

Introdução à Linguagem HTML

Introdução JavaScript



Prof. Fabrício Ribeiro Ferreira



UVV

**Melhor Universidade
Particular do Brasil (MEC)**



**Times
Higher
Education**



Prêmio Sebrae de
**Educação
Empreendedora**



Estrutura do curso

1

A Internet e seu funcionamento

2

Primeiros passos com HTML e CSS

3

HTML e CSS mais profissional

4

Colocando protótipos no ar

5

Aprofundando conhecimentos no HTML e CSS

6

Novas tecnologias e frameworks

7

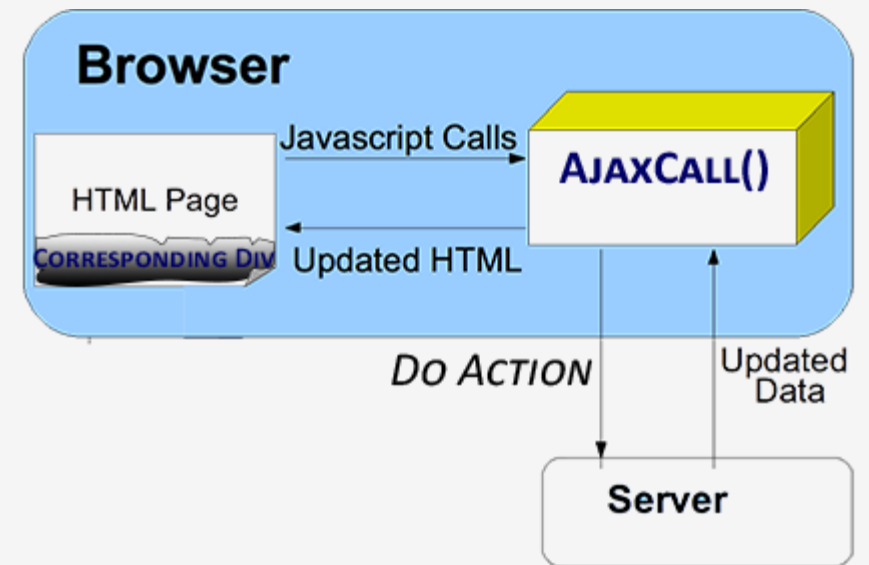
Introdução a JavaScript

História do JavaScript

- A Necessidade: A Netscape percebeu a necessidade de ter uma maneira de adicionar interatividade às páginas web, que até então eram em grande parte estáticas.
- A Criação: Em 1995, a Netscape contratou Brendan Eich com a tarefa de criar uma nova linguagem de programação para o navegador. Ele desenvolveu o protótipo dessa linguagem em apenas 10 dias! Esta linguagem foi inicialmente chamada de Mocha, depois renomeada para LiveScript e, finalmente, nomeada de JavaScript – uma jogada de marketing para aproveitar a popularidade da linguagem Java naquela época. Vale ressaltar que, apesar dos nomes parecidos, Java e JavaScript são linguagens muito diferentes.
- Padronização: Dado o sucesso do JavaScript no Netscape, a Microsoft desenvolveu sua própria versão da linguagem chamada JScript para o Internet Explorer. Para evitar uma "guerra" de versões e garantir consistência, a Netscape submeteu o JavaScript ao Ecma International, uma organização de padrões.

História do JavaScript

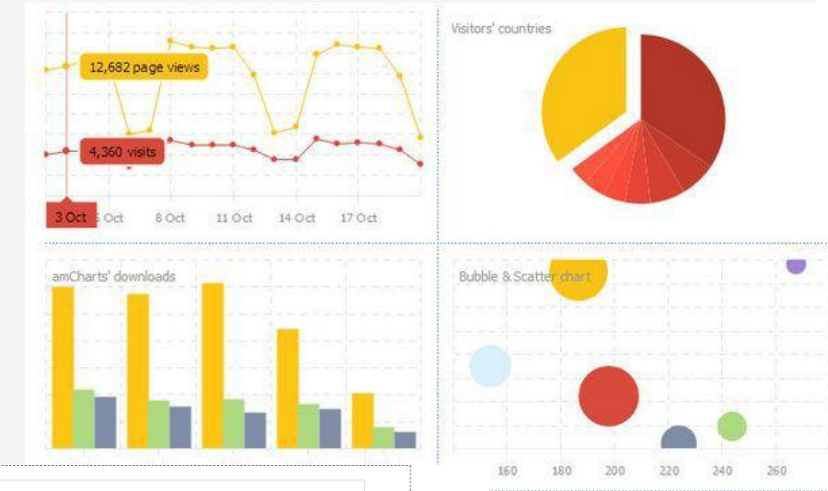
- **ECMA** significa European Computer Manufacturer's Association, essa linguagem é o padrão para linguagens de script, como JavaScript, JScript etc., sendo uma especificação de linguagem de script de marca registrada. O JavaScript é uma linguagem baseada no ECMAScript e é considerado uma das implementações mais populares do ECMAScript.
- O resultado foi a norma ECMA-262, que define a linguagem padrão conhecida como ECMAScript. JavaScript é, portanto, uma implementação do ECMAScript.
- **Anos 2000:** O JavaScript ganhou mais destaque com o surgimento das aplicações de "Página Única" (SPA - Single Page Applications) e o AJAX (Asynchronous JavaScript and XML), que permitia a atualização de páginas web sem a necessidade de recarregá-las por completo.



