

# JavaScript



**ANA CAROLINA N R GRACIOSO**

[carol.nrg@gmail.com](mailto:carol.nrg@gmail.com)

# Agenda

- Arrays;
- Objetos: propriedades, métodos e classes.

# Array

- Assim como em outras linguagens de programação, o JavaScript dá suporte a manipular uma estrutura de dados que armazena uma coleção de elementos.
- Essa estrutura de dados também é conhecida como variável indexada, vetor (para arranjos unidimensionais) e matriz (para arranjos bidimensionais).

# Array

- Criando Arrays:

```
var cars = new Array("Honda", "Volvo", "BMW");
```

ou simplesmente:

```
var cars = ["Honda", "Volvo", "BMW"];
```

# Array

- Acessando Arrays:

```
document.write(cars[0]);
```

```
document.write(cars);
```

# Array

- Em JavaScript, Arrays são um tipo particular de objeto:

```
var pessoa = ["Fulano", "de Tal", 28];
```

```
document.write("Nome: "+pessoa[0]+"<br>");
```

```
document.write("Sobrenome: "+pessoa[1]+"<br>");
```

```
document.write("Idade: "+pessoa[2]+"<br>");
```

# Array

- Porém, é preferível acessar os objetos por nomes do que por índices

```
var pessoa = {  
    primeiroNome: "Fulano",  
    sobreNome: "de Tal",  
    idade: 28};
```

```
document.write("Nome:  
"+pessoa.primeiroNome+"<br>");  
document.write("Sobrenome:  
"+pessoa.sobreNome+"<br>");  
document.write("Idade: "+pessoa.idade+"<br>");
```

# Array

- Elementos de um Array:

```
myArray[0] = Date.now; // objeto  
myArray[1] = myFunction; // função  
myArray[2] = myCars; // outro array
```



# Array

- Propriedades e Métodos:
  - length: tamanho do array;
  - sort(): ordena o array;
  - toString(): converte um array em string;
  - join(" \* "): converte em string permitindo especificar o separador;

# Array

- Métodos:
  - push(): adiciona elementos no final do array e retorna o novo tamanho;
  - pop(): remove elementos do final do array e retorna o elemento removido;
  - shift(): remove elementos do início do array;
  - unshifts(): adiciona elementos no início do array;

# Array

- Excluindo elementos:

```
delete frutas[0];
```

- Emendando um array:

```
var frutas= ["Banana", "Laranja", "Manga"];  
frutas.slice(2, 0, "Limão", "Kiwi");
```

2 - posição de inserção

0 - quantidade de elementos a serem excluídos

# Alterando o conteúdo de um elemento HTML

```
<!DOCTYPE html>
<html>
<body>
<p id="demo" onclick="myFunction()">
    Click me to change my HTML content (innerHTML).
</p>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph
changed!";
}
</script>
</body>
</html>
```

# Criando elementos HTML

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to create a P element with some text.</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
    var e = document.createElement("P");
    var t = document.createTextNode("This is a paragraph.");
    e.appendChild(t);
    document.body.appendChild(e);
}
</script>
</body>
</html>
```

# Criando elementos HTML

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to create a P element with some text.</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
    var e = document.createElement("P");
    var t = document.createTextNode("This is a paragraph.");
    e.appendChild(t);
    e.setAttribute("id", "meuParagrafo");
    e.setAttribute("style", "font-size: 20px; text-align:center;");
    document.body.appendChild(e);
}
</script>
</body>
</html>
```

# Evento Listener

```
<!DOCTYPE html>
<html>
<body>
<div id="minhaDiv">Clique na DIV</div>
<script>
    var d = document.getElementById("minhaDiv");
    d.addEventListener('click', function() {
alert('Olá Mundo')  });
</script>
</body>
</html>
```

# Recuperando elementos pelo ID

```
<!DOCTYPE html>
<html>
<body>
  <table id="minhaTabela">
    <tr>
      <td id="l1c1">Linha 1 Coluna 1</td>
      <td id="l1c2">Linha 2 Coluna 2</td>
    </tr>
    <tr>
      <td id="l2c1">Linha 2 Coluna 1</td>
      <td id="l2c1">Linha 2 Coluna 2</td>
    </tr>
  </table>
  <button onClick="alterarTexto()">Alterar Texto</button>
  <script>
    function alterarTexto(){
      celulas = document.querySelectorAll('[id^=l]');
      for (var i = 0; i < celulas.length; i++) {
        celulas[i].innerHTML = "Novo Texto";
      }
    }
  </script>
</body>
</html>
```



# Atividade

1. Implemente um script com um array cujos valores sejam adicionados dinamicamente pelo usuário utilizando-se formulário. A cada nova inserção exiba todos os dados cadastrados em ordem alfabética em uma lista ordenada do html.

# Objetos

- Considerando o objeto carro, podemos dizer que todos os carros possuem propriedades semelhantes cujos valores podem diferenciar-se. Todo carro executa ações semelhantes mas em momentos diferentes.



## Object

## Properties

`car.name = Fiat`

`car.model = 500`

`car.weight = 850kg`

`car.color = white`

## Methods

`car.start()`

`car.drive()`

`car.brake()`

`car.stop()`

# Objetos

- Propriedades do objeto:

```
var pessoa = {  
    nome: "Fulano",  
    sobrenome: "de Tal",  
    idade: 28  
};
```

```
document.write(pessoa.nome);
```

# Objetos

- Métodos do objeto:

```
var pessoa = {  
    nome: "Fulano",  
    sobrenome: "de Tal",  
    idade: 28,  
    nomeCompleto: function() {  
        return this.nome + " " + this.sobrenome;  
    }  
};
```

```
document.write(pessoa.nomeCompleto());
```

# Objetos

- Classes: definição modelo das propriedades e métodos de um objeto

```
function Pessoa () {  
    var nome;  
    this.setNome = function(vNome) {  
        this.nome = vNome;  
    }  
    this.getNome = function() {  
        return this.nome;  
    }  
}
```

# Objetos

```
var pessoa = new Pessoa();  
pessoa.setNome("Fulano");  
document.write(pessoa.getNome());
```

# Atividade

2. Implemente um formulario HTML para input das características de um carro:

marca, modelo, ano, cor, kilometragem, valor\_fipe

Implemente um script que armazene as características preenchidas no formulário em um objeto e que contenha, além das propriedades acima, dois métodos:

anosUtilizacao() – deve retornar quantos anos de utilização o veículo possui;

valorMercado() – deve retornar o valor de mercado considerando a seguinte regra:

carros que rodam até 30.000 km/ano – 110% do valor\_fipe

carros que rodam entre 30.000 e 50.000 km/ano – 100% do valor\_fipe

carros que rodam mais que 50.000 km/ano – 90% do valor\_fipe

Preenchidos os campos os métodos criados devem ser utilizados para exibir quantos anos o veículo tem de utilização juntamente com seu valor de mercado.

# Referências

- [http://www.w3schools.com/js/js\\_arrays.asp](http://www.w3schools.com/js/js_arrays.asp)
- [http://www.w3schools.com/js/js\\_objects.asp](http://www.w3schools.com/js/js_objects.asp)
- <http://tableless.com.br/introducao-a-programacao-orientada-a-objetos-em-javascript/>