

Javascript – Módulo 2 – B7Web

Material de estudo desenvolvido por:
Daniel Teixeira Quadros

Sumário

Aula 01 – O Que É DOM.....	2
Aulas 02 até 04 – Seleccionando Elementos.....	2
Aula 05 – Manipulando Class do Elemento 1.....	2
Aula 06 – Manipulando Class do Elemento 2.....	3
Aula 07 – Tipos de Dados.....	3
Aula 08 – Comentários.....	3
Aula 09 – Criação e execução de Funções.....	4
Aula 10 – Parâmetros de Funções.....	4
Aula 11 – Manipulando Atributos.....	4
Aula 12 – Dimensões.....	4
Aula 13 – Distâncias e Scroll Suave.....	5
Aula 14 – Resposta do Exercício Scroll.....	7

Aula 01 – O Que É DOM

Aulas 02 até 04 – Selecionando Elementos

```
document.getElementById('exemplo').innerHTML = "Pedro <button>Botão</button>"
document.getElementsByClassName('lista')[1].innerHTML = 'Ítem alterado'
document.getElementsByTagName('button')
document.getElementsByTagName('input')
document.getElementsByTagName('div')
document.getElementsByName('email')
document.getElementsByName('telefone')
```

```
document.querySelector('#exemplo')// pelo ID
document.querySelector('.lista')//pela Claas -> apenas o 1º elemento encontrado
document.querySelectorAll('.lista')//pela Class -> Todos os elementos
document.querySelector('button')// pela Tag

//# -> para ID
//. -> para Classe Obs: Seleciona apenas o primeiro elemento que ele encontrar
// para selecionar todos, é necessário utilizar o comando
// querySelectorAll
//-> Se não adicionarmos nada na chamada, podemos chamar pela Tag
```

Aula 05 – Manipulando Class do Elemento 1

```
function verde(){
    document.querySelector('#exemplo').classList.remove('vermelho')
    document.querySelector('#exemplo').classList.remove('azul')
    document.querySelector('#exemplo').classList.add('verde')
    //classList retorna o objeto que controla as classes do elemento específico
    //neste exemplo, ele está buscando lá do nosso CSS 'estilo as configurações
    //do elemento 'verde'
    //Parâmetro '.add' adiciona o elemento, mas não substitui o elemento que já estava lá
    // para isto, utilizamos o parâmetro '.remove'
}

function vermelho(){
    document.querySelector('#exemplo').classList.remove('verde')
    document.querySelector('#exemplo').classList.remove('azul')
    document.querySelector('#exemplo').classList.add('vermelho')
}

function azul(){
    document.querySelector('#exemplo').classList.remove('vermelho')
    document.querySelector('#exemplo').classList.remove('verde')
    document.querySelector('#exemplo').classList.add('azul')
}
```

```

m02a05 > # estilo.css > .verde
1  #exemplo {
2      width: 200px;
3      height: 200px;
4      margin: 20px;
5      border: 1px solid #ccc;
6  }
7
8  .verde{
9      background-color: #0f0;
10     color: blue;
11 }
12 .vermelho{
13     background-color: #f00;
14     color: yellow;
15 }
16 .azul{
17     background-color: #00f;
18     color: white;
19 }

```

Aula 06 – Manipulando Class do Elemento 2

```

m02a06 > JS script.js > trocar
1  function trocar(){
2      //document.querySelector('button').classList.contains('preto')
3      // 'contains' função que procura se tem aquela Class no elemento específico
4      // o comando 'contains' retorna 'true' ou 'false'
5      if(document.querySelector('button').classList.contains('preto')){
6          document.querySelector('button').classList.remove('preto')
7          document.querySelector('button').classList.add('verde')
8      }else{
9          document.querySelector('button').classList.remove('verde')
10         document.querySelector('button').classList.add('preto')
11     }
12 }

```

Aula 07 – Tipos de Dados

Tipos de variáveis
Algo interessante:

```

var tipoNome = typeof nome
var tipoIdade = typeof idade

```

Uma variável armazenando o tipo de uma outra variável.

Aula 08 – Comentários

Comentários no código

Aula 09 – Criação e execução de Funções

Aula 10 – Parâmetros de Funções

Aula 11 – Manipulando Atributos

HTML:

```
<body>
  <br>
  <button onclick="trocarImagem('cachorro.jpg', 'Cachorro')">Cachorro</button>
  <button onclick="trocarImagem('gato.jpg', 'Gato')">Gato</button>
  <button onclick="pegarAnimal()">Qual é o Animal</button>
  <script type="text/javascript" src="script.js"></script>
</body>
```

JavaScript:

```
function trocarImagem(fileName, animalName){
  document.querySelector('.imagem').setAttribute('src', 'images/'+fileName)
  //selecionado o item pela class '.imagem' e utilizado o comando 'setAttribute'
  //setAttribute(configurar o atributo) -> o primeiro item dentro dos parênteses
  //mostra qual atributo
  //queremos trocar, neste caso o 'src' e o segundo atributo é qual valor
  //queremos trocar. Neste caso o 'images/'+fileName
  document.querySelector('.imagem').setAttribute('data-animal', animalName)
}
function pegarAnimal(){
  let animal = document.querySelector('.imagem').getAttribute('data-animal')
  alert(`O animal é: ${animal}`)
  // Aqui, o comando getAttribute é utilizado para pegar um determinado atributo
  // Neste exemplo, pegamos o atributo 'data-animal' e jogamos na tela com um alert
}
```

