JS4GIRLS

Florianópolis - SC

Introdução ao HTML

Visão Geral - Como funciona um Website

Um website (normalmente) contém 3 linguagens:
HTML, CSS e JS

Cada linguagem tem um objetivo, eles são:

HTML - Define os elementos da página
CSS - Aplica estilos visuais à página
JS - Responsável pela a interação

HTML

HTML é uma linguagem de marcação (não de programação), e uma linguagem de marcação é um conjunto de tags de marcação, simples assim

TAGS

Estruturas da linguagem de marcação com instruções para os navegadores/dispositivos possam renderizar a página.

Em pares: ...

TAGS

```
<!DOCTYPE   html>
<html lang=”pt-br”>
        <head>
              <meta charset=utf--8" />
              <title>Minha primeira pagina em HTML</title>
         </head>
         <body>
               <h1>Meu primeiro título</h1>
              <p>Meu primeiro parágrafo</p>
         </body>
</html>
```

Introdução ao CSS

Aqui enfeita as tags

Seletores Visão geral

Seletores selecionam um (ou vários) elementos, para que você possa aplicar estilos nele(s).

p { color: red; }	⇨	Seleciona todos os elementos <p>
p a { color: blue; }	⇨	Seleciona todos os elementos <a> dentro de <p>

Seletores CSS básicos

Simples ou compostos?

Existem duas categorias básicas de seletores: os simples e os combinados.
Um seletor simples consiste em um tipo qualquer de seletor ou o seletor universal seguido por nenhum ou algum seletor de atributo, seletor tipo ID, seletor de classe ou pseudo-classe.

```
p { color: red; }
```

Um seletor combinado (algumas vezes chamado de seletor contextual ou composto) consiste de dois ou mais seletores simples separados por um elemento de combinação. A seguir um exemplo de seletor combinado.

```
div p { font-weight: bold; }
```

Seletores Universal

O seletor universal é representado por um asterisco, “*”, e casa com todos os elementos em seu escopo.

* { margin:0; padding:0; } // todos os elementos -> Global White Space Reset

#menu-lateral * { padding-left: 20px } //todos elementos dentro do #menu-lateral

Seletores Tipo

Seleciona um tipo de elemento. Seu valor é exatamente o texto da TAG HTML.

```
p { color: red; }
```

```
a { color: blue; }
```

```
img { max-width: 100% }
```

Seletor – classe

Classe é representada por um ponto [.] antes do nome, que é o que está no atributo class do html

```
.rosa { color: pink; }
<p> Meu parágrafo normal </p>
<p class=”rosa”> Meu parágrafo rosa </p>
```

O atributo class no HTML pode conter uma lista de vários nomes separados por espaço em branco.

```
<p class=”rosa grande”>Meu texto rosa grande</p>
<p class=”rosa pequeno”> Meu texto rosa pequeno </p>
.rosa { color: pink; }
.rosa.grande { font-size: 20px; }
rosa.pequeno { font-size: 10px; }
```

Seletor – ID

ID’s tem a função semelhante das classes, porém:

1. São exclusivos. Não deve existir mais de 1 elemento com o mesmo ID no HTML.
2. São representados por # ao invés de .

```
p { font-size: 10px; }
p.italico { font-style: italic; }
#introducao { font-size: 20px; }
```

```
<p class=”italico”> Este texto será em itálico com 10px</p>
<p id=”introducao”> Este terá 20px </p>
<p id=”introducao” class=”italico”> Este terá 20px e será em itálico </p>
<p> Este será apenas um texto com 10px </p>
```

Elementos de combinação

Elementos de combinação de seletores são usados para separar dois ou mais seletores simples que compõem um seletor combinado.

