

# Desenvolvimento WEB

## PARTE 3

Professor: Gilmar Luiz de Borba

2025-1



UNIVERSIDADE  
**FUMEC**

# Conteúdo

Cap. 13 – JavaScript – Breve Introdução

Cap. 14 – Estrutura de programação Sequência Simples

Cap. 15 – Estrutura de programação com desvio condicional IF

Cap. 16 – Estrutura de programação com desvio condicional CASE

Cap. 17 – Estrutura de programação repetição WHILE

Cap. 18 – Estrutura de programação repetição FOR

# Capítulo 13

## Breve Introdução

# JavaScript (JS) Introdução

- Criada pela *Netscape* (Brendan Eich) em 1994
- É uma linguagem de *script* para *WEB*
- Usada por bilhões de páginas
- Adiciona funcionalidades às páginas *HTML*
- É usada dentro das páginas *HTML*

# JavaScript (JS) Introdução

Índice TIOBE (Linguagens mais usadas), veja também: *Google Trend, LinkedIn e GITHub*

Mar 2025	Mar 2024	Change	Programming Language		Ratings	Change
1	1			Python	23.85%	+8.22%
2	3	▲		C++	11.08%	+0.37%
3	4	▲		Java	10.36%	+1.41%
4	2	▼		C	9.53%	-1.64%
5	5			C#	4.87%	-2.67%
6	6			JavaScript	3.46%	+0.08%
7	8	▲		Go	2.78%	+1.22%
8	7	▼		SQL	2.57%	+0.65%
9	10	▲		Visual Basic	2.52%	+1.09%
10	15	▲		Delphi/Object Pascal	2.15%	+0.94%
11	14	▲		Fortran	1.70%	+0.48%
12	9	▼		Scratch	1.66%	+0.21%
13	12	▼		PHP	1.48%	+0.16%
14	17	▲		Rust	1.23%	+0.20%
...						



# Comandos Javascript

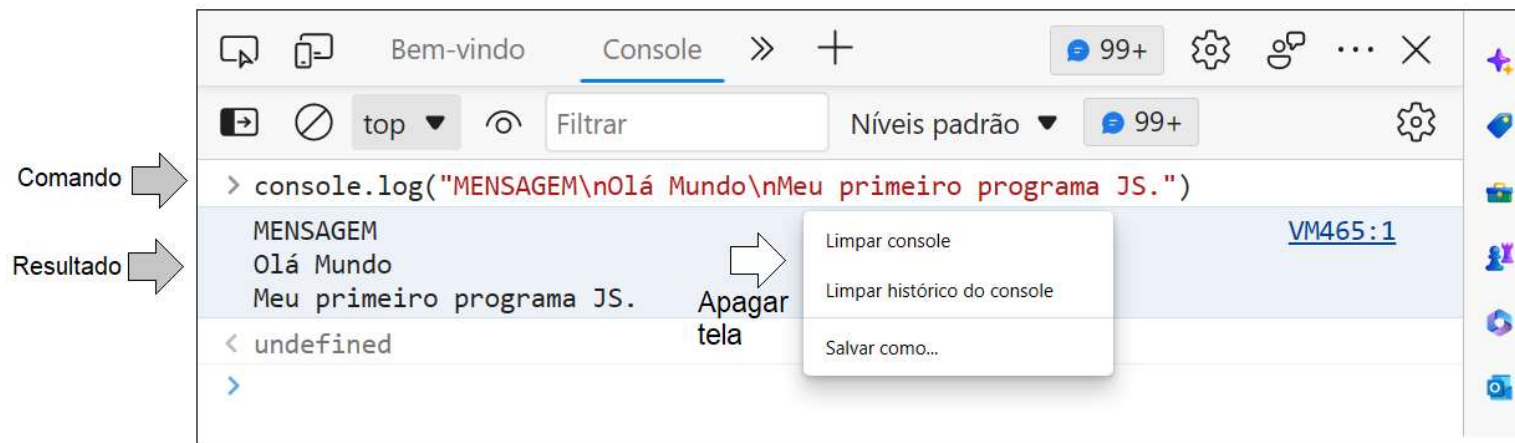
usando a console do navegador

# Executando JavaScript no navegador

Para executar comandos em Javascript, faça:

- 1 – Abra o navegador
- 2 – Acione a tecla F12
- 3 – Abra a console
- 4 – Digite o comando Javascript

Veja a console com o primeiro programa no navegador [Microsoft Edge](#)

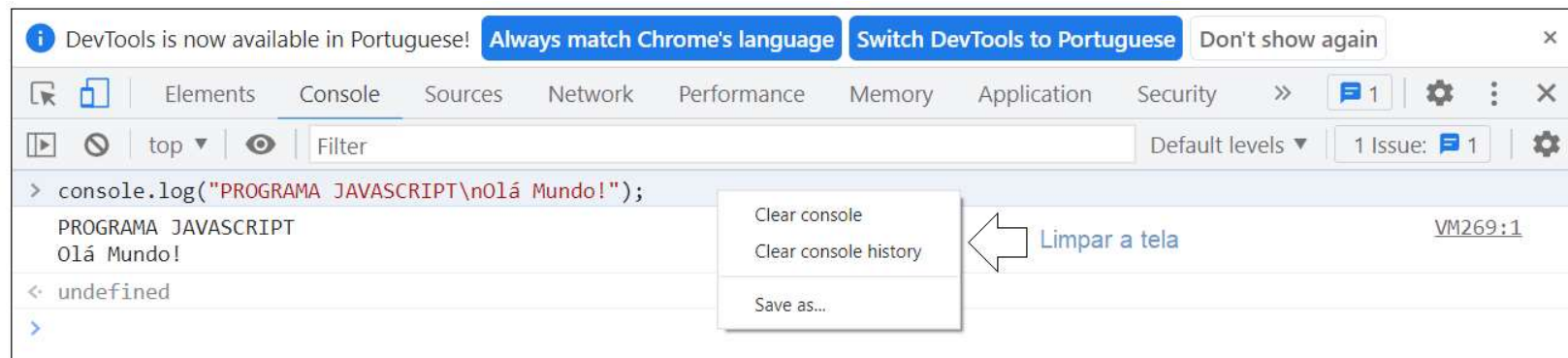


[VM465:1](#), acesso ao debugger e outros recursos.