Aula 12 – Dimensões

Atributo – offset

Veja a estrutura:

HTML:

```
<body>
  <div class="texto">
    Lorem ipsum dolor sit, amet consectetur adipisicing elit. Mollitia minima
    et vel sed dolores, voluptatum incidunt, ex quam earum modi unde ut
    sapiente veniam libero tempora, perspiciatis impedit dolore expedita!
    Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ex aperiam ea
    tenetur impedit et magnam maiores veniam voluptatibus, necessitatibus,
    sapiente quas ab dolore! Aliquid aut iure sunt hic fugit dolor?
  </div>
  <script type="text/javascript" src="script.js"></script>
</body>
```

CSS:

```
m02a12 > # style.css > .texto
1  .texto{
2      width: 200px;
3      height: 200px;
4      overflow: auto; /*Se o conteúdo dentro desta DIV for maior que as medidas
5      acima, neste
6      exemplo, 200px por 200px ele pode por um scrool(barra de rolagem)*/
7      background-color: #CCC;
8      padding: 20px; /*margem*/
9  }
```

Atributo offset no navegador:

```
> document.querySelector('.texto').offsetWidth
< 200
> document.querySelector('.texto').offsetHeight
< 200
> document.querySelector('.texto').offsetWidth
< 240
> document.querySelector('.texto').offsetHeight
< 240
```

As duas primeiras linhas são o resultado apresentado antes de adicionar a linha 'padding' no CSS e as duas últimas linhas, o resultado depois de adicionado o 'padding'.

O Comando offset retorna as configurações do atributo relacionado. Neste caso class '.texto'

Obs.: O resultado engloba largura, scrollbar(dependendo do navegador), padding e borda

Atributo client:

clientWidth → pega a largura do conteúdo + o padding excluindo o scrollbar

clientHeight → pega a altura do conteúdo + o padding

```
> document.querySelector('.texto').clientWidth
< 223
> document.querySelector('.texto').clientHeight
< 240
```

Atributo scroll: → dá o tamanho real do conteúdo, ou seja, do texto

scrollWidth → largura total apenas do conteúdo, ou seja, neste exemplo, do texto

scrollHeight → altura total apenas do conteúdo, ou seja, neste exemplo, do texto

```
> document.querySelector('.texto').scrollWidth
< 223
> document.querySelector('.texto').scrollHeight
< 346
```

Aula 13 – Distâncias e Scroll Suave

Atributo scroll:

scrollTop → Mostra a posição que a scrollbar(vertical) está, a partir do topo.

scrollLeft → Mostra a posição que a scrollbar(horizontal) está, a partir da esquerda.

Navegador:

consectetur adipisicing elit.
Mollitia minima et vel sed
dolores, voluptatum
incididunt, ex quam earum
modi unde ut sapiente
veniam libero tempora,
perspiciatis impedit dolore
expedita! Lorem ipsum
dolor sit amet consectetur,
adipisicing elit. Ex aperiam
ea tenetur impedit et
magnam maiores veniam
voluptatibus, necessitatibus,
sapiente quas ab dolore!

Console:

```
> document.querySelector('.texto').scrollTop
< 89
> document.querySelector('.texto').scrollLeft
< 0
```

Scroll para a tela inteira:

window.scrollToY → Mostra a posição que a scrollbar(vertical) está, a partir do topo.

window.scrollToX → Mostra a posição que a scrollbar(horizontal) está, a partir da esquerda.

```
> window.scrollToY
< 0
> window.scrollToX
< 0
```

scrollTo()

O comando scrollTo() leva o scroll para um ponto específico da tela.

ScrollTo(0, 0) → O primeiro parâmetro é a posição 'X'(horizontal) e o segundo parâmetro é a posição 'Y'(vertical).

```
> document.querySelector('.texto').scrollTo(0,0)
< undefined
> window.scrollTo(0,0)
< undefined
```

CONFIGURANDO UM BOTÃO SCROLL NO FINAL DA PÁGINA PARA LEVAR A PÁGINA PARA O TOPO:

HTML:

```
<body>
  <div class="texto">
    Lorem ipsum dolor sit, amet consectetur adipisicing
    elit. Mollitia minima et vel sed dolores, voluptatum
    incididunt, ex quam earum modi unde ut sapiente veniam
    libero tempora, perspiciatis impedit dolore expedita!
    Lorem ipsum dolor sit amet consectetur, adipisicing
    elit. Ex aperiam ea tenetur impedit et magnam maiores
    veniam voluptatibus, necessitatibus, sapiente quas ab
    dolore! Aliquid aut iure sunt hic fugit dolor?
  </div>
  <div onclick="subirTela()" class="scrollbutton"></div>
  <script type="text/javascript" src="script.js"></script>
</body>
```

