Hello World

Módulo HTML5,CSS3 E JS

Relembrando...

Javascript

1. Variáveis e tipos de dados
2. Operadores
   a. Aritmeticos
   b. Atribuição
   c. Comparação
   d. Lógicos
3. Estruturas condicionais
4. Laços
5. Funções

Sobre a aula anterior...

Dúvidas?

Javascript

Aula 09

1. Objetos

Vamos aprender o conceito de objetos em JavaScript.

Objetos

Objetos são como objetos na vida real, possuem características/propriedades e ações/habilidades

Objetos

Carros têm as mesmas **ações** (liga, desliga, acelera, freia), mas podem possuir **características** (nome, cor, modelo, peso, etc) diferentes. Pensando em carros como objetos, eles tem métodos (comuns a todos) e propriedades (variando de carro para carro);

Podemos armazenar objetos em variáveis, e acessar suas propriedades e métodos através da sintaxe de ponto.

Objetos

Properties	Methods
car.name = Fiat	car.start()
car.model = 500	car.drive()
car.weight = 850kg	car.brake()
car.color = white	car.stop()

Objetos

Objetos são variáveis que podem conter diversos valores, escritos como um conjunto de **chave : valor**

Objetos - Propriedades

Propriedades podem receber quaisquer tipos de valores, inclusive funções (métodos) do objeto!

Property	Value
firstName	John
lastName	Doe
age	50
eyeColor	blue
fullName	function() {return this.firstName + " " + this.lastName;}

Objetos - Métodos

Métodos são funções pertencentes aos objetos, atribuídos a uma propriedade.

Declarando Objetos

```
// Declarando objeto (uma forma)
var meuCarro = new Object();
meuCarro.marca = 'Fiat';
meuCarro.modelo = 'Palio';
meuCarro.ano = '2013';
meuCarro.marca = 'Passeio';
meuCarro.valor = 26.000;
```

```
// Declarando objeto (outra forma)
var carro = {
   nome: 'Fiat',
   modelo: 'Uno',
   ano: '2015',
   valor: 30.000
}
```

Objetos

Declarando objetos - com métodos

```
var carro = {
   nome: 'Fiat',
   modelo: 'Uno',
   ano: '2015',
   valor: 30.000,
   ligar: function () {
       console.log('comandos para ligar um carro - aumenta velocidade')
   },
   parar: function () {
       console.log('comandos para parar um carro - diminui velocidade até chegar a zero')
   }
}
```

Acessando Propriedades/ Métodos

Retornando os dados do objeto

Podemos acessar propriedades apenas indicando o nome dela.

Para acessar métodos é necessário colocar "(" e ")" ao final do nome do método.

```
// Retornando valores do objeto
// basta pegar o nome do objeto, usar o
ponto e colocar o nome da
//propriedade/método que deseja chamar
console.log(carro.nome);
carro.ligar();
```

Objetos - this

Quando utilizamos "this", queremos dizer que estamos acessando uma propriedade pertencente ao objeto.

Objetos - this

```
// this - Usamos quando queremos acessar propriedades que pertencem ao mesmo objeto
var luz = {
   tipo: 'led',
   ligada: false,
   ligar: function() {
       if(this.ligada === false){
           this.ligada = true;
           console.log(`Ação de ligar a luz: OK - Luz ligada`);
       }else{
           console.log('Sem ação, luz já está ligada')
       }
   },
   desligar: function() {
       if(this.ligada === true){
           this.ligada = false;
           console.log(`Ação de desligar a luz: OK - Luz desligada`);
       }else{
           console.log('Sem ação, luz já está desligada')
       }
   }
}
```

Time to code!

Aula08/exercicio01.html

Time to code

---

1. Crie um objeto chamado "boleto", que deve possuir as propriedades "valor", "diaVencimento" e "mesVencimento".
2. Crie um segundo objeto chamado "pessoa", que possui as propriedades "nome" e "valorDisponivel". Além disso, esse objeto deve possuir um método "pagarBoleto", que recebe como parâmetro um "boleto", e altera o "valorDisponível" da pessoa para o restante que ela possuirá após pagara o boleto. Além disso, deverá retornar a mensagem "Boleto de valor <valor do boleto> foi pago por <nome da pessoa> e agora restam <valor restante da pessoa>. #TaPago"*.