JAVASCRIPT

JavaScript é uma linguagem de script usada para criar e controlar o conteúdo dinâmico de um site, ou seja, qualquer coisa que se mova, atualize ou mude de outra forma na tela sem exigir que você recarregue manualmente uma página web. Recursos como:

- Gráficos animados.
- Apresentações de slides de Imagens.
- Sugestões de preenchimento automático de texto.
- Formulários interativos.



The form is titled 'titulo de la encuestas' and includes a header 'esto es un encabezado'. It contains a dropdown menu for 'Direccion de camaras' and a dropdown for 'Profesores'. Below these are two sections for adding questions:

- pregunta 1:** Includes a text input for 'opcion 1', a numeric input for 'opcion 2', and a dropdown for 'opcion 3'. There are buttons for '+ opcion' and 'X Pregunta'.
- pregunta 2:** Includes a text input for 'opcion 1' and a numeric input for 'opcion 2'. There are buttons for '+ opcion' and 'X Pregunta'.

J A V A S C R I P T

Veremos em javascript as construções básicas da linguagem:

- Variáveis, operadores lógicos, operadores aritméticos, estruturas condicionais e estruturas de repetição;
- Descrevemos os fundamentos do Document Object Model, modelo de objeto de documento (o DOM);
- Com o DOM, podemos acessar e manipular facilmente tags, IDs, classes, atributos ou elementos, usando comandos ou métodos fornecidos pelo objeto Document

J A V A S C R I P T

- Document Object Model, modelo de objeto de documento (o DOM), que fornecerá uma base sólida para a compreensão de construções JavaScript.
- O DOM é uma forma de representar a página web de uma forma hierárquica e estruturada, para que seja mais fácil para os programadores e usuários deslizarem pelo documento.
- Através do DOM, podemos acessar e manipular facilmente tags, IDs, classes, atributos ou elementos, usando comandos ou métodos fornecidos pelo objeto Document
- Por exemplo, uma página web do tipo formulário é construída com a estrutura do DOM, e essa estrutura permite que os formulários processem dados inseridos pelo usuário quando esse usuário clica em um botão

J A V A S C R I P T

- Requisitos básicos: Para mergulhar no JavaScript, tudo que você precisa é um editor de texto simples e um navegador.
- Para definir um bloco JavaScript em sua página web, basta usar o elemento container script no interior do elemento head

```
<!DOCTYPE html>
<html lang="pt-br">
<head> ...
</head>
<body>
  <script type='text/javascript'>
    document.writeln('Olá Mundo Javascript');
  </script>
</body>

</html>
```

J A V A S C R I P T

- **Node.js:** Em 2009, Ryan Dahl introduziu o Node.js, um ambiente de execução JavaScript no lado do servidor. Isso expandiu as capacidades do JavaScript, permitindo que ele fosse usado não apenas no navegador, mas também no servidor.



J A V A S C R I P T

Saídas

➤ **Conceitos Básicos:** Uma das coisas mais importantes a se fazer ao aprender uma nova linguagem é dominar a entrada e a saída básicas. No JavaScript temos 3 formas básicas de saídas para se comunicar com o usuário:

❖ ***Document.writeln(string)*:** Isso pode ser usado enquanto a página está sendo construída. Conforme a página está carregando, o JavaScript encontrará esse script e mostrará na página

```
<body>
  <script type="text/JavaScript">
    document.writeln("Olá, meu código JavaScript! Usando o Document Writeln ")
  </script>
  <h1>Olá Mundo, JavaScript</h1>
  <p></p>
  <script>
    document.writeln("Olá, meu código JavaScript! Usando o Document Writeln pela
segunda vez ")
  </script>
</body>
```