# Executando JavaScript no navegador

Veja a console com o primeiro programa no navegador [Chrome](#)

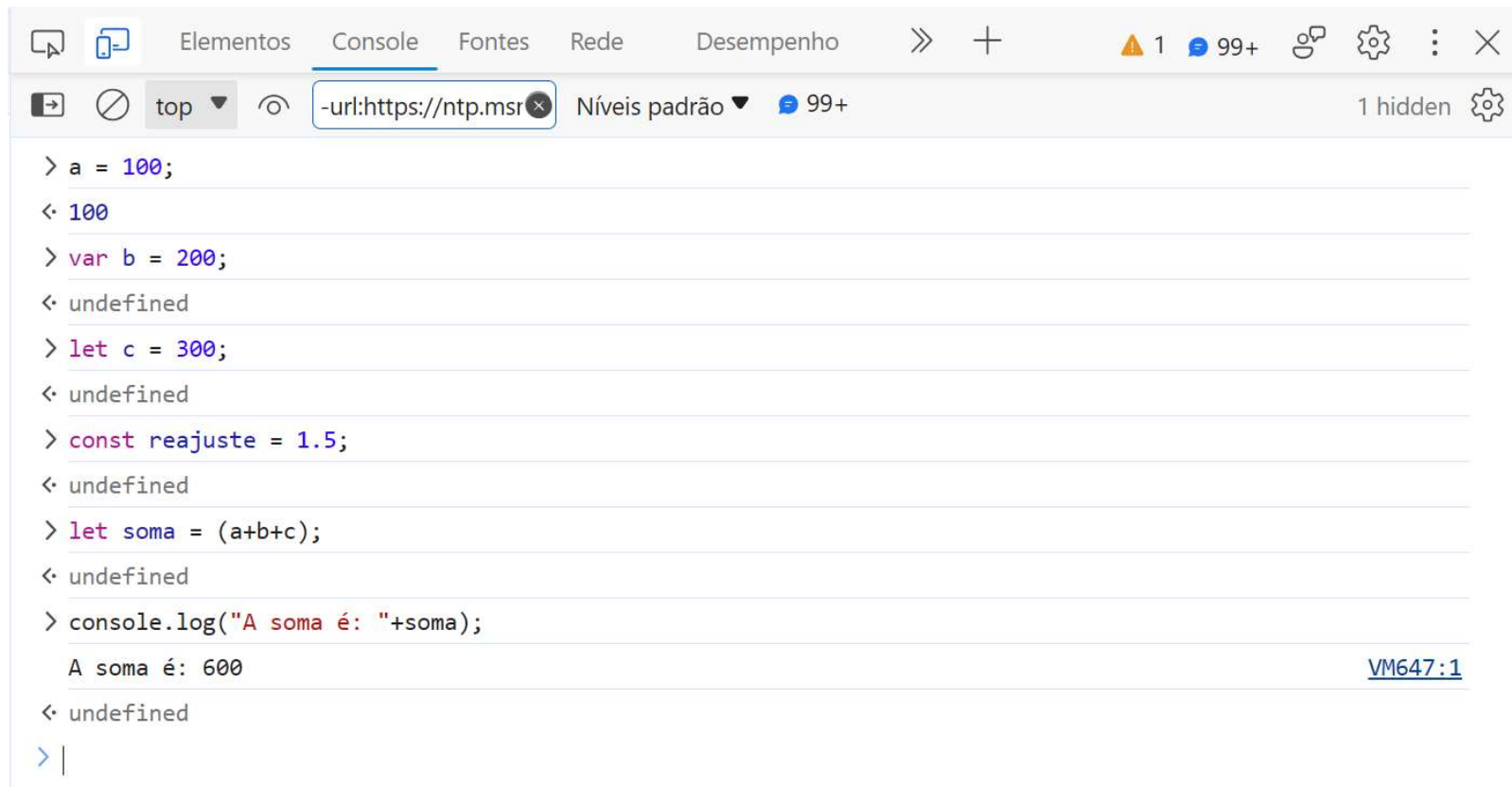
Comando →  
Resultado →



VM269:1, acesso ao debugger e outros recursos.



# Programa Console Browser



The screenshot shows a web browser's developer console with the 'Console' tab selected. The address bar shows a local file path: `-url:https://ntp.msr`. The console displays the following sequence of commands and outputs:

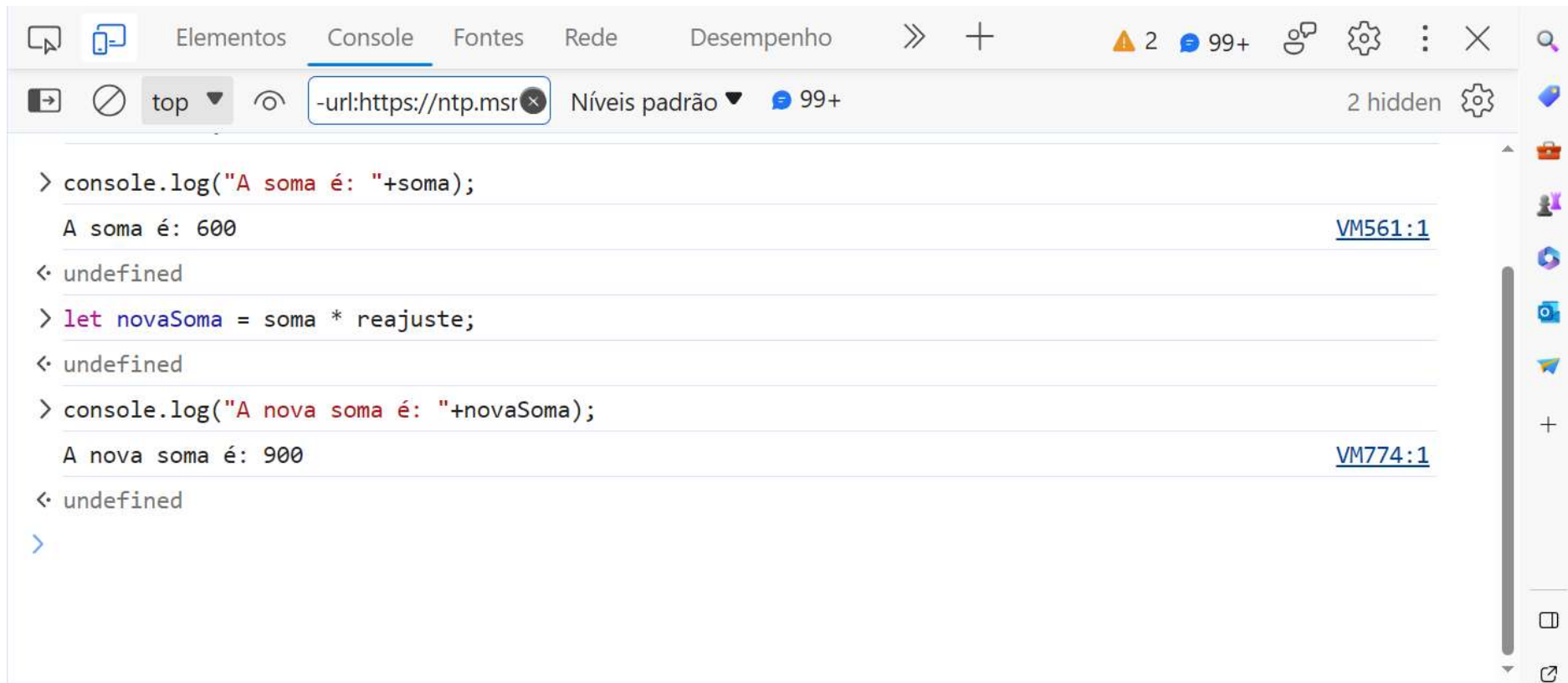
```
> a = 100;  
< 100  
> var b = 200;  
< undefined  
> let c = 300;  
< undefined  
> const reajuste = 1.5;  
< undefined  
> let soma = (a+b+c);  
< undefined  
> console.log("A soma é: "+soma);  
A soma é: 600  
< undefined  
> |
```

The output `A soma é: 600` is displayed in red text. A link `VM647:1` is visible at the end of the log entry.

**Observação 1:** “var” basicamente trabalha com um escopo global (e escopo inteiro da função). A palavra reservada “let” tem escopos de código, bloco, função e global. Veremos mais em capítulos posteriores.

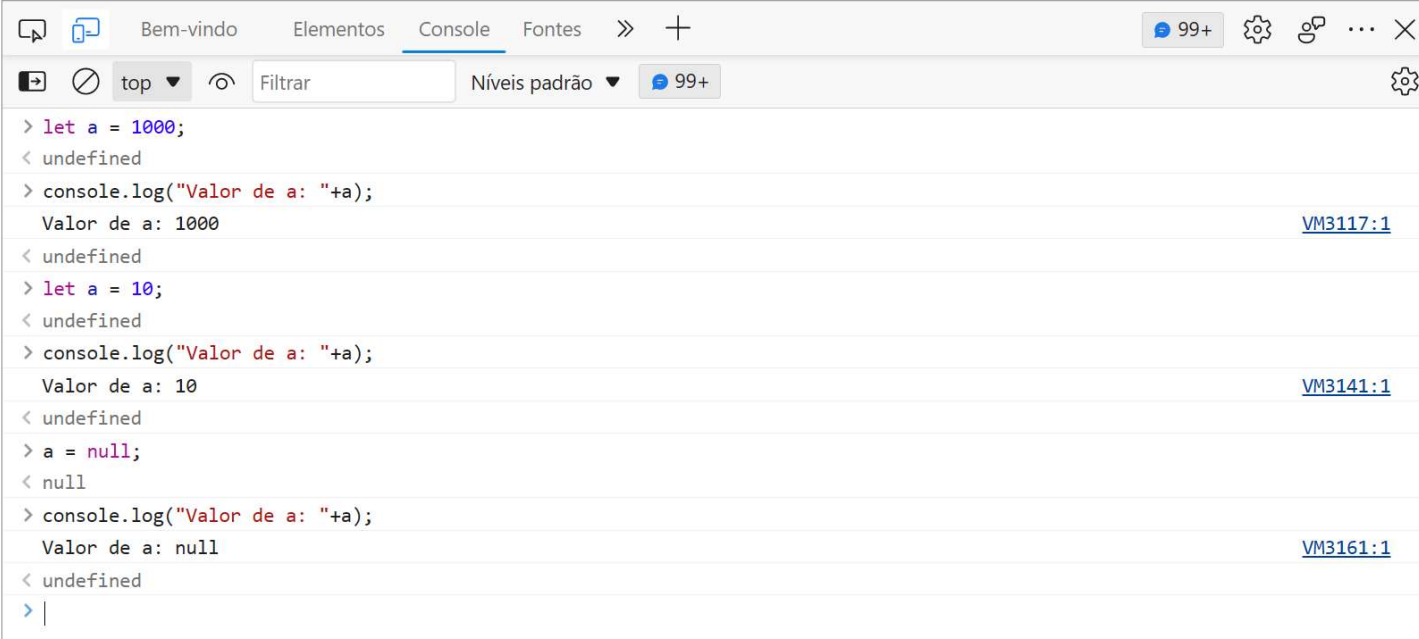
**Observação 2:** Javascript é case sensitive.

## Programa Console Browser (continuação)



# Programa Console Browser (O operador null)

Alterando valores de uma mesma variável. Usando o operador “null”



```
> let a = 1000;
< undefined
> console.log("Valor de a: "+a);
Valor de a: 1000
< undefined
> let a = 10;
< undefined
> console.log("Valor de a: "+a);
Valor de a: 10
< undefined
> a = null;
< null
> console.log("Valor de a: "+a);
Valor de a: null
< undefined
> |
```

11

## Nulo (null)

Representa a ausência explícita (quando declarada) de qualquer valor de objeto, ou variável. É tratado como falso para operações “booleanas”.