* <valor do boleto>, <nome da pessoa> e <valor restante da pessoa> devem ser substituídos pelos valores dos objetos referidos.

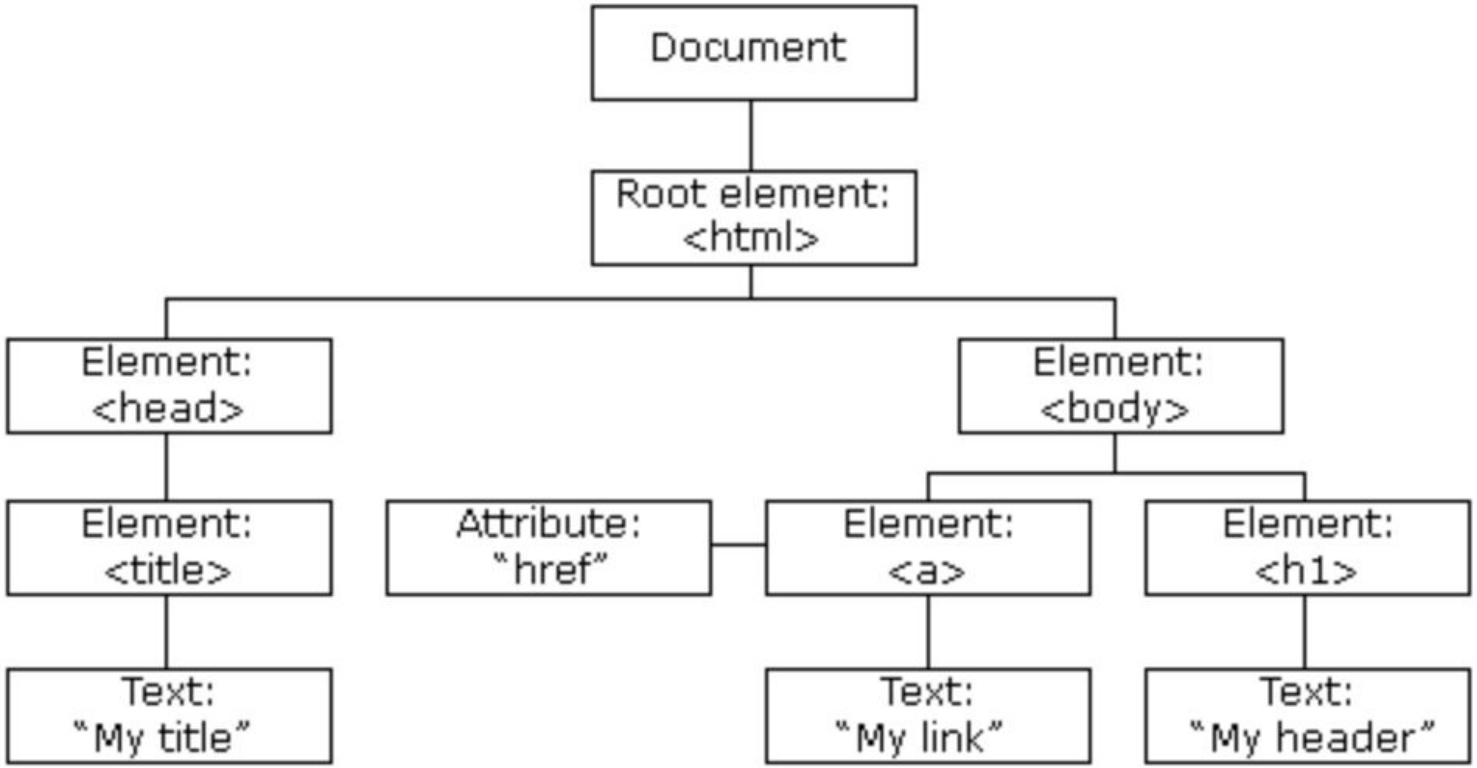
2. DOM

Vamos aprender sobre Document Object Model e como usar isto com JS.