Os elementos de combinação disponíveis são:

- espaço em branco
- >
- +

Seletores descendentes

- É uma combinação de dois ou mais seletores simples separados por um espaço em branco
- Refere-se à qualquer elemento descendente. Em qualquer nível.

```
p { color:black; }
div p { color:green; }
```

```
<p> Este parágrafo será preto </p>
<div>
   <p> Um parágrafo terá a cor verde, pois está dentro do div.</p>
   <article>
      <p>Este também será verde, pois ainda está dentro do div</p>
   </article>
</div>
```

```
#nav li { display:inline; }
#nav a { font-weight:bold; }
```

```
<ul id="nav">
  <li><a href="#">Link 1</a></li>
</ul>
<ul>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
</ul>
```

Seletores Filho

Um seletor filho tem como alvo um ***filho imediato*** de um elemento.

```
div > p { font-weight: bold }
```

```
<div>
    <p>Este terá negrito</p>
    <div>
        <p>Este não terá negrito</p>
    </div>
</div>
```

Seletores Irmãos adjacentes

Um seletor de irmão adjacente tem como alvo o elemento diretamente ao lado do primeiro.

```
p + strong { color: red }
```

```
<p>
    <strong> Este texto não será afetado </strong> mas ficará em negrito devido ao strong.
</p>
<strong>Este ficará vermelho</strong>  pois é irmão do p.
<strong> Já este não, pois há um elemento entre o p e este. </strong>
```

Agrupando Seletores

Você pode agrupar os seletores em uma lista e separando-os por uma vírgula, ao invés de escrever repetidamente a mesma regra para cada um dos seletores.

```
p { color: red }
span { color: red }
```

```
p,
span { color: red }
```

```
div#news h3,
ul { margin:0 2em; }
```

```
div#news h3,
div#news ul { margin:0 2em; }
```

Qual a diferença?

```
<div id="news">
  <h3>News</h3>
  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
  </ul>
</div>
```

Seletores de Atributo

Seletores de atributo atingem elementos ***com base no valor de atributo*** declarado no seletor.

- [att] Qualquer elemento com o atributo att independente do seu valor.

- [att=val] - Qualquer elemento com o atributo att cujo valor seja “val”.

- [att~=val] - Qualquer elemento com atributo **att** com valores separados entre espaço, dentre eles **val** *(http://codepen.io/romuloctba/pen/GppwJL)*

- [att|=val] - Qualquer elemento com atributo att de valor igual a um valor qualquer separado por um hífen de um valor começando com “val”. *O principal uso deste seletor é o de casar elementos com um valor de idioma especificado no atributo lang (xml:lang em XHTML), por exemplo;“en”, “en-us”, “en-gb”, etc.*

Introdução ao JavaScript

;)

História do JS

JavaScript foi originalmente desenvolvido por ***Brendan Eich da Netscape*** sob o nome de Mocha. Posteriormente teve seu nome mudado para LiveScript e por fim JavaScript. LiveScript foi o nome oficial da linguagem quando foi lançada pela primeira vez na versão beta do navegador Netscape 2.0 em setembro de 1995, mas teve seu nome mudado em um anúncio conjunto com a Sun Microsystems em dezembro de 1995 quando foi implementado no navegador Netscape versão 2.0B3.

A mudança de nome de LiveScript para JavaScript coincidiu com a época em que a Netscape adicionou suporte à tecnologia Java em seu navegador (Applets).

A escolha final do nome causou confusão dando a impressão de que a linguagem foi baseada em java, sendo que tal escolha foi caracterizada por muitos como uma estratégia de marketing da Netscape para aproveitar a popularidade do recém-lançado Java. JavaScript rapidamente adquiriu ampla aceitação como linguagem de script client-side de páginas web.

Como consequência, a Microsoft desenvolveu um dialeto compatível com o próprio JavaScript, mas que levou o nome de JScript para evitar problemas de trademark. JScript foi incluído no Internet Explorer 3.0, liberado em Agosto de 1996. Em novembro de 1996 a Netscape anunciou que tinha submetido o JavaScript para Ecma internacional como candidato a padrão industrial e o trabalho subsequente resultou na versão padronizada chamada ECMAScript.

O JavaScript tem se transformado na linguagem de programação mais popular da web.

