Curso de Jogos Digitais

Disciplina de Tecnologias Web

Aula 06 – Introdução ao JavaScript

Professor: André Flores dos Santos

Santa Maria – RS 2025

SUMÁRIO

Introdução

- JavaScript é uma linguagem de programação essencial para criar websites interativos e dinâmicos.

- O JavaScript foi criado em 1995 por Brendan Eich, inicialmente como uma linguagem de script para tornar as páginas web mais dinâmicas.

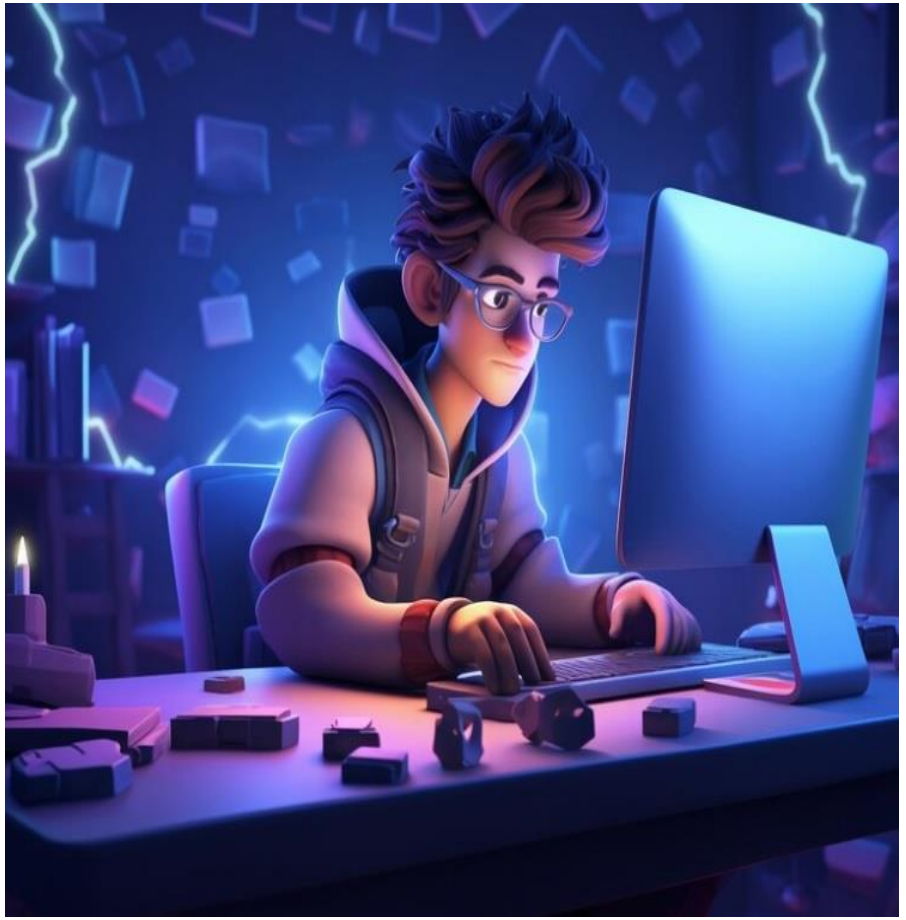
- Desde então, o JavaScript evoluiu significativamente e se tornou uma das linguagens de programação mais populares do mundo.

Introdução

- JavaScript é uma linguagem de programação principalmente tratada no cliente (client side), ou seja, executada localmente.

- Caso se pretenda segurança de dados sensíveis, então deve ser tratado no servidor (server side), ou seja, executada remotamente.

- Tem diversos usos, tais como validar formulários, alteração de estilo, inserção dinâmica de conteúdo, comunicar-se com servidor (usando o recurso *ajax*)



Introdução

- Enquanto HTML é usado para estruturar o conteúdo de uma página web e CSS para estilizá-la, JavaScript é responsável pela interatividade e dinamismo.