# Comandos Javascript

na página HTML

# JavaScript (JS) – Passo a Passo

## 1 – Criar o marcador *script* no *head* da página.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Engenharia de Software</title>
  <link rel="stylesheet" type="text/css" href="css/estilo.css">
  <script>
  </script>
</head>
```



Ou `<script type="text/javascript">`

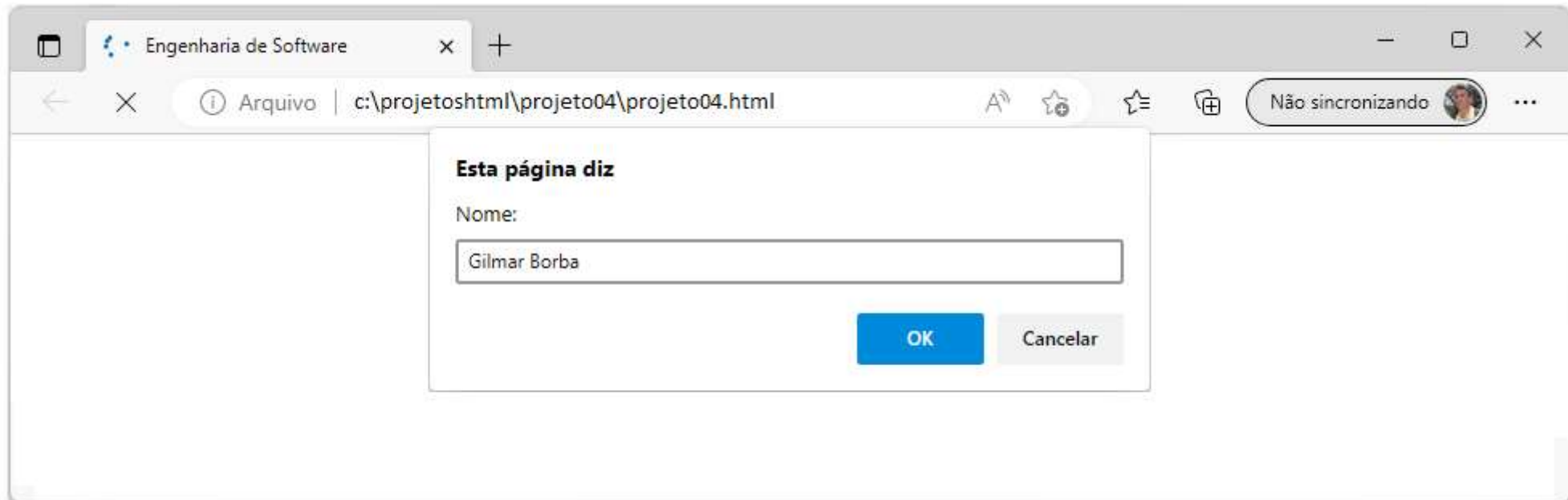
## JavaScript (JS) – Passo a Passo

2 – Inserir o código usando a linguagem *JS*, nesse caso vamos personalizar o *site* com o nome do usuário. No `<head>` da página, crie o bloco `<script>`, dentro desse bloco `<script>`, escreva o código:

```
<script>  
|      var nome = prompt("Nome: ")  
</script>
```

# JavaScript (JS) – Passo a Passo

Ao salvar o código e executar a página, observe o resultado:



# JavaScript (JS) – Passo a Passo

3 – Após a marcação `<h1>`, que está no `<body>` do documento, criar uma classe (nesse caso “nome”), criar um novo marcador `<script>` para mostrar/escrever (*write*) o nome informado na página *HTML*.

```
...  
<h1>O que é a Engenharia de Software?</h1>  
  
<p class="nome">  
  <script type="text/javascript">  
    // Escreve o conteúdo da variável nome  
    if (nome != null)  
      document.write("(Usuário: "+nome+"")  
    else  
      document.write("Usuário não informado")  
  </script>  
</p>
```



# JavaScript (JS) – Passo a Passo

## 3 – Outra opção para a solução do slide anterior.

```
<script>
    if (nome == null || nome.trim() == "") {
        alert("MENSAGEM DO PROGRAMA:\n" +
            "O nome deve ser informado!");
        window.location.reload(false);
    }
    else {
        document.write("Usuário: " + nome);
    }
</script>
```

Observação:

`location.reload()` é um método do objeto `window.location`

Se **"false"** ou vazio, o navegador pode usar uma versão em cache da página.

Se **"true"** o navegador sempre recarrega do servidor.

Se usar `window.location.reload()` no script sem condição -> loop infinito.

# JavaScript (JS) – Passo a Passo

4 – Abrir a folha de estilo (arquivo .css). Inserir a chamada à classe criada:

```
.nome {  
    text-align: center;  
    font-size: 1em;  
    color: ■ #006400;  
    font-family: Calibri;  
}
```

# JavaScript (JS) – Passo a Passo

Ao salvar o código e executar a página, observe o resultado:



# JavaScript (JS) – Criando botões no formulário

Seja o código HTML para criar um botão no formulário:

```
<button title="Informações sobre o site" name="btn_close" value="ok" type="button" onclick="mensagem()">Sobre a página</button>
```

## 1. title="Informações sobre o site"

Define o texto (tooltip), ou dica ao passar o mouse no botão.

## 2. name="btn\_close"

Atribui um nome ao botão, esse será usado para identificar o botão em formulários ou em scripts.

# JavaScript (JS) – Criando botões no formulário

Continuação ...

## 3. `value="ok"`

É o valor associado ao botão, que poderá ser enviado em um formulário ou script.

## 4. `type="button"`

Tipo do botão.

`button`: Um botão genérico. `submit`: Envia o formulário em que o botão está contido. `reset`: Apaga os campos do formulário.

## 5. `onclick="mensagem()"`

Ao clicar no botão será executado o código JavaScript ou a função.

# JavaScript (JS) – Criando botões no formulário

## Código do método mensagem()

Continuação ...

```
<script>
  function mensagem() {

    const tempo = new Date();
    var dados =
      "MENSAGEM:\n" +
      "Página..: index.html\n" +
      "Data....: " + tempo.toLocaleDateString('pt-BR') + "\n" +
      "Versão..: 1.0"
    alert(dados);
    console.log(dados);
  }
</script>
```

# JavaScript (JS) – Criando botões no formulário

```
const tempo = new Date();
```

Cria uma nova instância do objeto Date, representa o momento atual.

