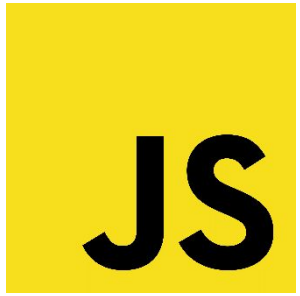
Tecnologia para Front- end

Prof. Leandro Melo

Agenda

- **Historia;**
- **Introdução Javascript;**
- **Exercícios.**

JavaScript



Origem do nome da linguagem

- **O nome "JavaScript" não tem uma origem direta relacionada à sua funcionalidade. A história por trás do nome é um pouco curiosa.**
- **Quando a linguagem foi criada por Brendan Eich enquanto trabalhava na Netscape Communications Corporation no início dos anos 1990, ela era originalmente chamada de "Mocha".**
- **No entanto, logo depois, foi renomeada para "LiveScript" como parte de uma parceria estratégica com a Sun Microsystems.**

Origem do nome da linguagem

- **Foi apenas mais tarde, quando a linguagem foi lançada no navegador Netscape Navigator 2.0 em dezembro de 1995, que a Netscape e a Sun fizeram um acordo de licenciamento com a Sun para incluir o Java no Netscape Navigator.**

- **Como parte desse acordo, a linguagem LiveScript foi renomeada para "JavaScript" como uma jogada de marketing, aproveitando o hype em torno da linguagem de programação Java da Sun na época.**

Origem do nome da linguagem

- **É importante notar que, apesar do nome "JavaScript", a linguagem em si não está relacionada diretamente ao Java.**

- **Ela tem uma sintaxe semelhante, mas é uma linguagem completamente diferente, com funcionalidades e propósitos distintos.**

Mozilla Firefox é a evolução do Netscape?

- **Netscape foi um dos primeiros navegadores populares nos anos 1990.**
- **Em 1998, a Netscape lançou o projeto Mozilla ao liberar seu código-fonte.**
- **O projeto evoluiu e, em 2002, resultou no lançamento do Firefox.**
- **O Firefox não é exatamente o Netscape, mas surgiu a partir do mesmo código e legado.**

O que é Javascript ?

E o que é o Javascript?

- **É uma linguagem de programação script.**
- **Um script (ou scripting ou linguagem de script) é uma linguagem de programação que suporta scripts, ou seja, programas escritos para um sistema com tempo de execução especial para automatizar a execução de algumas tarefas.**
- **O tempo é especial porque muitas vezes depende da ação humana no sistema para disparar a execução do script.**

Javascript

- **Permite que códigos de programação possam ser inseridos em páginas HTML.**
- **Pode ser executada por todos os navegadores (*browsers*) modernos.**
- **Linguagem de programação simples e leve.**

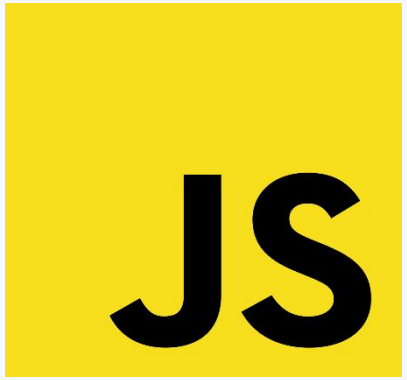
Introdução Javascript

=

JS Javascript

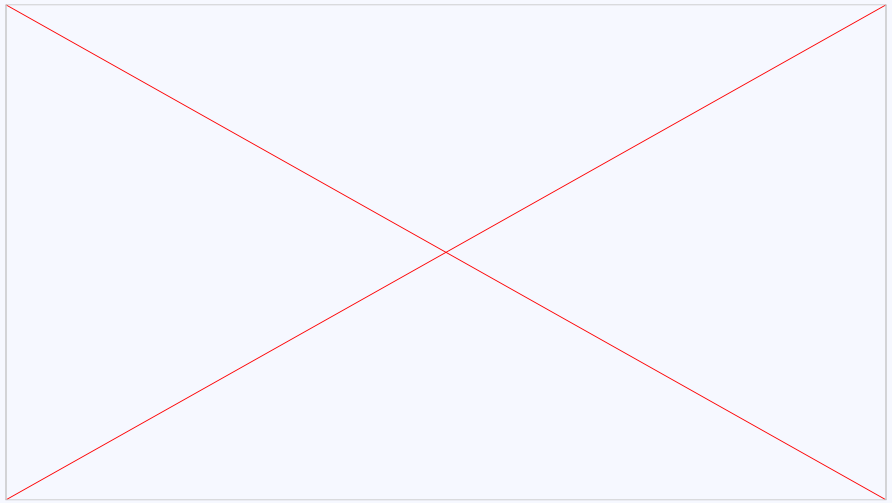
Introdução Javascript

≠



Javascript

Ambiente - Configuração



Navegadores

Visual Studio Code

SublimeText

Noje.js

Como inserir Javascript no HTML?