J A V A S C R I P T

Saídas

- ❖ ***window.Alert(string)***: O segundo método é usar uma caixa de alerta do navegador. Embora sejam incrivelmente úteis para depurar (e aprender a linguagem), não é uma boa maneira de se comunicar com o usuário. Caixas de alerta impedirão a execução de seus scripts até que o usuário clique no botão OK.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Meu documento JavaScript</title>

  <script type="text/JavaScript">
    window.alert("Olá, meu código javascript!")
  </script>
</head>
<body>
  <h1>Olá Mundo, JavaScript</h1>
</body>
</html>
```

J A V A S C R I P T

Saídas

❖ ***getElementById***: O último método é o mais poderoso e o mais complexo.

- ✓ Tudo em uma página web reside em um container. Um parágrafo (<P>) é um container.
- ✓ No documento HTML, ao marcar algo como negrito, você cria um pequeno container ao redor do texto que conterá o texto em negrito.
- ✓ Você pode dar a cada container em HTML um identificador exclusivo (um ID), e o JavaScript pode localizar os containers que você rotulou e permitir que você os manipule

```
<body>
  <h1>Olá Mundo, JavaScript</h1>
  <p></p>
  <div id='resposta'> </div>
  <script>
    document.getElementById('resposta').innerHTML= 'Resposta de saída: Olá Mundo'
  </script>
</body>
```

JAVASCRIPT

ENTRADAS

Da mesma forma que você utiliza o **Alert** para uma saída, útil para debugar, você pode usar um comando simples para realizar uma entrada de dados com JavaScript, usando o "**prompt**" ou "**confirm**"

```
<body>
  <p></p>
  <script>
    window.prompt("Olá, meu código JavaScript! Usando o Document Writenl pela segunda
vez ")
  </script>
  <script>
    window.confirm("Está gostando na JS?")
  </script>
</body>
```


JAVASCRIPT

ENTRADAS

Os cliques são poderosos e fáceis e você pode adicionar um evento onClick a praticamente qualquer elemento HTML, mas às vezes você precisa ser capaz de solicitar a entrada do usuário e processá-la, você precisará de um elemento de formulário básico e um botão. -

```
<body>

  <input id="inputUsuario" size="60"> <button onclick="enviaResposta()">Enviar</button>
<br>
  <script type="text/JavaScript">
    function enviaResposta() {
      var IU = document.getElementById('inputUsuario').value;
      window.alert('Foi digitado ' + IU);
    }
  </script>
</body>
```

J A V A S C R I P T

EVENTOS

- Alguns eventos comuns do JavaScript

Evento	Descrição
onAbort	Falha ao carregar imagem.
onBeforeUnload	O usuário está navegando para fora de uma página.
onBlur	Um campo de formulário perdeu o foco (o usuário mudou para outro campo).
onChange	O conteúdo de um campo mudou.
onClick	O usuário clicou neste item.
onDbClick	O usuário clicou duas vezes neste item
onError	Ocorreu um erro ao carregar uma imagem
onFocus	O usuário acabou de se mover para este elemento de formulário.
onKeyDown	Uma tecla foi pressionada.
onKeyPress	Uma tecla foi pressionada OU liberada.

JAVASCRIPT**EVENTOS**

- Alguns eventos comuns do JavaScript

Evento	Descrição
onKeyUp	Uma tecla foi liberada.
onLoad	Este objeto (iframe, image, script) terminou de carregar
onMouseDown	Um botão do mouse foi pressionado.
onMouseMove	O mouse se moveu.
onMouseOut	Um mouse se moveu para fora deste elemento.
onMouseOver	O mouse se moveu sobre este elemento.
onMouseUp	O botão do mouse foi liberado.
onReset	Um botão de redefinição de formulário foi pressionado.
onResize	A janela ou quadro foi redimensionado.
onSelect	Texto foi selecionado.

FLEXBOX**• Exercício : Calculadora de Idade**

- Crie uma página web que tenha um campo de entrada (input) onde o usuário o dia de nascimento.
- Crie uma página web que tenha um campo de entrada (select) onde o usuário o escolha o mês de nascimento.
- Crie uma página web que tenha um campo de entrada (input) onde o usuário o ano de nascimento.
- Adicione um botão "Calcular Idade".
- Quando o botão for clicado, calcule a idade do usuário baseada na data atual e exiba-a o resultado em anos em um label .



Referências

Apostila:

Guanabara, Gustavo. Curso completo e atual de HTML5 e CSS

Sintaxe JavaScript - Variáveis

- JavaScript não é uma linguagem fortemente tipada, o que significa que você raramente tem de se preocupar com o tipo de dados que uma variável está armazenando, apenas com o que a variável está armazenando.
- Em JavaScript, as variáveis podem armazenar qualquer coisa, até mesmo funções.