Inicialmente muitos profissionais denegriram a linguagem, pois a mesma tinha como alvo principal o público leigo. Com o advento do Ajax, o JavaScript teve sua popularidade de volta e recebeu mais atenção profissional. O resultado foi a proliferação de frameworks e bibliotecas, práticas de programação melhoradas e o aumento no uso do JavaScript fora do ambiente de navegadores bem como o uso de plataformas de JavaScript server-side.

Onde usar?

Inicialmente o Javascript era utilizado apenas nos navegadores.

Hoje em dia, com a evolução das engines de Javascript como SpiderMonkey e V8, eles levaram o Javascript também para o lado do servidor com o Node.js e bancos NoSQL que utilizam Javascript como CouchDb e MongoDb.

O que oferece?

O JavaScript além de ser a linguagem mais usada no Universo nos oferece algumas coisas interessantes que sem elas a Internet como existe hoje não seria possível:

- Dinamismo
- Validação de Formulários
- Interatividade
- Controle de Comportamento
- Personalização da página

Atualmente o JavaScript é o ***motor*** da Internet, principalmente com o advento do AJAX que fez nossas interfaces ficarem mais ricas e interativas.

O Facebook só existe do jeito como é graças a ele.

Principais Características

- Tipagem dinâmica
- Funcional
- Orientada a Objetos
- Baseada em protótipos
- Detalhes

Javascript não é Java

Java está para o Javascript assim como Bola está para Bolacha.

Principais diferenças

Java é uma linguagem compilada, ao passo que JavaScript é uma linguagem interpretada;

Java é fortemente tipado, enquanto que o JavaScript é fracamente tipado.

Isso só para citar as maiores diferenças, sem contar a sintaxe.

Exemplo Hello JSGirls

Usando o console do navegador (f12), execute:

```
alert("Hello JS4Girls!");
```

Console do Navegador

1. `prompt("Qual é sua idade?");`

2.
```
var idade = prompt("Qual é sua idade?");
alert("Minha idade é " + idade + " anos." );
```

[Exercício] Escreva um código em que você responda qal seu nome e depois escreva com `alert` a mensagem: "Meu nome é " + nome

Introdução à Estrutura de Dados

null

O valor null é um literal em JavaScript que representa um valor nulo ou "vazio" (p/ex: que aponta para um objeto que existe, mas com valor inexistente).

undefined

Representa um valor indefinido.

Objetos inexistentes são undefined

String

String é o tipo utilizado para armazenar textos.

- Uma das operações mais usadas nas strings é checar seu tamanho
- Para concatenar (juntar) usamos os operadores + e +=

```
var palavra = "JS4Girls";
typeof palavra; // "string"
```

```
palavra[2]; //J
palavra.length //8
palavra[0] > palavra[1]
```

```
var palavra = new String("JS4Girls");
typeof palavra; // "object"
```

Number

Armazena números.

```
var a = 420;
var b = 800;
if (a < b) // true pois 420 é menor que 800
  console.log(a + " é menor que " + b);
else if (a > b)
  console.log(a + " é maior que " + b);
else
  console.log(a + " e " + b + " são iguais.");
```

[Exercício] Escreva um código que receberá o ano de nascimento via `prompt` e teste se é o usuário é maior de idade, caso sim mostre a mensagem: "Pode entrar". Caso não, mostre: "Entrada NEGADA!"

Dica:

var idade = prompt(“qual sua idade?”);

Boolean

0 e 1. Verdeiro ou falso. Existe ou não. Ying & Yang.

- ATENÇÃO: Não confunda os valores primitivos Boolean true e false com os valores true and false do objeto Boolean.

Primitivos:	Objeto Boolean:
var a = false;	var b = new Boolean(false)

Qualquer objeto cujo o valor não é undefined ou null, incluindo um objeto Boolean que o valor seja false, é avaliado para true quando passa por uma declaração condicional.

```
if(a){ //nao vai rodar };

if (b) { //este vai rodar };
```

Array

Matrizes

```
var frutas = ['uva', 'maçã', 'tomate'];
```

```
console.log(frutas[0]); // uva
console.log(frutas[1]); // maçã
console.log(frutas[2]); // tomate

console.log(frutas.length); // 23
```

Iterar = Passear pelo Array

```
frutas.forEach( function(item) { console.log(item); } );
```

[Exercício] Escreva um código onde você inicie um *array* com o nome de 5 dos seus amigos e depois faça ele mostrar a mensagem: "Eu gosto muito da(o) " + nome.