HTML

HTML + CSS

HTML + CSS + JAVA SCRIPT

Como inserir Javascript no HTML?

Existem 3 formas

1. **Relacionando um arquivo externo a partir do atributo src da tag <script></script>.**
2. **Delimitando o código a partir dos conjuntos de tags HTML <script></script> em um documento HTML. Normalmente logo após a tag <body>.**
3. **Implementando o script em um atributo de tag que representa um evento na página.**

Vamos exemplificar as 3 formas...

1. A partir da tag **src** – arquivo externo

Exemplo de importação do script meu_script.js

```
<html>
    <head>
    </head>
    <body>

        <script type="text/javascript" src="meu_script.js">
    </body>
</html>
```

Um detalhe...

Para garantir que a chamada ao arquivo Javascript externo seja executada apenas após o carregamento completo da página, você pode usar o atributo defer na tag <script>.

```
<!DOCTYPE html>
<html>
<head>
    <title>Sua Página</title>
</head>
<body>
    <!-- Outros conteúdos da página -->

    <!-- Seu script JavaScript externo -->
    <script src="seu-arquivo.js" defer></script>
</body>
</html>
```

2. A partir da tag <script> - interno

Exemplo de utilização do Javascript no HTML:

```
<html>
    <body>
        <script>
            // Código-fonte Javascript
        </script>
    </body>
</html>
```

3. No atributo das tags - local

Exemplo de chamada a função confirmação() no atributo onload de body.

```
<html>
  <body onload="confirmacao()">
    <script>
      function confirmacao() {
       confirm("Página carregada com sucesso!");
    }
    </script>
    </body>
</html>
```

Sintaxe Básica

Sintaxe Básica

- **Características Gerais**

  É uma linguagem de programação interpretada; Possui tipagem fraca e dinâmica.

- **Sintaxe Básica**

Em javascript, é facultativo o uso do ; (ponto e vírgula) para delimitar o fim de um comando.

Valores são divididos em LITERAIS e VARIÁVEIS.

Exemplo de valores LITERAIS: “joão e maria”, 10.5, 19, true;

Exemplo de valores variáveis: var nome = “joão e maria”;

Sintaxe Básica

Uso do ponto e vírgula

Em javascript, é facultativo o uso do ; (ponto e vírgula) para delimitar o fim de um comando.

```
<html>
  <head></head>
  <body>
    <script type = "text/javascript">
      document.write("Fim de linha com ponto e vírgula");
      document.write("Fim de linha sem ponto e vírgula")
    </script>
  </body>
</html>
```

Pulando linha em Javascript

Como você deve ter percebido no exemplo anterior, as duas frases ficaram na mesma linha.

Este comportamento acontece porque toda saída de comando enviada pelo Javascript é tratada como código HTML. Por isso, deve-se fazer o uso da tag “
” ao final da frase.

```
<script type = “text/javascript”>
  document.write(“Fim de linha com ponto e vírgula <br/>”);
  document.write(“Fim de linha sem ponto e vírgula <br/>”)
</script>
```

Sintaxe Básica

Comentários de linha e multilinhas

Javascript oferece dois tipos de comentários “//” e “/**/”

A semântica é a mesma do que acontece em linguagens como Java e C++.

```
<html>
  <head></head>
  <body>
    <script type = “text/javascript”>
      /*
        Isso é um comentário
      */
      document.write(“<h1>Isso é um cabeçalho</h1>”)
    </script>
  </body>
</html>
```

Usando Variáveis

- Variáveis em Javascript possui a mesma semântica que em outras linguagens, isto é, armazenar informações na memória;
- O Javascript, é case sensitive;
- Podem ser iniciadas por [aA-zZ], _ ou $;
- A partir do segundo caractere pode ter os seguintes tipos [aA-zZ], [0-9], _ ou $.

Comportamento das Variáveis

Diferente de outras linguagens de Programação, com C/C++ e Java, no Javascript não é necessário indicar o tipo da variável na sua declaração.

O Javascript possui **inferência** de **tipo**, o que possibilita a **tipagem dinâmica**.