```
var issoEumaString = 'Isso é uma string';
var tambemString = '25';
var eUmNumero = 25;
var eIgual = (tambemString == eUmNumero); // Isso é verdade, ambos são 25.
var eIgual = (tambemString === eUmNumero); // Falso, um é um numero e o outro string.
var concat = tambemString + eUmNumero; // concat é agora 2525
var adicao = eUmNumero + eUmNumero; // adicao é agora 50
var tambemUmNumero = 3.05; // é igual a 3.05 (normalmente).
var floatErro = 0.06 + 0.01; // é igual a 0.06999999999999999
var umaPotencia = 1.23e+3; // é igual a 1230
var hexadecimal = 0xff; // é igual a 255.
var octal = 0377; // é igual a 255.
var eVerdade = true; // Isso é um booleano, pode ser verdadeiro ou falso, true ou false.
var eFalso = false; // Isso é um booleano, pode ser verdadeiro ou falso, true ou false.
var eVetor = [0, 'um', 2, 3, '4', 5]; // Isso é um vetor.
var quatro = eVetor[4]; // atribui um único elemento do vetor a uma variável. Nesse caso quatro = '4'
var eObjeto = {
  'cor': 'azul', // Esse é um objeto Javascript
  'cão': 'latir',
  'vetor': [0, 1, 2, 3, 4, 5],
  'minhafuncao': function () { alert('faça alguma coisa!'); }
}
var cao = isObjeto.cao; // cao agora armazena a string latir
isObjeto.minhafuncao(); // Cria uma caixa de alerta com o valor "faça alguma coisa!"
var algumaFuncao = function () {
  return "Eu sou uma função!";
}
var tambemFuncao = algumaFuncao; //Sem os parenteses tambemFuncao se torna uma função
var resultado = tambemFuncao(); /* tambemFuncao é executada aqui porque está acompanhada de parênteses
a variavel resultado armazena o retorno da função que nesse caso é "Eu sou uma função!" */
```


Sintaxe JavaScript – Escopo da Variáveis

- Variáveis em JavaScript têm escopo. Ou seja, todas as variáveis são globais, a menos que sejam explicitamente definidas dentro de uma função e mesmo assim as funções-filho têm acesso às variáveis de seus pais.
- Se uma função define uma nova variável sem usar a palavra-chave `var`, essa variável terá escopo global

```
var global = 'Essa é global';
function escopoFuncao() {
    tambemGlobal = 'Essa também é global!';
    var naoGlobal = 'Essa é privada a escopoFuncao!';
    function subFuncao() {
        alert(naoGlobal); // Ainda podemos acessar naoGlobal nessa sub função.
        aindaGlobal = 'Não antecedida da palavra chave var então é global!';
        var ePrivada = 'Essa é privada a subFuncao!';
    }
    alert(aindaGlobal); // Isso é um erro porque subFuncao não foi executada
    subFunction(); // executa a sub função
    alert(aindaGlobal); // A saída será 'Não antecedida da palavra chave var então é global!'
    alert(ePrivada); // Gera um erro porque ePrivada é privada a subFuncao ()
    alert(global); // Saída: 'Essa é global'
}
alert(global); // Saída: 'Essa é global'
alert(tambemGlobal); // Gera um erro porque escopoFuncao ainda não foi executada.
escopoFuncao();
alert(tambemGlobal); // Saída: 'Essa também é global!';
alert(naoGlobal); // Gera um erro.
```

Sintaxe JavaScript – Operadores Aritméticos

Não há nada de particularmente exótico na maneira como o JavaScript faz aritmética

Operador	Descrição
+	Adição (também concatenação de string).
-	Subtração (também menos unário)
*	Multiplicação
/	Divisão
%	Restante da divisão (ou módulo).
++	Pré ou pós incremento.
--	Pré ou pós decremento.

Sintaxe JavaScript – Operadores de Comparação

Não há nada de particularmente exótico na maneira como o JavaScript faz aritmética

Operador	Descrição
=	Atribuição <code>x = 5;</code> // atribui 5 a x
==	Igualdade, é <code>x == 5</code> ?
===	Identidade, x 5 e um número também?
!=	Diferente, x é diferente de 5?
!==	Não é idêntico, x é diferente do número 5?
!	Não, se falso é verdadeiro.
	OR, é <code>(x == 5) OR (y == 5)</code>
&&	E, é <code>(x == 5) AND (y == 5)</code>
<	Menor que. x é menor que 5?
<=	Menor ou igual. é x menor ou igual a 5?
>	Maior que ou qual. é x maior ou igual a 5?
>=	Maior que ou qual. é x maior ou igual a 5?

Sintaxe JavaScript – Operadores de Condicionais - IF

São operadores que expressão uma condição para um trecho do código;

A instrução if permite que você execute um bloco de código se algum teste for aprovado.