Introdução à Lógica

O que é lógica de programação?

Técnica de desenvolver sequências de ações para atingir um objetivo.

Essas sequências são adaptadas para linguagem de computador pelo programador a fim de produzir um sistema.

Essa sequência lógica é denominada algoritmo.

Sequência Lógica

Sequência Lógica

1. Bata as claras em neve
2. reserve
3. Bata bem as gemas com a margarina e o açúcar
4. Acrescente o leite e farinha aos poucos sem parar de bater
5. Por último agregue as claras em neve e o fermento
6. Coloque em forma grande de furo central untada e enfarinhada
7. Asse em forno médio, preaquecido, por aproximadamente 40 minutos
8. Espete um palito. Se sair limpo estará assado.

Assim como uma receita de bolo, em uma sequência lógica você deve dizer passo-a-passo o que o computador deve fazer

Algorítmo

Uma seqüência finita de passos que levam a execução de uma tarefa.

Estas tarefas não podem ser redundantes nem subjetivas na sua definição, devem ser claras e precisas.

Algoritmo para Chupar uma bala:

1. Pegar a bala
2. Retirar o papel
3. Chupar a bala
4. Jogar o papel no lixo

```
escreva("Qual sua idade?")
leia(idade)
se idade > 18 então
    escreva("Maior de idade")
 senão
    escreva("Menor de idade")
 fimse
fim
```

[Exercício] Tendo esse conhecimento agora você deverá criar o seu algoritmo para ir dormir, com no mínimo 4 e no máximo 10 passos.

Instruções

Em informática uma instrução é a informação que indica a um computador uma ação elementar a executar, convém ressaltar que uma ordem/instrução isolada não permite realizar o processo completo, para isso é necessário um conjunto de instruções colocadas em ordem seqüencial lógica.

A instrução mais simples que temos no JavaScript é uma atribuição de valor.

```
var evento = "JS4Girls"
```

Por que nomeamos uma variável?

Para que possamos utilizar o seu valor em outras partes do nosso programa

Boolean

Boolean é um tipo lógico de variável, dado em homenagem da lógica booleana, George Boole.

Só tem 2 valores: “Verdadeiro”, ou “Falso”.

Preciso ir para a Avenida Brasil e pergunto para o GPS:

- Onde estou?

“Avenida Uruguai.”

- Essa é a av. Brasil?

```
local = leiaGPS()
se local = "Avenida Brasil"
    retorne verdadeiro
senão
    retorne falso
```

(vou à praia) = [NOT (está chovendo)]

(está chovendo) = (FALSO)

[NOT (está chovendo)] = [NOT (FALSO)]

NOT (FALSO) = VERDADEIRO

(vou à praia) = VERDADEIRO

AND / E

(vou ao cinema) = [(estou de folga) AND (tenho dinheiro)]

[FALSO] AND [FALSO] = FALSO

[FALSO] AND [VERDADEIRO] = FALSO

[VERDADEIRO] AND [FALSO] = FALSO

[VERDADEIRO] AND [VERDADEIRO] = [VERDADEIRO]

OR / OU

(saio de casa) = (tenho dinheiro) OR (amigos têm dinheiro);

(vou para praia) = (tenho carro) OR (tenho carona) or [(tenho dinheiro) AND (tem onibus)]

NOT / NEGAÇÃO

(tomar banho) = [NOT (frio) AND (ter água)]

IF / SE

Testa SE algo é verdadeiro.

```
if( proposições ) {
  // meu código
}
```

```
var idade = 30;

if(idade > 18) {
    console.log('MAIOR DE IDADE');
}
```

ELSE / SENÃO

Se o if não for verdadeiro.

```
var idade = 30;

if(idade >= 18) {
    console.log('MAIOR DE IDADE');
}
else {
    console.log('MENOR DE IDADE');
}
```

[Exercício] Escreva um código que irá receber o ano que você nasceu em uma variável chama idade e irá testar se é MENOR que 1996, caso sim exiba a mensagem: "OK vc é de maior". Caso não, exiba: "Proibida entrada!"