- Juntos, essas três tecnologias formam a base do desenvolvimento web moderno.

Introdução

- JavaScript é amplamente utilizado em uma variedade de contextos, desde pequenos scripts em páginas web até aplicativos web complexos e robustos.
- Exemplos de aplicações práticas incluem galerias de imagens interativas, sistemas de login e autenticação, jogos online e muito mais.
- Existem muitos recursos e ferramentas disponíveis para aprender e trabalhar com JavaScript, incluindo documentação oficial, tutoriais online, comunidades de desenvolvedores e editores de código.

Introdução

- Documentação recomendada (MDN Web Docs):

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript

- Também tem a especificação oficial do **ECMAScript** feita pela ECMA International:

https://tc39.es/ecma262/

Este site oferece uma documentação abrangente e atualizada sobre todos os aspectos da linguagem JavaScript, incluindo sintaxe, recursos, métodos, propriedades e muito mais. É uma excelente fonte de referência para desenvolvedores de todos os níveis de experiência.

Introdução

- Pode-se inserir código JavaScript:
  - Direto no corpo através do conteúdo da **tag <script>**
  - Através de arquivos externos em conjunto com o atributo src da *tag script.*
  - Via eventos, diretamente nos atributos de evento dos elementos por exemplo: *onclick* e *onblur.*

  - É usado para adicionar funcionalidades dinâmicas às páginas web, como animações, validação de formulários, manipulação de eventos e muito mais.

Exemplos

```
<!-- Corpo da tag SCRIPT -->
<script>
    var teste = "Ola";
</script>

<!-- Attributo src -->
<script src="js/exemplo.js"></script>

<!-- Atributos de eventos -->
<button onclick="alert('Ola')">Teste</button>
```

JavaScript no **Corpo da Página**

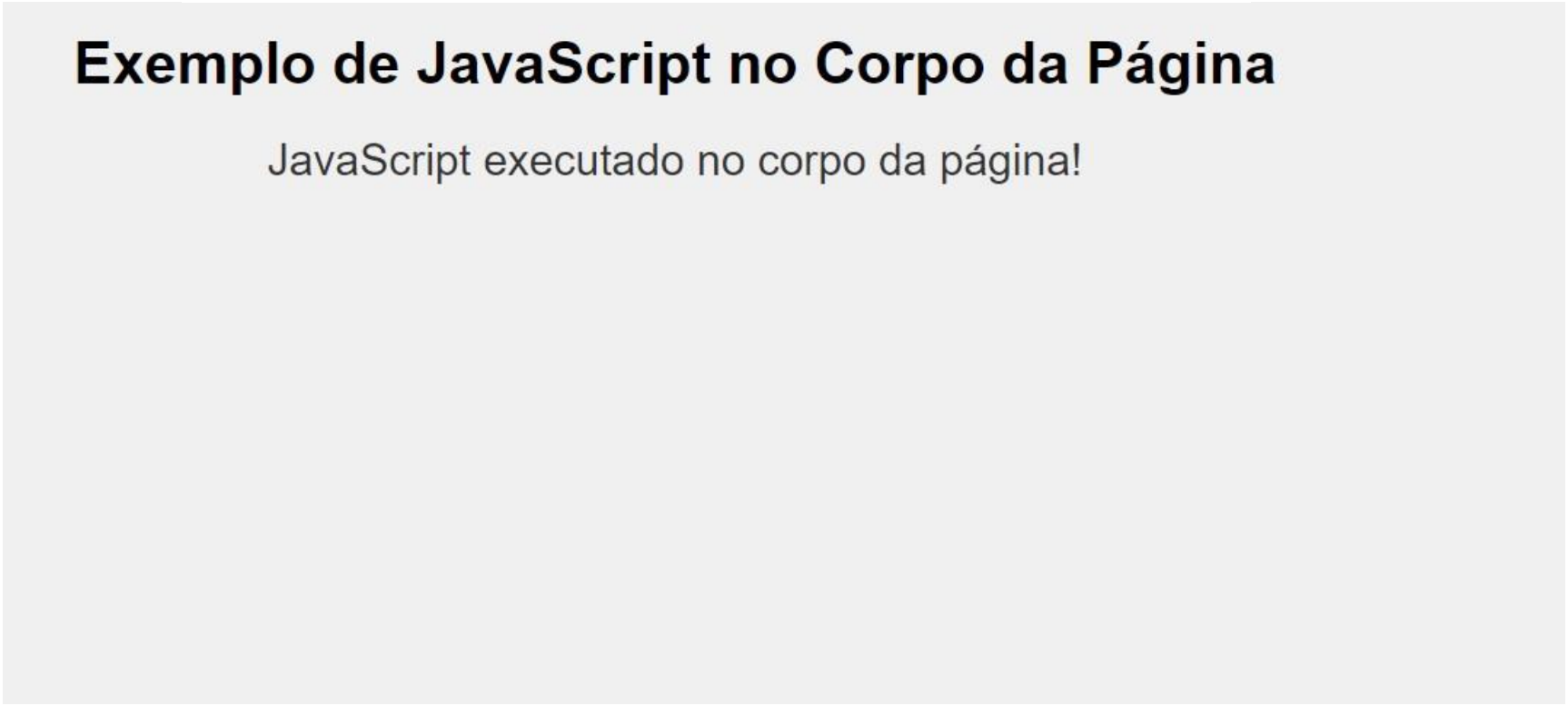
```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-
width, initial-scale=1.0">
  <title>Exemplo de JavaScript no Corpo da
Página</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f0f0f0;
      text-align: center;
      padding-top: 50px;
    }
    #javascript-msg {
      font-size: 24px;
      color: #333;
    }
  </style>
</head>
```