```
var dados = "..."
```

Declara e inicializa a variável dados, que contém uma string formatada

```
tempo.toLocaleDateString('pt-BR');
```

Data atual formatada no padrão brasileiro (dd/mm/aaaa)

## 6. Sobre a página

Rótulo visível no botão, conteúdo textual.

# JavaScript (JS) – toLocaleString() e toLocaleDateString()

## toLocaleString()

Retorna data e hora, de acordo com as configurações regionais do sistema.

```
const agora = new Date();  
console.log(agora.toLocaleString());  
Retorna: "17/04/2025 10:15:00" (formato brasileiro)
```

## toLocaleDateString()

Retorna somente a data, também formatada localmente.

```
const hoje = new Date();  
console.log(hoje.toLocaleDateString());  
Retorna: "17/04/2025"
```

Uma outra opção é o [toLocaleTimeString\(\)](#), retorna somente a hora.



# JavaScript (JS) – Criando botões no formulário

Mensagem alert():



Mensagem na console:



# Criando um botão JavaScript na página

## Button

Representa um botão que pode ser acionado para executar uma ação específica.

## Parâmetros

**Title:** uma mensagem de ajuda "*tip*" ou "*hint*"

**Name:** nome de identificação do botão no código

**Value:** retorne o valor do atributo "value" de um botão

```
var x = document.getElementById("myBtn").value;
```

**Type:** tipo do elemento

**OnClick:** evento que ocorrerá quando o botão for "clicado"

Botão sem formatação:

[ Mensagem ] [ Fechar ]

# Criando um botão JavaScript na página

## Buttom

alert("Mensagem"): emite uma mensagem para o usuário

close(): Fecha a aba do navegador

```
<button title="Se você clicar aqui a página será fechada ..." name="btn_close"
  value="OK" type="button" onclick="fechar()">Fechar Formulário</button>

<script>
  function fechar() {
    alert("O formulário será fechado");
    close();
  }
</script>
```

d:\profgilmarborba\desenvweb\imagens\

### Esta página diz

O formulário será fechado

OK

# Criando um botão JavaScript na página

## Buttom

Alguns caracteres de *scape*:

\“ → insere o *character* “ dentro de uma *string* (cadeia de caracteres)

\n → insere uma quebra de linha em uma *string*.

```
<button title="Ver mensagem da página ..." name="btn_msg" value="OK" type="button"
  onclick="mensagem()">Mensagem</button>

<script>
  function mensagem() {
    alert("MENSAGEM DO SISTEMA\n\n" +
      "Olá! Você clicou no botão \"Mensagem\".Essa página trata o uso de \"header\" " +
      "transparente e também botões de comando.");
  }
</script>
```

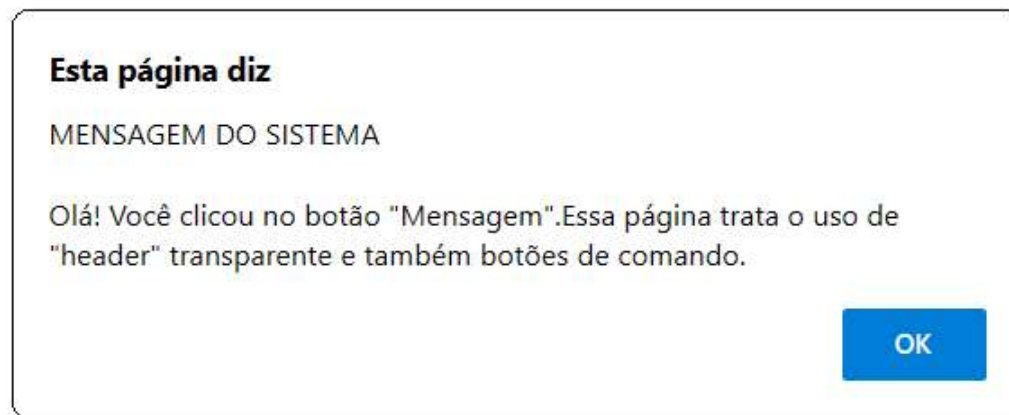
## Criando um botão JavaScript na página

### Buttom

Alguns caracteres de *scape*:

\“ → insere o *character* “ dentro de uma *string* (cadeia de caracteres)

\n → insere uma quebra de linha em uma *string*.



## Criando um botão JavaScript na página

### Buttom - Formatação

Inserir um id no botão:

```
<button id="botoes" class="botao" title="Mostrar mensagem da página" ...
```

Aplicar o estilo:

```
#botoes {  
  border: 3px;  
  border-style: groove;  
  background-color: #808080;  
  color: #fff;  
  text-align: center;  
  padding: 5px;  
  font-weight: bold;  
}
```

Detalhe do botão "fechar"  
formatado:





# Capítulo 14

Estrutura de programação

Sequência Simples



# Estrutura sequencial

Seja um programa para receber dois números inteiros, calcular e imprimir a soma e o produto.

IHM (Interface Homem Máquina) do programa.



The screenshot shows a web browser window with the address bar displaying "Arquivo | D:/projetoshtmlcssjs/somaemultiplica/index.html". The main content area has a heading "Soma e multiplica". Below the heading are two input fields: "Valor A:" with the value "4" and "Valor B:" with the value "8". A "Calcular" button is positioned below these fields. At the bottom, the results are displayed: "Soma: 12" and "Produto: 32".

# Estrutura sequencial

Recebe dois números inteiros,  
Calcula e imprime a soma e o produto.

Componentes usados:

**Soma e multiplica**

Valor A:

Valor B:

**Soma: 12**  
**Produto: 32**

document.getElementById (para acessar os campos)  
innerHTML (para escrever tags HTML)

# Estrutura sequencial

## Parte HTML do programa

```
<body>
  <div>
    <h2>Soma e multiplica</h2>
    <p>
      <label for="a">Valor A: </label>
      <input type="number" id="a" placeholder="Digite o valor para a">
    </p>

    <p>
      <label for="b">Valor B: </label>
      <input type="number" id="b" placeholder="Digite o valor para b">
    </p>
    <button onclick="calcular()">Calcular</button>

    <div class="resultado" id="resultado"></div>
  </div>
```

Explicações na próxima página...

# Estrutura sequencial

## Parte HTML do programa

O parâmetro `placeholder` serve para exibir um texto breve e informativo dentro do campo de entrada.