Parabéns, isso é um sistema de validação de entrada.

ELSE IF / Ou então

“Se o if não for verdadeiro, tenta isso aqui pra ver se é”

```
var tempo = prompt("Que horas são?")
if (tempo < 13) {
    saudacao = "Bom dia";
} else if (tempo < 19) {
    saudacao = "Boa tarde";
} else {
    saudacao = "Boa noite";
}
```

switch

Serve para testarmos várias condições e executar o código necessário de acordo.

```
var estadoCivil = prompt("Qual seu estado civil?");

switch(estadoCivil) {
    case 'solteira':
        console.log("Bora pra festa?");
    break;
    case 'casada':
        console.log("Parabéns pelo casamento!");
    break;
    case 'divorciada':
        console.log("Deve ser um alívio!");
    break;
    case 'viúva':
        console.log("Meus pesames!");
    break;
    default: console.log("Complicado");
}
```

Utilizamos instrução 'default 'para executar um código quando nenhum dos outros cases foi verdadeiro. Dessa forma deixando nosso código mais simples e claro do que se colocássemos um monte de else if

while

Enquanto...

Para facilitar nossa vida temos os laços de repetição. Podemos repetir um bloco de comandos quantas vezes necessário.

```
while( proposição ) {
    //seu código
}
```

```
var numero = 1;
while(numero <= 10) {
    console.log(numero);
    numero++;
}
```

[Exercicio] Escreva um código onde inicie um número com o valor 0 e vá até 20, mostrando apenas os valores pares.

do while

Muito parecido com o while, porém sempre irá executar primeiro, e conferir a condição depois.

```
var numero = 1;

do {
    console.log(numero);
    numero++;
} while(numero <= 10);
```

for

para ... faça... e depois....

```
for(inicialização; condição; expressão final) {
    // seu código
}
```

```
for(var numero = 1; numero <= 10; numero++) {
    console.log(numero);
}
```

```
var numero = 1;
for(; numero <= 10; numero++) {
    console.log(numero);
}
```

```
for(var numero = 1; ; numero++) {
    if(numero > 10) break;
    console.log(numero);
}
```

```
var numero = 1
for(; ; ) {
    if(numero > 10) break;
    console.log(numero);
    numero++;
}
```

Funções

um conjunto de instruções utilizadas para executar uma determinada tarefa. Seu principal objetivo é evitar que um trecho de código seja repetido sempre que for preciso efetuar uma operação.

Sintaxe

```
function nome ( parametro ) {
  //código a ser executado
}
```

Parâmetros e Argumentos

```
function boasVindas (nome) {
    return "Eu estou feliz por você estar aqui, " + nome;
}

function multiplicar (x, y) {
    console.log(“resultado: “, x * y);
}

boasVindas(“Kátia”);

var pessoa = “Kátia”;
boasVindas(pessoa);
```

Retorno

```
function retorna(algo){
    return algo;
}

function multiplicar (x, y) {
    return x * y;
}
```

Invocando uma função

```
nomeDaFuncao(arg1, arg2);

var resultado = nomeDaFuncao(arg1, arg2);

console.log(resultado) //retorno da funcao
```

[Exercício] Crie uma função para calcular o IMC e invoque-a passando dois argumentos.
*Cálculo do IMC: peso / (altura * altura)*