Basta escrever o código entre as tags **<script>** e **</script>**

```
<body>
  <h1>Exemplo de JavaScript no Corpo da Página</h1>

  <div id="javascript-msg"></div>

  <script>
    // Inserindo JavaScript diretamente no corpo da página
    alert("Oi rodando JavaScript no corpo da página!!");
    // Atualizando uma div com uma mensagem
    document.getElementById("javascript-msg").textContent =
"JavaScript executado no corpo da página!";
  </script>
</body>
</html>
```

JavaScript no **Corpo da Página**

Resultado!



Javascript

- Neste caso, quando o navegador do usuário carregar este trecho de código aparecerá uma janela escrita “**Oi rodando JavaScript no corpo da página!!!”**
- Deve-se evitar escrever o código diretamente no corpo da página pois ele não poderá ser reutilizado em outras páginas (assim como o código CSS já visto).

JavaScript em **Arquivo Externo**

- É a maneira preferida pela maioria dos desenvolvedores.
- Consiste em criar um novo arquivo com a extensão ‘.js’ e lá escrever o código.
  - Assim, como nas folhas de estilo, pode-se reutilizar o código em diversas páginas sem precisar reescrevê-lo, devendo apenas referenciar o arquivo.

```
<script src=“javascript.js">
</script>
```

JavaScript em **Arquivo Externo – Ex2**

Index.html:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Exemplo de JavaScript com Arquivo Externo</title>
 <style>
  body {
   font-family: Arial, sans-serif;
   background-color: #f0f0f0;
   text-align: center;
   padding-top: 50px;
  }
  #javascript-msg {
   font-size: 24px;
   color: #333;
  }
 </style>
</head>
<body>
 <h1>Exemplo de JavaScript com Arquivo Externo</h1>