```
<p>  
  <label for="a">Valor A: </label>  
  <input type="number" id="a" placeholder="Digite o valor para a">  
</p>
```

Valor A:  ← Texto do placeholder

Valor B:

`<input>` em HTML é usado para criar campos de entrada de dados em formulários web.

No input é possível alterar o comprimento da caixa, para isso use a propriedade `size`.

Exemplo: `size = "12"`

`<type="number">` é o tipo de dados de entrada, nesse caso um número.

# Estrutura sequencial

## Parte HTML do programa

`<label for="a">`

é o label referenciado para a variável "a" do id do input.

`<div class="resultado">`

será usado para estilo CSS

```
<body>
  <div>
    <h2>Soma e multiplica</h2>
    <p>
      <label for="a">Valor A: </label>
      <input type="number" id="a" placeholder="Digite o valor para a">
    </p>
    <p>
      <label for="b">Valor B: </label>
      <input type="number" id="b" placeholder="Digite o valor para b">
    </p>
    <button onclick="calcular()">Calcular</button>

    <div class="resultado" id="resultado"></div>
  </div>
```

`<div id="resultado">` será usado para referência no programa JavaScript. A referência será `resultadoDiv`.

# Estrutura sequencial

## Parte JavaScript do programa

```
<script>
  function calcular() {
    let a = document.getElementById("a").value;
    let b = document.getElementById("b").value;
    if (a == "" || b == "") {
      alert("Impossível calcular para valores nulos!");
      return;
    }
    if (a <= 0 || b <= 0) {
      alert("Números negativos não são permitidos!");
      return;
    }

    let soma = parseInt(a) + parseInt(b);
    let produto = a * b;

    let resultadoDiv = document.getElementById("resultado");
    resultadoDiv.innerHTML = `<br><br><b>Soma: </b>${soma}<br>
      <b>Produto: </b>${produto}`
  }
</script>
```

(1) Atribui valores dos "inputs" para as variáveis "a" e "b"

(2) Faz as validações dos valores entrados nos "inputs"

(3) Processamento. O `parseInt()` é usado para converter o String para número e evitar a concatenação.

(4) Atribuição do resultado à variável `resultadoDiv` criada. Criação dinâmica dos resultados em HTML.

# Estrutura sequencial

## INTERPOLAÇÃO DE VARIÁVEIS

Template Strings (com crase ``...``)

São strings que permitem interpolação de variáveis, ou seja, permitem inserir valores dentro de uma *string*, o *template* é criado com delimitadores com crases (```), e não por aspas simples ou duplas.

O conteúdo da variável é inserido dentro da *string* no lugar do operador `${...}`.

# Estrutura sequencial

## Informações adicionais

### DOM (Document Object Model)

É uma interface de programação que permite que o JavaScript interaja com o HTML. É uma estrutura criada para representar o documento HTML, XML etc. É uma forma de fazer a conexão do JavaScript com o HTML durante a interpretação das páginas no navegador.

### `document.getElementById()`

Esse comando em JavaScript serve para retornar um elemento HTML do DOM (Document Object Model) que tem um ID especificado pelo programador. É usado para manipular os elementos do programa através do JavaScript.

`getElementById()` é chamada no objeto *document*, que representa o documento HTML.

Se não for encontrado o elemento ID, a função retorna *null*.



# Estrutura sequencial

## Informações adicionais

### innerHTML

Essa propriedade JavaScript serve para acessar e modificar o conteúdo HTML de um elemento do programa.

```
let x = document.getElementById("elemento").innerHTML;
```

Nesse caso atribui a `x` o conteúdo do elemento "elemento"

```
document.getElementById("elemento").innerHTML = "novo conteúdo";
```

Nesse caso modifica o conteúdo do elemento substituindo pelo conteúdo da string, nesse caso "novo conteúdo".

## Estrutura sequencial

Botão (<button>) para fechar o formulário

### Soma e multiplica

Valor A:

Valor B:

Criar o botão na parte HTML da página.

Criar a função fechar() usando o método adequado.



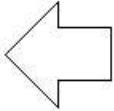
# Estrutura sequencial

## Código para a função fechar()

Criar o botão para fechar o formulário:

```
<script>
  function calcular() {
    let a = document.getElementById("a").value;
    let b = document.getElementById("b").value;
    ...
  }

  function fechar() {
    window.close();
  }
</script>
```



## Fechamento de Páginas/abas

Para fechar um formulário ou uma aba do navegador usando JavaScript, é possível usar os seguintes métodos:

(1) `window.close();`

Esse comando fecha o formulário ou página, porém o navegador pode impedir o fechamento de janelas que não foram abertas pela própria aplicação JavaScript usando o `window.open()`.

(2) `document.getElementById("caixaDeDialogo").close();`

Use esse comando caso o formulário esteja dentro de um `<dialog>`.

(3) `document.getElementById("Formulario").style.display = "none";`

Use esse comando para esconder o formulário, caso tenha sido usado o `display: block`.

Importante: Para o SO (Windows) o processo continuará em execução.

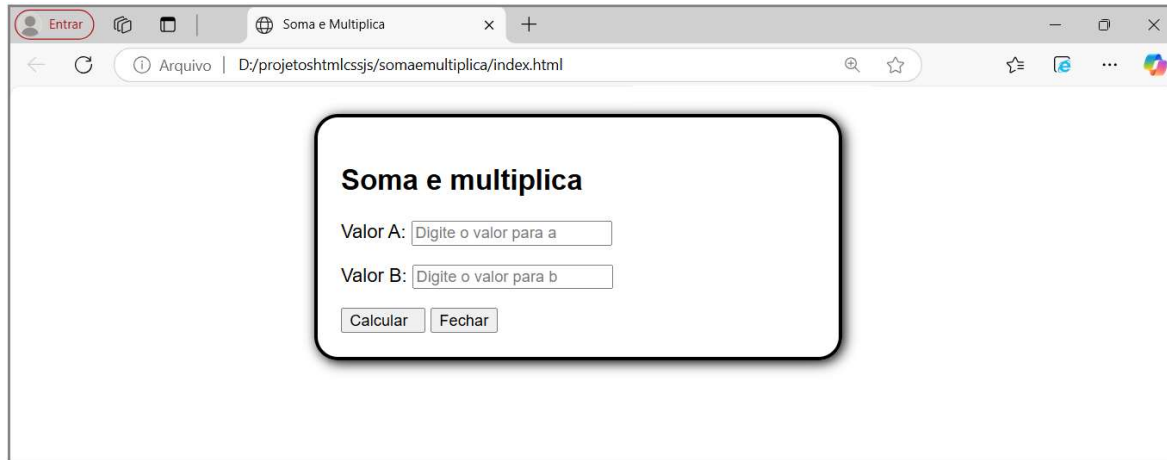
(4) No caso de usar o Bootstrap, pode ser usado:

```
var modal = new bootstrap.Modal(document.getElementById('telaModal'));  
modal.hide();
```

# Estrutura sequencial

## Alterando o formato da janela

Para customizar a janela conforme mostrado faça as seguintes alterações




- Crie uma classe para a **div** do formulário (parte HTML).
- Marque os estilos adequados. Veja na próxima página.

# Estrutura sequencial

## Alterando o formato da janela

Criando a classe para a div (na parte HTML) da página:

```
<body>  
  <div class="formato">   
    <h2>Soma e multiplica</h2>  
    ...
```

- Marcar os estilos adequados. Veja na próxima página.

# Estrutura sequencial

## Alterando o formato da janela

Para customizar a janela conforme mostrado faça as seguintes alterações



- Veja o estilo para a classe formato na próxima página.
- A mensagem dentro do componentes input é conseguida com a propriedade: **placeholder**.
- O tamanho do input pode ser alterado coma propriedade: **size**.



# Estrutura sequencial

## Alterando o formato da janela

Estilos para a classe formato:

```
.formato {  
    max-width: 400px;  
    margin: auto;  
    padding: 20px;  
    border: 3px solid ■ #000;  
    border-radius: 20px;  
    box-shadow: 2px 2px 10px rgba(0, 0, 0, 2.5);  
}
```

box-shadow: x, y, desfoque, espalhamento, cor, inset (sobreamento dentro da caixa/form);

x = deslocamento eixo x,                      y = deslocamento eixo y

(Para mais detalhes, veja teoria parte 1)

# Capítulo 15

Estrutura de programação

Condicional

# Estrutura Condicional

Implementar um programa para receber o salário bruto, o desconto e a comissão. Ao final da entrada de dados o programa deverá calcular e imprimir o salário líquido ( $\text{salarioLiquido} = \text{salarioBruto} - \text{descontos} + \text{comissao}$ ) e o tipo de cartão que o empregado recebe (Bronze ou Prata ou Ouro).

## Objetivos:

- Exemplificar a estrutura de programação condicional ( if / else)
- Trabalhar com componentes <input type ...>
- Trabalhar com referência a resultadoDiv
- Entender a propriedade innerHTML, que define/retorna o conteúdo HTML de um elemento.

# Estrutura Condicional

## Siga os passos:

- Crie um novo projeto em “projetoshtml”: **projeto15**.
- No projeto projeto15, crie o arquivo index.html.
- Crie o stub do arquivo, faça as alterações recomendadas no arquivo.
- No bloco <body> faça a programação: parte HTML.
- Em seguida, faça a programação da parte JAVASCRIPT.

... siga o passo a passo nas próximas páginas ...

## Estrutura condicional

IDE (Interface Homem Máquina) do programa.

### **Salário Líquido e Classificação de Cartão**

Salário Bruto:

Descontos:

Comissão:

Informe um valor inteiro positivo

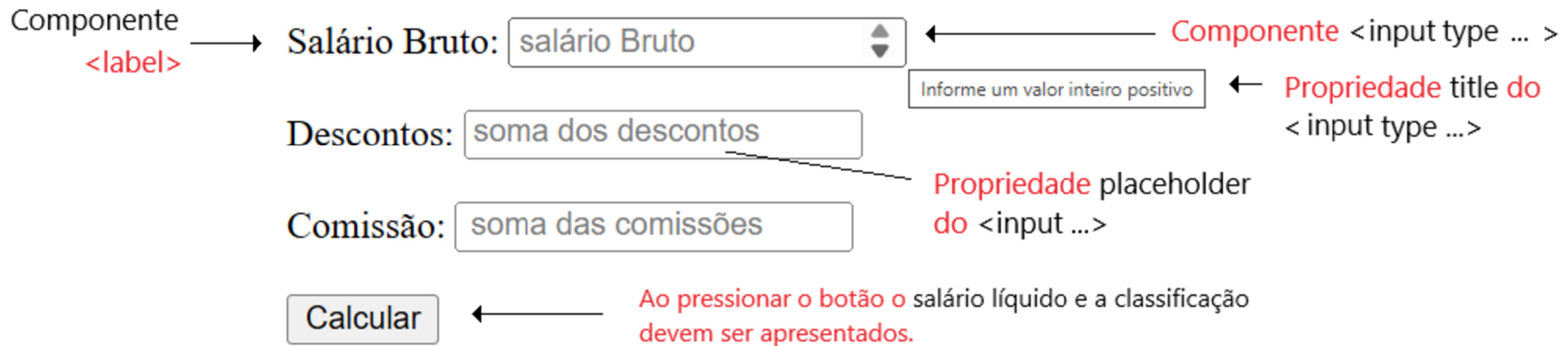
**Salário Líquido:**

**Classificação do Cartão:**

# Estrutura condicional

IDE (Interface Homem Máquina) do programa (Detalhamento).

## Salário Líquido e Classificação de Cartão



**Salário Líquido:**

**Classificação do Cartão:**

} document.getElementById (para acessar os campos)  
innerHTML (para escrever tags HTML)

## Estrutura condicional

Tabela de classificação do cartão em função do salário líquido.

Salário Líquido	Cartão
< 5000,00	Cartão Bronze
$\geq 5000,00$ e $< 8000,00$	Cartão Prata
$\geq 8000$	Cartão Ouro

# Estrutura condicional

## Parte HTML do Programa:

```
<body>
  <h2>Salário Líquido e Classificação de Cartão</h2>
  <label for="salarioBruto">Salário Bruto:</label>
  <input type="number" id="salarioBruto" step="100"
  placeholder="salário Bruto" title="Informe um valor inteiro positivo"><br><br>

  <label for="descontos">Descontos:</label>
  <input type="number" id="descontos" step="50"
  placeholder="soma dos descontos" title="Informe um valor inteiro positivo"><br><br>

  <label for="comissao">Comissão:</label>
  <input type="number" id="comissao" step="50"
  placeholder="soma das comissões" title="Informe um valor inteiro positivo"><br><br>

  <button onclick="calcularSalarioLiquido()">Calcular</button>

  <h3>Salário Líquido: <span id="salarioLiquido"></span></h3>
  <h3>Classificação do Cartão: <span id="classificacaoCartao"></span></h3>

  <span>
```

É um elemento genérico usado para quebrar seções de texto no que se refere a questões de estilo. Basicamente é usado para estilizar partes específicas de um texto, sem alterar a estrutura do documento.



# Estrutura condicional

## Parte JavaScript do Programa:

```
<script>
```

```
function calcularSalarioLiquido() {  
  let salarioBruto = parseFloat(document.getElementById('salarioBruto').value) || 0;  
  let descontos = parseFloat(document.getElementById('descontos').value) || 0;  
  let comissao = parseFloat(document.getElementById('comissao').value) || 0;  
  
  let salarioLiquido = salarioBruto - descontos + comissao;  
  document.getElementById('salarioLiquido').textContent = salarioLiquido.toFixed(2);  
  
  let classificacao;  
  
  if (salarioLiquido < 5000) {  
    classificacao = "Cartão bronze";  
  } else if (salarioLiquido <= 8000) {  
    classificacao = "Cartão prata";  
  } else {  
    classificacao = "Cartão ouro";  
  }  
  
  document.getElementById('classificacaoCartao').textContent = classificacao;  
}
```

```
</script>
```

Se o resultado à esquerda for "falsy" considera o valor "0" (zero)

Fixa duas casas decimais para impressão do valor (float)

## Estrutura condicional

### SOBRE O "FALSY" e "TRUTHY"?

O JavaScript considera "falsy" (falsas):

Resultado da operação booleana	Descrição
false	Falso (booleano)
0	Zero
" "	String vazia
null	Nulo
undefined	Variável foi declarada, mas não recebeu nenhum valor
NaN	Não é um número.

O restante é considerado "truthy" (verdadeiro).

# Estrutura condicional

## SOBRE O "FALSY" e "TRUTHY"?

**Truthy** é um resultado dado como verdadeiro em um contexto booleano. Exemplos de valores **truthy** incluem:

- Qualquer valor diferente de zero; qualquer string não vazia, palavra-chave true.

## DIFERENÇA ENTRE TRUE E TRUTHY

- **True** é um valor booleano.
- **Truthy** é quando uma condição é considerada verdadeira.

O mesmo pode ser considerado para false e falsy.

## Estrutura condicional simples

### Testando o programa

Classificação Bronze (salário líquido inferior a 5.000,00)

#### **Salário Líquido e Classificação de Cartão**

Salário Bruto:

Descontos:

Comissão:

**Salário Líquido: 2000.00**

**Classificação do Cartão: Cartão bronze**

## Estrutura condicional simples

## Testando o programa

Classificação Prata (salário líquido superior ou igual a 5.000,00 e inferior a 8.000,00)

### Salário Líquido e Classificação de Cartão

Salário Bruto:

Descontos:

Comissão:

**Salário Líquido: 5500.00**

**Classificação do Cartão: Cartão prata**

## Estrutura condicional simples

### Testando o programa

Classificação Ouro (salário líquido superior a 8.000,00)

#### **Salário Líquido e Classificação de Cartão**

Salário Bruto:

Descontos:

Comissão:

**Salário Líquido: 12000.00**

**Classificação do Cartão: Cartão Ouro**

# REFERÊNCIAS

*Castro , Elizabeth; Hyslop, Bruce. HTML5 e CSS3: Guia Prático e Visual. EditoraAlta Books. Edição7ª. 2013.*

*Flanagan, David. JavaScript: O Guia Definitivo. Trad. João Eduardo Nóbrega Tortello. Editora Bookman. 6ª Edição. 2012.*

*Freeman, Eric; Robson, Elisabeth. Use a Cabeça!: Programação JavaScript. EditoraAlta Books. 1ª Edição. 2016. ISBN-13978-8576089902.*

*Sampaio, Cleuton. JavaScript de Cabo a Rabo. Editora Ciência Moderna. 2015. ISBN-13978-8539906581.*

*Silva, Maurício Samy. CSS Grid Layout: Criando Layouts CSS Profissionais. Editora Novatec. 1ª Edição. 2017. ISBN-13978-8575226322.*

# REFERÊNCIAS

*W3C ORG - Making the Web work*

Localizado em: <https://www.w3.org>

Acessado em: outubro/2024

*Internacionalização - Dicas rápidas para a Web*

Localizado em: <https://www.w3.org/International/quicktips/index.pt>

Acessado em: outubro/2024

*Iperius Backup*

Entendendo os conceitos entre os modelos TCP/IP e OSI

Localizado em: <https://www.iperiusbackup.net/pt-br/entendendo-os-conceitos-entre-os-modelos-tcpip-e-osi/>

Acessado em: outubro/2024

64 *JavaScript - mdn web docs*

Localizado em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

Acessado em: outubro/2024



## REFERÊNCIAS

*Learn Responsive Design*

*Localizado em: <https://web.dev/learn/design/>*

*Acessado em: outubro/2024*

*w3Schools - HTML: The language for building web pages*

*localizado em: <https://www.w3schools.com/>*

*Acessado em: outubro/2024*

*Tabela de Cores HTML (hexadecimal e RGB) - Amostras de cores para HTML e CSS em código hexadecimal e código RGB*

*Localizado em: <https://erikasarti.com/html/tabela-cores>*

*Acessado em: outubro/2024*