Objetos

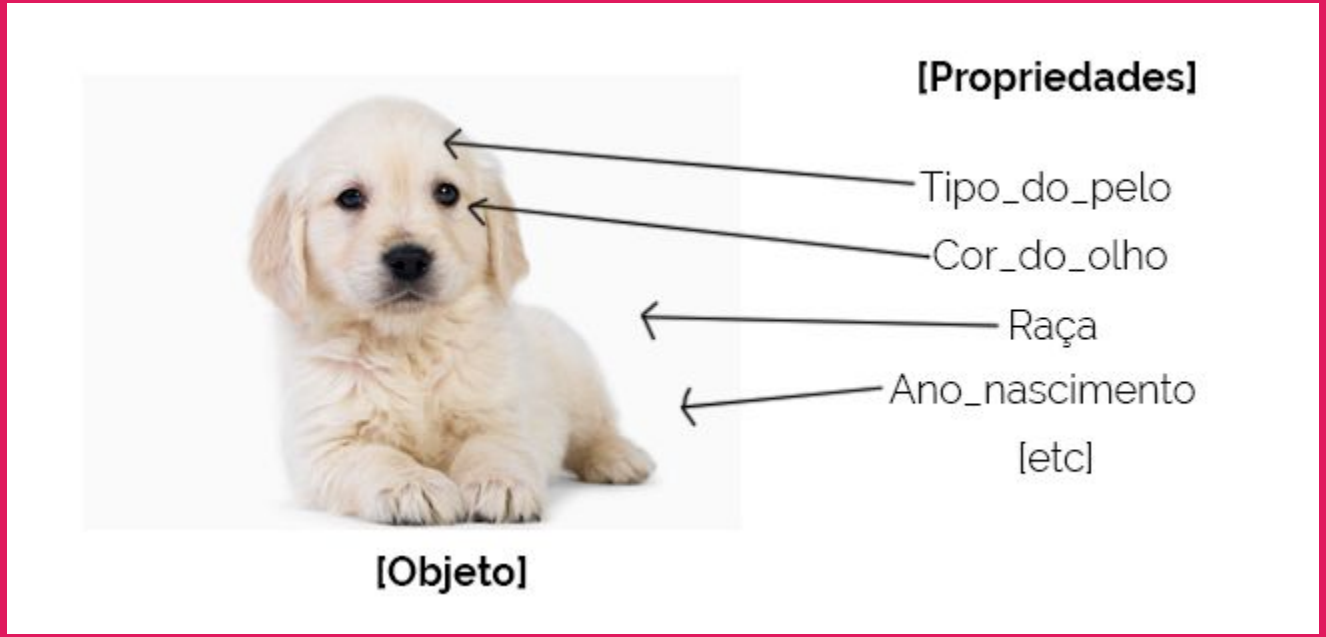
O que é um objeto?

O JavaScript é orientado a objetos.

- O que é um objeto?
Um objeto é uma coleção de propriedades.

- O que é uma propriedade?
Propriedade é uma associação entre um nome e um valo

Visão geral de objetos



Objetos e Propriedades

Acessa-se as propriedades de um objeto com uma simples notação de ponto:

nome_do_objeto.nome_da_propriedade

```
var meuAviao = new Object();
meuAviao.fabricante = "Airbus";
meuAviao.modelo = "A380";
meuAviao.ano = 2012;
```

```
meuAviao["fabricante"] = "Airbus";
meuAviao["modelo"] = "A380";
meuAviao["ano"] = 2012;
```

[Exercício] Crie um objeto e chame ele de "MeuVestido". Em seguida crie as seguintes propriedades: "Cor","Tamanho","Marca" e "Tipo" (Curto ou longo).

Criando novos objetos

- Usando inicializadores de objeto
- Usando uma função construtora
- Usando o método Object.create Link

typeof

Retorna o tipo do item.
Mas pode enganar as desavisadas.