 <div id="javascript-msg"></div>

 <!-- Incluindo o arquivo JavaScript externo -->
 <script src="javascript.js"></script>
</body>
</html>
```

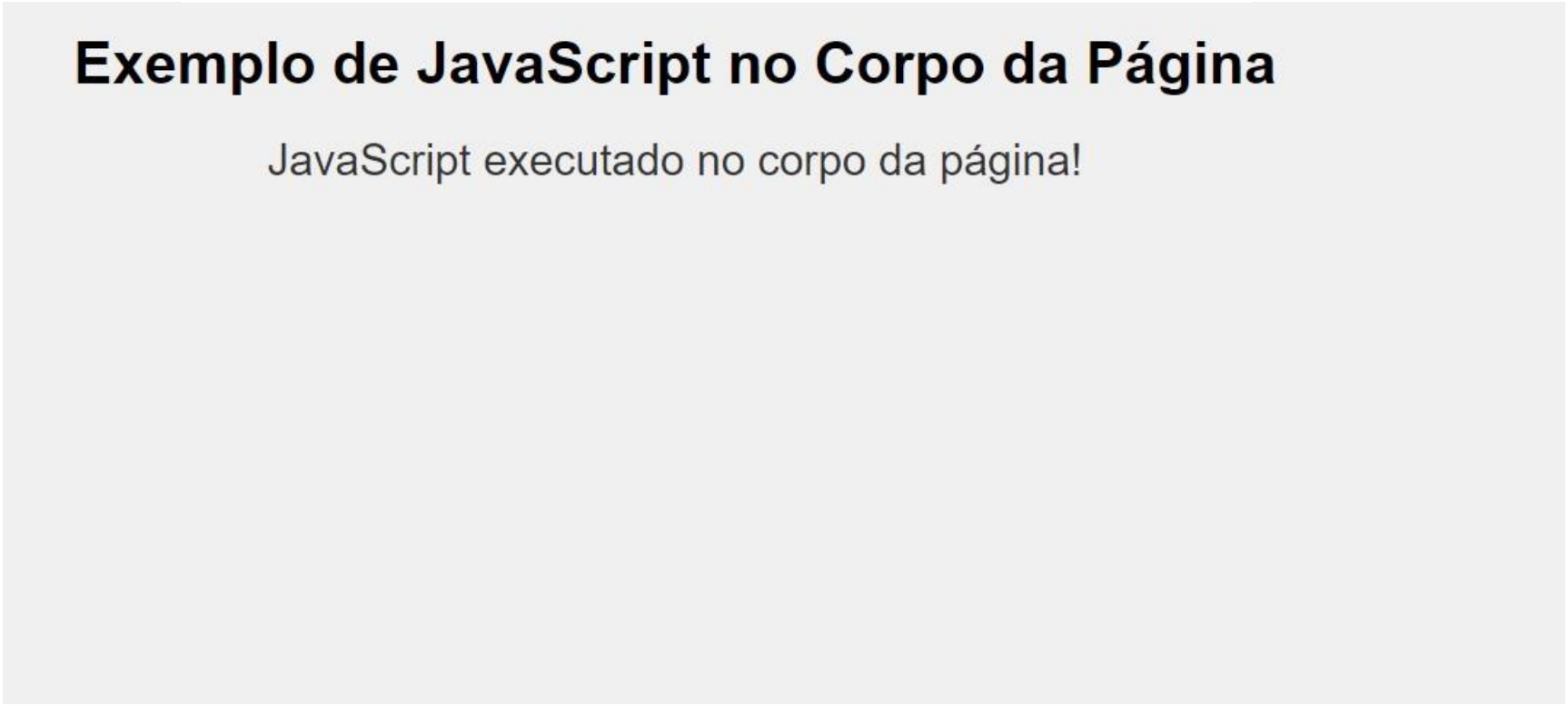
JavaScript em Arquivo Externo – Ex2

Javascript.js

```
// Código JavaScript externo
alert("Oi rodando JavaScript em um arquivo externo!!");