```
var x = 5;
if (x == 5) {
    alert('x é igual a 5!');
}
```

A instrução também pode usada para a estruturar o else para executar o código se o teste falhar

```
var x = 5;
if (x == 5) {
    alert('x é igual a 5!');
} else {
    alert('x não é igual a 5!');
}
```

Sintaxe JavaScript – Condicionais: Switch

Caso precise utilizar um grande número de testes, faz sentido usar uma instrução switch em vez de ifs aninhados. Os switches em JavaScript são bastante poderosos, permitindo avaliações tanto na estrutura switch quanto na estrutura case.

```
var x=5;
switch (x) {
case 1: alert('x é igual a 1!'); break;
case 2: alert('x é igual a 2!'); break;
case 5: alert('x é igual a 5!'); break;
default: alert("x não é 1, 2 ou 5!");
}
```


Sintaxe JavaScript – Loops - For

O loop for segue a sintaxe C básica, consistindo em uma inicialização, uma avaliação e um incremento

```
for (var i = 0; (i < 5); i++) {  
    document.writeln('I é igual a ' + i + '<br>');  
}  
// saídas:  
// I é igual a 0  
// I é igual a 1  
// I é igual a 2  
// I é igual a 3  
// I é igual a 4
```

Sintaxe JavaScript – Variáveis

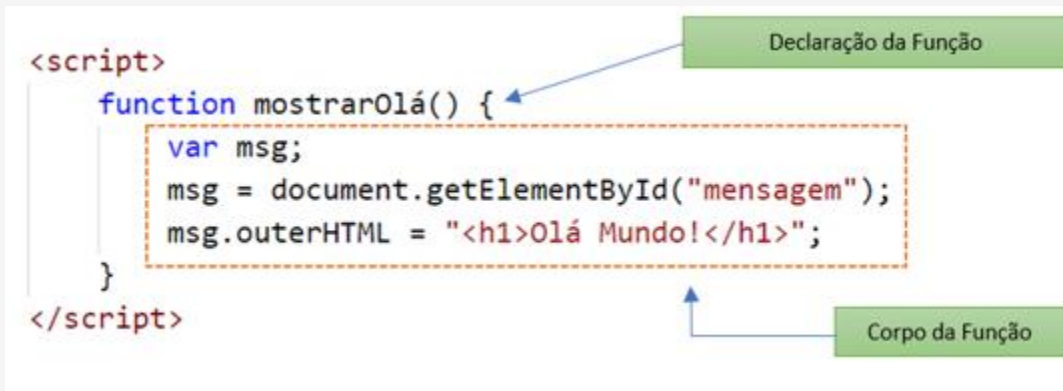
Variáveis são áreas da memória que são reservadas para guardar conteúdos

```
var time;  
var saldoDeGols;
```

```
time = "Vasco";  
saldoDeGols = 100;
```

Sintaxe JavaScript – Declaração de Objetos e atribuição

No exemplo abaixo temos uma atribuição de valores em uma função, buscando a partir de um objeto.



The diagram illustrates the syntax of a JavaScript function declaration within a script block. The code is as follows:

```
<script>
  function mostrarOlá() {
    var msg;
    msg = document.getElementById("mensagem");
    msg.innerHTML = "<h1>Olá Mundo!</h1>";
  }
</script>
```

Annotations in the diagram:

- A green box labeled "Declaração da Função" points to the `function` keyword and the function name `mostrarOlá()`.
- A green box labeled "Corpo da Função" points to the curly braces `{ }` enclosing the function body.
- A dashed orange box highlights the function body, containing the variable declaration `var msg;` and the two assignment statements: `msg = document.getElementById("mensagem");` and `msg.innerHTML = "<h1>Olá Mundo!</h1>";`.