instanceof

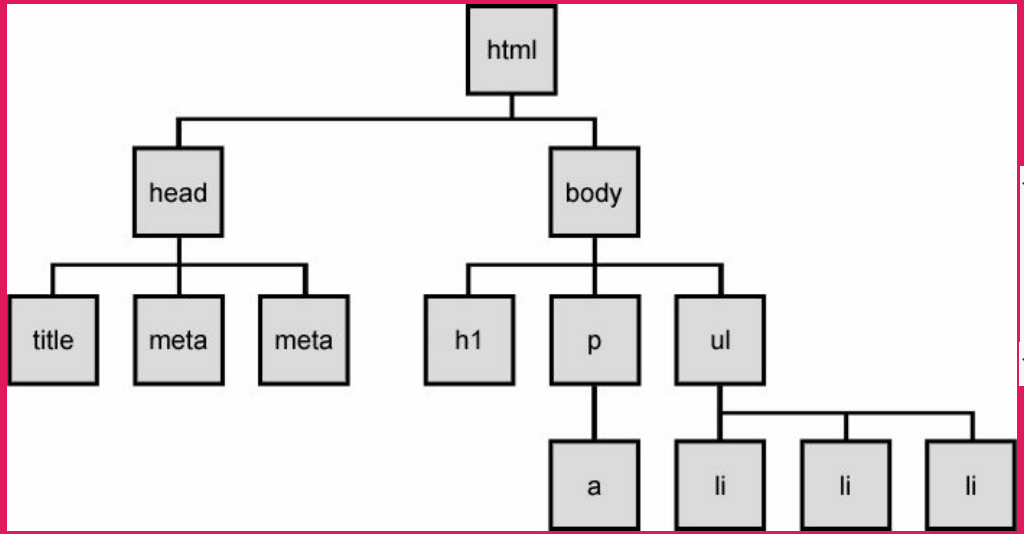
Retorna true or false. Não te engana.
É sincero.

```
var teste3 = [2, 3, 5, 1, 2, 3];
console.log(teste3 instanceof Array);
//true
```

JavaScript e o Navegador

DOM: sua página no mundo JavaScript

DOM (Document Object Model)




```
<p>
  Sou o pai, e
  <a href="#">eu o filho</a> com um
  <span>irmão</span>
</p>
```

Localizando elementos na nossa página

- getElementById
- getElementsByTagName
- getElementsByClassName
- getElementsByName
- querySelector
- querySelectorAll

Manipulando elementos da nossa página

- innerHTML
- innerText
- value

Eventos

Avisam quando algo acontece.

Exemplos de eventos:

Um clique no mouse (onclick)

O carregamento de uma página ou imagem web (onLoad)

Quando o mouse passa sobre um anúncio em uma página web (onmouseover)

Selecionar um campo de entrada em um formulário HTML (onfocus)

Submeter um formulário HTML (onsubmit)

Pressionar uma tecla (onkeydown)

Navegando pelo DOM

- parentNode
- nextSibling
- previousSibling
- firstChild
- lastChild
- childNodes

!!!! Atenção !!!! esses métodos podem retornar espaços em branco entre as tags HTML ao invez da primeira tag HTML filha do elemento selecionado causando resultados não esperados

JavaScript e CSS

```
var elemento = document.getElementById('idDoElemento');
elemento.style.backgroundColor = #222;
```

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Properties_Reference
http://codepen.io/romuloctba/pen/vNNMqZ

Criando elementos dinamicamente

var elemento = document.createElement('elemento');
var ondeVai = document.getElementById('id-do-elemento');
ondeVai.appendChild(elemento);

```
<div id="conteudo">
</div>
```

```
//Selecionando o elemento que irá
receber o novo elemento
var conteudo = document.getElementById
('conteudo');

//Criando um novo elemento
var text = document.createElement
('span');
text.innerText = 'Eu não estava aqui
antes';

//Usando o método append para colocar o
elemento que criamos na div conteudo que
selecionamos
conteudo.appendChild(text);
```

```
<div id="conteudo">
  <span>Eu não estava aqui antes</span>
</div>
```

Mais métodos

Projeto Final

Mais Atributos: https://developer.mozilla.org/pt-BR/docs/HTML/Attributes

Outros projetos feitos para mulheres

https://www.facebook.com/AnitasFloripa/

Grupo de mulheres engajadas no empoderamento feminino na área de tecnologia e empreendedorismo através da troca de experiências e conhecimentos.

https://www.facebook.com/PyLadies-Floripa-763762813730310/

Somos um grupo internacional, com foco em ajudar mais mulheres a tornarem-se participantes ativas e líderes da comunidade de código aberto Python.