Exemplo do uso de variáveis

```
<html>
  <head></head>
  <body>
    <script type = "text/javascript">
      var firstName;
      firstName = "João";
      document.write(firstName + "<br/>")
      firstName = "Pedro"
      document.write(firstName + "<br/>")
    </script>
  </body>
</html>
```

Consideração de variáveis declaradas com “var”

- É possível definir a mesma variável várias vezes (Mas sempre irá sobrescrever o valor da primeira definição).
- Declara a variável na memória antes da expressão que à utilizou (Mas não atribui o valor da variável (valor=undefined)). **(Script)**
- Pode ser acessada fora do bloco (escopo) de estruturas condicionais ou repetição.
- Funciona de uma forma local dentro de funções.

Sintaxe Básica

Uso de variáveis não declaradas

É possível atribuir valores à variáveis ainda não declaradas.

```
<html>
  <head></head>
  <body>
    <script type = “text/javascript”>
      firstName = “João”;
      document.write(firstName + “<br/>”)
    </script>
  </body>
</html>
```

Objeto window (Armazena as propriedades da janela que está sendo utilizada) pode-se usar o **window** para declarar as variáveis globais.

```
let cidade = "São Paulo";
console.log(window.cidade); //
Saída: undefined
```

Nesse caso a variável é criada no momento da atribuição.

Dessa forma a variável possui um escopo global (Pode-se acessar a variável de qualquer parte do Script)

(Não é Recomendado)

Sintaxe Básica

Nova forma de usar as variáveis

let/const veio para substituir o **var**

- O **let** e o **const** trabalham com a restrição de escopo;
- O valor de uma variável const deve obrigatoriamente ser definido na sua declaração;
- O uso do var não se tornou proibido, mas deve ser evitado.

Obs. O let e o const são novidade do **ECMAScript 6,** versões mais antigas dos browsers não irão aceitar essas definições.

Sintaxe Básica

Escopo (Global, Função e Bloco)

- **var** tem escopo de função, ou seja, se declarada dentro de uma função, só existe dentro dela.
- **let** tem escopo de bloco, ou seja, só existe dentro do {} onde foi declarada.

```
function exemplo() {
    if (true) {
        var a = "Sou var";
        let b = "Sou let";
    }
    console.log(a); // Funciona (var ignora o bloco e fica acessível na função)
    console.log(b); // Erro! (b só existe dentro do bloco if)
}
exemplo();
```

Sintaxe Básica

Declaração	Dentro de `if`, `for`, `while`	Fora de `if`, `for`, `while`	Fora da função
`var`	✅ Pode ser acessada	✅ Ainda existe	❌ Não pode ser acessada
`let`	✅ Pode ser acessada	❌ Não pode ser acessada	❌ Não pode ser acessada

Sintaxe Básica

No JavaScript, a diferença entre **let** e **const**:

Que **let permite que uma variável mude de valor**, **enquanto const não**

Nova forma de usar variáveis

```
<html>
  <head></head>
  <body>
    <script type = "text/javascript">
      let firstName
      const lastName = "Silva"
      firstName = "João"
      document.write(firstName + "<br/>")
      document.write(lastName + "<br/>")
    </script>
  </body>
</html>
```

Tipos de Variáveis

Tipos

Boolean: possuem apenas dois valores; verdadeiro ou falso

Undefined: indica que não foi definido um valor

Null: indica que o valor é nulo

Number: armazena valor numéricos

Infinity, -Infinity

NaN (not a number)

String: armazena textos

“texto” | ‘texto’ : aspas dupla ou aspas simples

Symbol: armazena símbolos

Objeto: armazena uma referência na memória

Operadores aritméticos

Javascript oferece 7 operadores aritméticos

Função	Operador
Soma	+
Subtração	-
Multiplicação	*
Divisão	/
Resto	%
Incremento	++
Decremento	--
Potenciação	**

Operadores de atribuição

Função	Operador
Atribuir	=
Atribuição de Soma	+=
Atribuição de subtração	-=
Atribuição de Divisão	/=
Atribuição de Multiplicação	*=
Atribuição de Resto	%=
Potenciação	**=

Sintaxe Básica

Operadores Unários

"-": Inverter o sinal de um número. Um número positivo se torna negativo e número negativo se torna positivo.

-3

```
let numero = -3;
console.log(-numero); // Saída: 3
```

"!": Inverte os valores do tipo boolean. Então um valor true se torna false e um false se torna true.

!true

"typeof": Ajuda a descobrir o tipo de um valor, se é String, Number e etc...

typeof 3

Concatenação de String

O operador “+” também pode ser usado no processo de concatenação de String.
Se pelo menos um dos operadores for uma string então será realizado a concatenação.