DOM

Document Object Model é um modelo criado assim que cada página do browser é aberta. Possuí uma árvore de objetos padronizada e acessíveis via JavaScript.

The HTML DOM Tree of Objects



DOM - Interface de programção

No DOM, todos os elementos são considerados objetos. Logo, todos eles possuem métodos e atributos.

Métodos

Para iniciarmos a nossa junção de JavaScript com HTML, vamos introduzir alguns métodos bastante utilizados.

- **getElementById()**: retorna o objeto do DOM que possui o id passado por parâmetro.
- **innerHTML**: retorna o conteúdo de um objeto HTML ou insere um conteúdo dentro se um objeto HTML

```
document.getElementById("demo").innerHTML = "Hello World!";
```

getElementById()

Retorna o elemento que tiver o nome do ID passado. Como IDs devem ser únicos, ele só retorna um elemento.

É possível armazenar o elemento numa variável, para ficar mais fácil de manipular o elemento

```
// getElementById
var secaoAbout = document.getElementById('about');
secaoAbout.style.background = 'red';
```

innerHTML

Quando precisamos pegar o que tem dentro daquele elemento, ou queremos adicionar alguma outra coisa nele;

```
// Inserindo no HTML
secaoAbout.innerHTML = '<h1>Ola mundo</h1>';
```

addEventListener()

Os elementos do DOM também possuem eventos, e podemos atribuir eventos aos elementos utilizando o método **addEventListener()** juntamente do elemento que deseja atribuir algum evento. O mais comum é o evento de "click"

```
// Eventos
var linkMenuAbout = document.getElementById('link-about');
linkMenuAbout.addEventListener('click', () => {
  // para este funcionar, a variavel "secaoAbout"precisa existir
  secaoAbout.style.background = 'green';
});
```

Métodos

Buscando elementos no DOM:

Method	Description
document.getElementById(*id*)	Find an element by element id
document.getElementsByTagName(*name*)	Find elements by tag name
document.getElementsByClassName(*name*)	Find elements by class name

Alterando elementos do DOM:

Method	Description
element.innerHTML = *new html content*	Change the inner HTML of an element
element.attribute = new value	Change the attribute value of an HTML element
element.setAttribute*(attribute, value)*	Change the attribute value of an HTML element
element.style.*property = new style*	Change the style of an HTML element

Métodos

Adicionando / deletando elementos do DOM:

Method	Description
document.createElement(*element*)	Create an HTML element
document.removeChild(*element*)	Remove an HTML element
document.appendChild(*element*)	Add an HTML element
document.replaceChild(*element*)	Replace an HTML element
document.write(*text*)	Write into the HTML output stream

Adicionando *handlers* para eventos:

Method	Description
document.getElementById(*id*).onclick = function(){*code*}	Adding event handler code to an onclick event

