getElementById("demo").innerHTML = "My First JavaScript";

</script>
```

Tag <script>

- Também é possível importar um .js externo

```
<script src="myScript.js"></script>
```

- Vantagens:
  - Separa código JS e HTML
  - Facilita manutenção
  - Navegador pode deixar .js em cache

Sintaxe JavaScript

HTML DOM

HTML DOM

- DOM = Document Object Model
- DOM define um padrão para acessar documentos
- HTML DOM é o modelo padrão e interface de programação para HTML
- Define:
  - Elementos HTML como objetos
  - A propriedades de elementos HTML
  - Métodos para acessar elementos HTML
  - Eventos para todos os elementos HTML

Recuperando elemento HTML

Elementos DOM

- Recuperar elemento por id
  - Método **getElementById()**
  - Se elemento não existir, myElement será nulo

```
let myElement = document.getElementById("intro");
```

- Recuperar elementos por nome da tag
  - Método `getElementsByTagName()`

```
let x = document.getElementById("main");

let y = x.getElementsByTagName("p");

// Recupera elemento com id=”main” e, depois, recupera todos elementos <p> dentro dele
```

Elementos DOM

- Recuperar elementos por classe
  - Método `getElementsByClassName()`

```
let x = document.getElementsByClassName("intro");
```

- Recuperar elemento por seletor CSS
  - Método **querySelectorAll()**

```
let x = document.querySelectorAll("p.intro");
```

Manipulando o HTML

Manipulando HTML

- **Propriedade innerHTML**
  - `Muda conteúdo HTML de um elemento HTML`

```
document.getElementById("p1").innerHTML = "New text!";
```

- **Manipulando atributos HTML**
  - `document.getElementById(id).attribute=new value`

```
document.getElementById("myImage").src =
"landscape.jpg";
```

Manipulando CSS

Manipulando CSS

- Mudando CSS
  - `document.getElementById(id).style.property=new style`

```
document.getElementById("p2").style.color = "blue";
```

- Nomes da propriedades CSS compostos separados por hífen viram CamelCase no JS
  - propriedade background-color -> object.style.backgroundColor="#00FF00"

Manipulando Classes

Manipulando Classes

- **API classList**
- **Adicionando classe**
  - el.classList.add('classOne', 'classTwo');
- **Removendo classe**
  - el.classList.remove('classOne');
- **Verificar existência de classe**
  - if(el.classList.contains('classFour') == true){...}
- **Alternar classe**
  - el.classList.toggle('classThree')

Eventos HTML

Eventos HTML

- Código JS pode ser disparado a partir de eventos
  - Quando um usuário clica o mouse
  - Quando a página é carregada
  - Quando uma imagem é carregada
  - Quando um mouse passa sobre um elemento
  - Quando um campo de entrada muda
  - Quando um formulário HTML é submetido
  - Quando o usuário aperta uma teclas

Eventos HTML

- onclick - no clique
  - HTML - `<button onclick="displayDate()">Try it</button>`
  - JS - `document.getElementById("myBtn").onclick = displayDate;`
- onload - no carregamento
  - `<body onload="checkCookies()">`
- onchange - mudança de estado (muito usado em fomulários)
  - `<input type="text" id="fname" onchange="upperCase()">`
- onmouseover - quando o mouse se desloca sobre o elemento
- onmouseout - quando o mouse se desloca para fora do elemento
- onmousedown - quando o botão do mouse é pressionado
- onmouseup - quando o botão do mouse é solto
- onclick - quando a ação de clicar em um elemento é concluída

HTML DOM EventListener

- É possível adicionar *listeners* de eventos
  - `element.addEventListener(event, useCapture);`
  - `ex: element.addEventListener("click", function(){ alert("Hello World!"); });`
  - `Nota: Não usar prefixo “on”, use “click” no lugar de “onclick”`
- Também é possível remover *listeners*
  - `element.removeEventListener("mousemove", myFunction);`

Criando Novos Nós

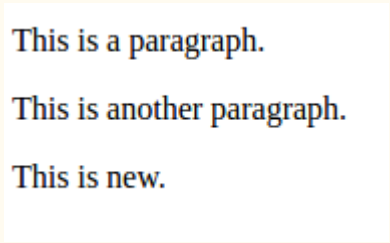
Nós HTML DOM

- Criar novos nós HTML.
  - `document.createElement(“p”)` para criar nó de elemento
  - `document.createTextNode("This is new.");` para criar nó de texto
  - `appendChild(node);` para adicionar elemento a um elemento já existente

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
let para = document.createElement("p");
let node = document.createTextNode("This is new.");
para.appendChild(node);

let element = document.getElementById("div1");
element.appendChild(para);
</script>
```



Exemplos

- Image slider
- Accordeon com clique
- Modal
- Menu hambúrguer reponsivo push
- Menu hambúrguer reponsivo overlay
- Navbar lateral overlay
- Navbar lateral push
- Navbar lateral com opacidade
- Lista de elementos com formulário

Perguntas?

Prof. Victor Farias