CSS:

```

1  v  .texto{
2      width: 200px;
3      height: 900px;
4  v  overflow: auto; /*Se o conteúdo dentro desta DIV for maior
    que as medidas acima, neste
5      exemplo, 200px por 200px ele pode por um scrool(barra de
    rolagem)*/
6      background-color: #CCC;
7      padding: 20px; /*margem*/
8  }
9  v  .scrollbutton{
10     width: 40px;
11     height: 40px;
12     border-radius: 15px; /* este comando deixa as bordas
    arredondadas, neste caso, com 15px deixará o botão redondo*/
13     background-color: red;
14     position: fixed; /* position em fixed deixa o botão fixo na
    tela na posição que está declarada abaixo com os comandos
15     right e bottom*/
16     right: 20px;
17     bottom: 20px;
18     z-index: 999; /* z-index com a configuração 999 deixa este
    botão sobre qualquer elemento na tela*/
19     cursor: pointer; /* para quando passar o Mouse ficar com
    aquela mãozinha*/
20 }

```

JAVASCRIPT:

```

/*function subirTela(){
    window.scrollTo(0,0) // Este comando joga o scroll para o
    topo
    // Porém, vai para o topo de uma só vez.
    //Ou seja, num salto.
}*/

function subirTela(){// aqui, faremos o exemplo de scroll suave
    window.scrollTo(0,0) // ao invés de passarmos os parâmetros X
    e Y
    // passaremos um objeto
    top: 0, //topo
    left: 0, //esquerda, que neste exemplo não temos o
    horizontal
    behavior: 'smooth' // comportamento -> smooth = suave.
    // Quem controla a velocidade de
    rolagem
    // é o próprio navegador
}

```

Aula 14 – Resposta do Exercício Scroll

SOLUÇÃO 01:

A SOLUÇÃO 1 NÃO É A MAIS EFETIVA, PORQUE ESTE MODELO FAZ COM QUE O SCRIPT FAÇA A VERIFICAÇÃO DA POSIÇÃO DO SCROLL A CADA SEGUNDO, MESMO QUE A TELA ESTEJA PARADA, GERANDO MUITO PROCESSAMENTO.

Nesta solução, copiamos os arquivos da aula 13 e modificamos apenas o nosso JavaScript:

→ Método utilizado: **Temporizador**

```
function subirTela(){
    window.scrollTo({
        top: 0,
        left: 0,
        behavior: 'smooth'
    })
}

function decBot(){
    if(window.scrollY === 0){
        // ocultar o botão
        document.querySelector('.scrollbutton').style.display = 'none'
    }else{
        //mostrar o botão
        document.querySelector('.scrollbutton').style.display = 'block'
        // ou -> document.querySelector('.scrollbutton').style.display = ''
    }
}

setInterval(decBot, 1000)//setInterval é um verificador de tempo
//onde o primeiro parâmetro indica do que verificaremos
// o tempo, neste caso, da função decBot
// e o segundo parâmetro indica em que intervalo de tempo
// verificaremos essa função. Esta declaração é em
//milissegundos. no nosso exemplo utilizamos 1000
// que é igual a 1 segundo
```

SOLUÇÃO 02:

A SOLUÇÃO 2 É MAIS EFETIVA. NESTE MODELO, O SCRIPT SÓ IRA FAZER A VERIFICAÇÃO DA POSIÇÃO DA TELA SE HOVER MOVIMENTO NA TELA. SE NÃO HOVER MOVIMENTO O SCRIPT NÃO IRA CONSUMIR PROCESSAMENTO

Nesta solução, copiamos os arquivos da aula 13 e modificamos o nosso JavaScript e o CSS:

→ Método utilizado: **Verificador de posição.**

JAVASCRIPT:


```

function subirTela(){
    window.scrollTo({
        top: 0,
        left: 0,
        behavior: 'smooth'
    })
}

function decBot(){
    if(window.scrollY === 0){
        // ocultar o botão
        document.querySelector('.scrollbutton').style.display = 'none'
    }else{
        //mostrar o botão
        document.querySelector('.scrollbutton').style.display = 'block'
        // ou -> document.querySelector('.scrollbutton').style.display = ''
    }
}

window.addEventListener('scroll', decBot)// O comando addEvent adiciona um evento
// com o complemento Listener(escutar)-> monitora
// que realiza a verificação.
//No primeiro parâmetro utilizamos o scroll
//comando já reconhecido pelo JavaScript.
//No segundo parâmetro chamamos a verificação
// que será realizada

```

CSS: A única modificação realizada aqui, foi a adição do comando display no `.scrollbutton` com a configuração 'none'

```

.texto{
    width: 200px;
    height: 900px;
    overflow: auto;
    background-color: #CCC;
    padding: 20px;
}

.scrollbutton{
    width: 40px;
    height: 40px;
    border-radius: 15px;
    background-color: red;
    position: fixed;
    right: 20px;
    bottom: 20px;
    z-index: 999;
    cursor: pointer;/* para quando passar o Mouse ficar com aquela mãozinha*/
    display: none;/* Este 'none' aqui faz com que a tela seja carregada pela
    primeira vez sem o botão, já que é carregada no topo*/
}

```