O resultado dessa operação será a string 2011 e não o número 31.

```
v1 = 20
v2 = “11”
V3 = v1 + v2
```

Operadores de comparação

Javascript especifica 7 operadores de comparação

Função	Operador
Maior	>
Menor	<
Igual	==
Não Igual	!=
Maior ou Igual	<=
Menor ou Igual	>=
Estritamente Igual	===

```
<script type="text/javascript">
    x = 5;
    document.write(x==5);
    document.write(x===5);
</script>
```

Sintaxe Básica

Operadores Lógicos

Função	Operador
And	&&
Ou	||
Não	!

Operadores Ternários

Assim como em Java e C++, em javascript é possível fazer uso do operador ternário.

Qual o resultado da variável valor após a execução do seguinte script:

```
<script type="text/javascript">
    x = 5;
    valor = (x!=5) ? "string teste" : 10
</script>
```

O operador ternário pode ser divido em três partes:

- (x!=5) : condição
- "string teste": Caso a resposta seja verdadeira o segundo elemento será retornado
- 10: Caso a resposta seja falsa o terceiro elemento é retornado

Conversão automática de tipos

Em muitas linguagens a conversão de tipos deve ser realizada manualmente. No entanto o javascript faz conversões automaticamente.

Isso pode deixar a linguagem mais dinâmica, mas se não for usado com cuidado pode causar muitos problemas e falhas lógicas, causando resultados inesperados.

```
"5" - 1:  4 = Subtração

"5" + 1:  51 = Concatenação
```

Expressões condicionais

Estrutura de seleção **if-else**

```
if(<condição>){ //Condicional Simples
  instrução 1;
  instrução 2;
}// Os dois comandos pertencem a estrutura condicional

if(<condição>){//Condicional Composta
  instrução 1;
}else{
  instrução 2;
  instrução 3;
}
//A instrução 1 pertence a estrutura "if" e as instruções 2 e 3 a
estrutura else.
```

Expressões condicionais

Condições Aninhadas

```
if(<condição1>){ //Condicional Aninhada
  instrução 1;
}else if(<condição2>){
  instrução 2;
}else if(<condição3>){
  instrução 3;
}else{
  instrução 4;
  instrução 5;
}
//A instrução 1 pertence a estrutura “if” de <condição1>
//A instrução 2 pertence a estrutura “else if” de <condição2>
//A instrução 3 pertence a estrutura “else if” de <condição3>
//A instruções 4,5 pertencem a estrutura “else”
```

Exemplo if-else

```
<html>
<body>
  <script type="text/javascript">
    var d = new Date();
    var hora = d.getHours();
    if(hora < 12){
      document.write("<b> Bom dia! </b>");
    }else{
      document.write("<b> Boa tarde! </b>");
    }
  </script>
</body>
</html>
```

Exercício

Criar uma página HTML que possua um script javascript que faz uso da função Math.random() para gerar números aleatórios e caso o valor for maior que 0.5 criar um link para a página do google. Caso contrário, criar um link para a página globo.com.

```
window.location.href = "http://página"
```

Desafio: Ao invés de criar o link tente redirecionar a página.

Sintaxe Básica

switch..case

Estrutura de seleção que compara vários valores de uma variável

```
switch(variavel){
    case valor:
        break;
    case valor:
        break;
    case valor:
        break;
    case valor:
        break;
    default:

}
```

Sintaxe Básica

Exemplo switch..case

```
<html>
<body>
  <script type="text/javascript">
    let dia = 5
    switch(dia){
      case 1:
        console.log("Domingo")
        break;
      case 2:
        console.log("Segunda-Feira")
        break;
      case 3:
        console.log("Terça-Feira")
        break;
      case 4:
        console.log("Quarta-Feira")
        break;
      case 5:
        console.log("Quinta-Feira")
        break;
      case 6:
        console.log("Sexta-Feira")
        break;
      case 7:
        console.log("Sábado")
        break;
      default: console.log("Não é um dia da semana")
    }
  </script>
</body>
</html>
```

O comando For

Em javascript, a estrutura “for” segue a mesma sintaxe da linguagem java

```
for(let i = valor_inicial; i <= valor_final; operação aritmética em i){
  //código a ser executado
}
```

Exemplo

```
for(let i = valor_inicial; i <= valor_final; i=i+incremento){
  //código a ser executado
}
```

Controles de Repetição

- **FOR IN**

```
for (variável que receberá o valor IN array de valor)
{

        //conjunto de instruções a serem
        executadas
}
```

Exemplo:

```
var valores = [1,2,3,4,5]
for (var numero in valores) {
}
        console.log("Número", numero);
```

Tecnologia para Front- end

Prof. Leandro Melo