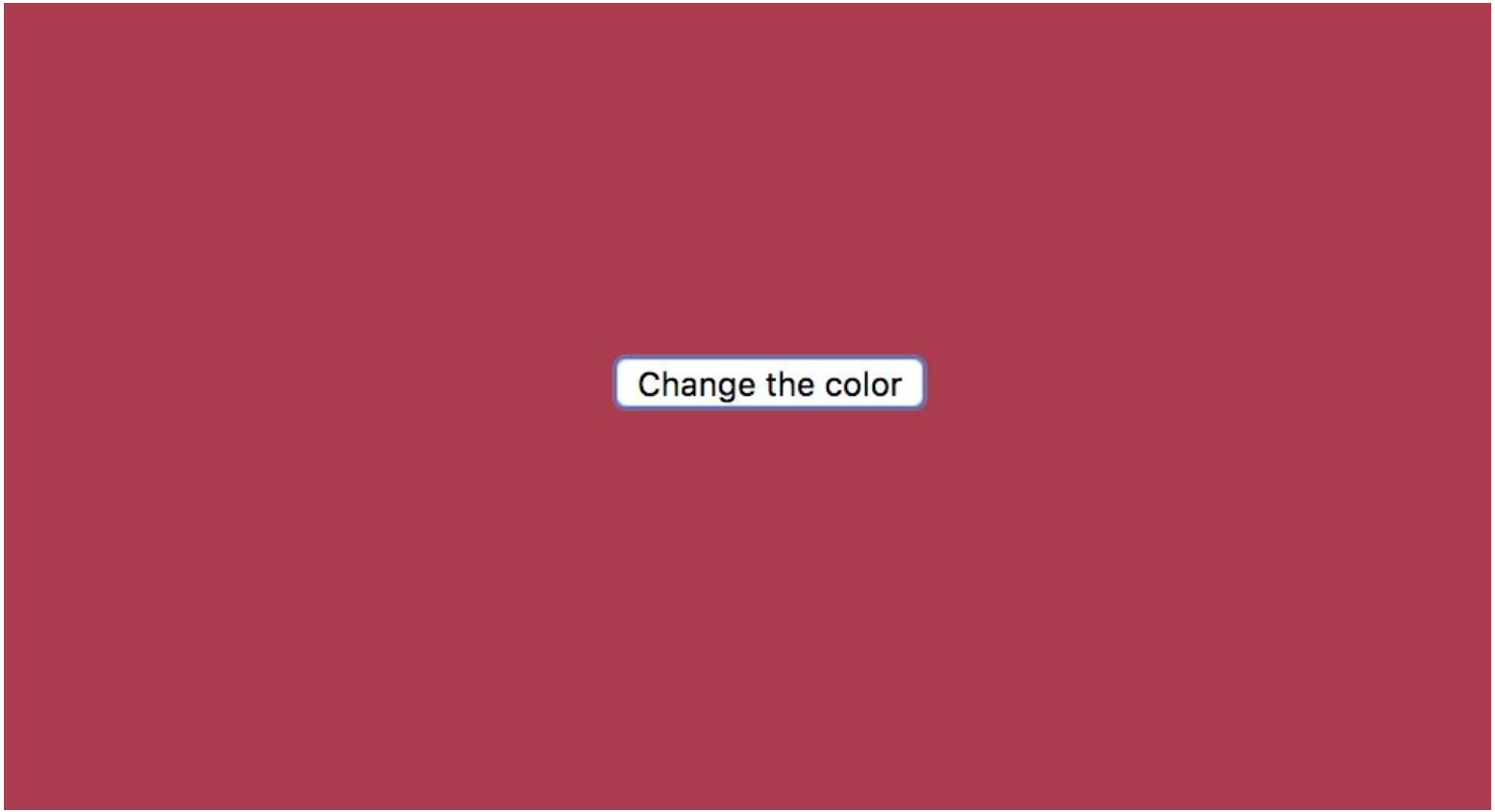
Time to code!

Aula08/exercicio02.html

Time to code!

---

1. Utilizando o CodePen, crie uma página que conterá:
    a. Um botão centralizado
    b. A cada vez que esse botão for clicado, a cor de fundo deverá mudar
2. Porém… NÃO PODEM UTILIZAR CÓDIGO HTML OU CSS! É UM EXERCÍCIO APENAS DE JAVASCRIPT!


Change the color

Time to code!

---

1. Para facilitar:
    a. As cores randômicas de background podem ser calculadas com rgb.
        i. Lembrem-se que o JavaScript aceita valores textuais, neste caso.
        ii. Procurem por Math.random e Math.floor para gerar valores randômicos num intervalo de valores.

Dúvidas?

Contato

github.com/talitaoliveira

litaa.olivera@gmail.com

linkedin.com/in/litaaoliveira

@liitacherry

Conteúdo baseado nos slides de:
Diogo Cabral - github.com/drcabral