// Atualizando uma div com uma mensagem
document.getElementById("javascript-msg").textContent = "JavaScript executado a partir
de um arquivo externo!";
```

JavaScript em **Arquivo Externo – Ex2**

Resultado é o mesmo!



JavaScript em **Atributo de Eventos – Ex3**

Index.html:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Exemplo de JavaScript com Atributos de Eventos</title>
 <style>
  body {
   font-family: Arial, sans-serif;
   background-color: #f0f0f0;
   text-align: center;
   padding-top: 50px;
  }
  #javascript-msg {
   font-size: 24px;
   color: #333;
  }
  button {
   padding: 10px 20px;
   font-size: 16px;
   background-color: #007bff;
   color: #fff;
   border: none;
   cursor: pointer;
  }
 </style>
</head>
<body>
 <h1>Exemplo de JavaScript com Atributos de Eventos</h1>

 <button onclick="mostrarMensagem()">Clique Aqui</button>

 <div id="javascript-msg"></div>

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

JavaScript em **Atributo de Eventos – Ex3**

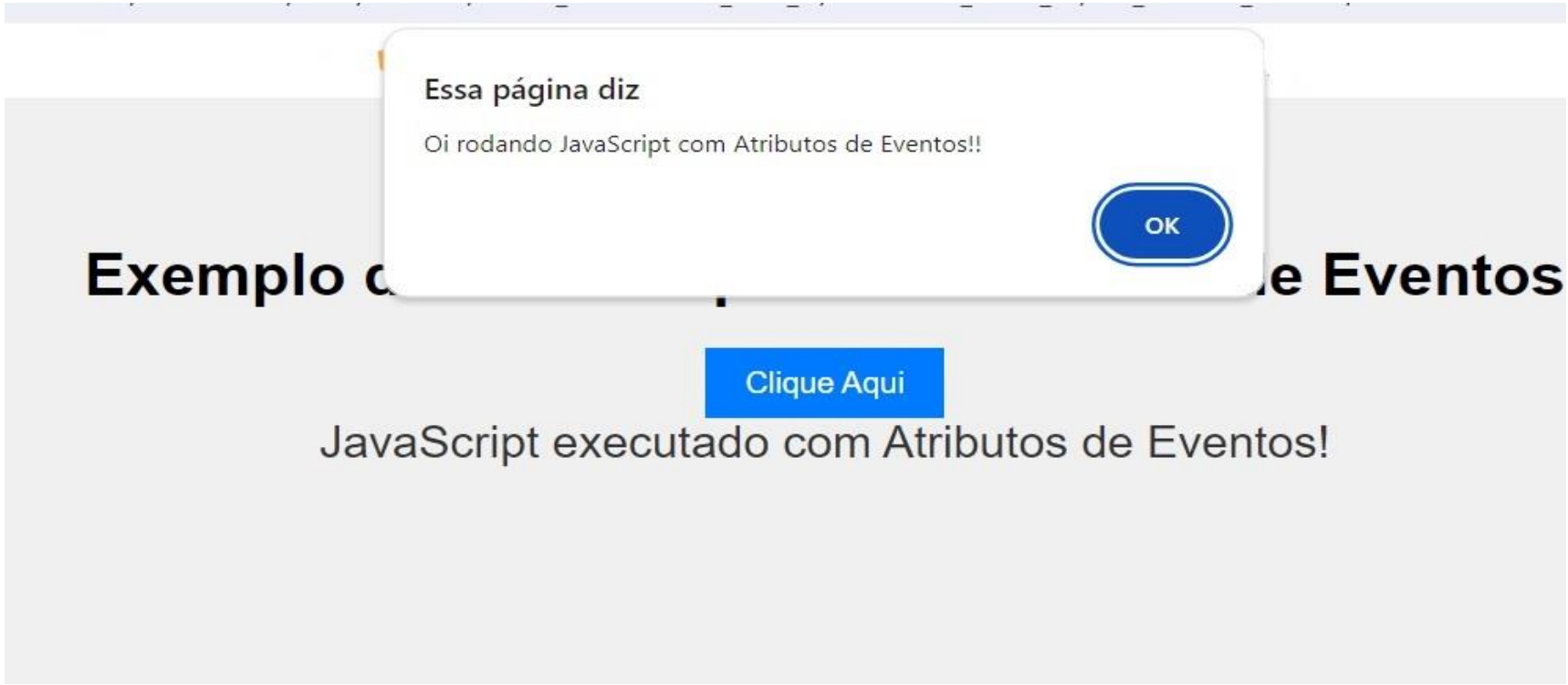
javascript.js:

```
function mostrarMensagem() {
  // Código para mostrar uma mensagem
  alert("Oi rodando JavaScript com Atributos de Eventos!!");

  // Atualizando uma div com uma mensagem
  document.getElementById("javascript-msg").textContent = "JavaScript
executado com Atributos de Eventos!";
}
```

JavaScript em **Atributo de Eventos – Ex3**

Resultado 1

Resultado 2 (clicando no botão)



Variáveis

- **Declarar variáveis:** algumas linguagens de programação exigem que as variáveis sejam declaradas no começo do código, já o JavaScript não exige isso. O nome da variável é *case sensitive.*

```
var teste = "teste";

teste = "teste";
```

- JavaScript é uma linguagem dinamicamente tipada, o que significa que não precisamos especificar o tipo de dados ao declarar uma variável.
- Os tipos de dados em JavaScript incluem números, strings, booleanos, arrays, objetos, undefined, null e outros.

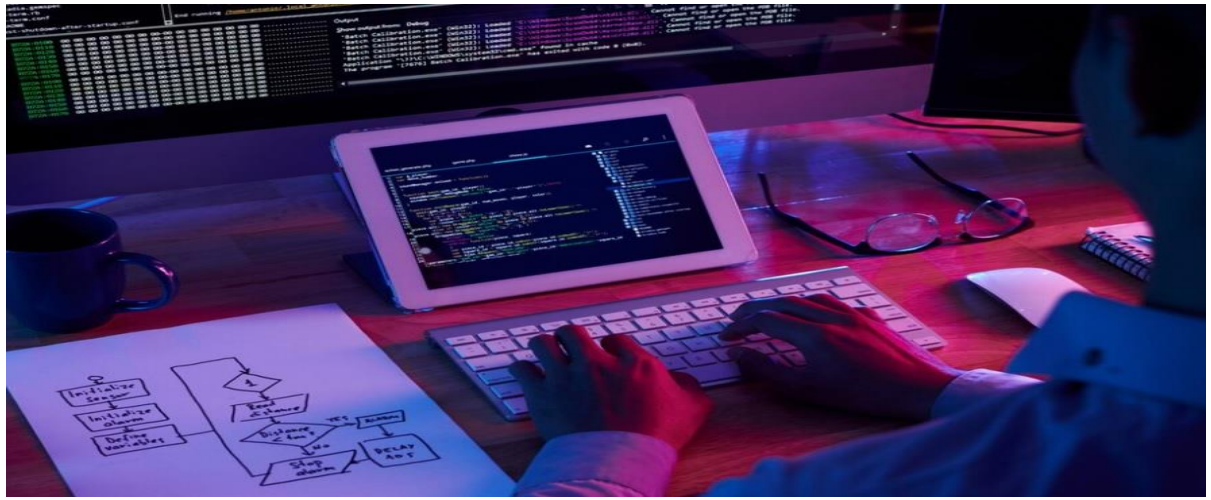
Variáveis

- Em JavaScript, podemos declarar variáveis usando as palavras-chave var, let e const.
- ‘var’ era a única forma de declarar variáveis antes do ECMAScript 6 (ES6), especificação padrão de JavaScript.
- ‘let’ e ‘const’ foram introduzidos no ES6 para oferecer formas mais seguras e flexíveis de declarar variáveis.

  - Use var para variáveis com escopo de função.
  - Use let para variáveis que precisam de escopo de bloco e podem ser reatribuídas.
  - Use const para constantes que não serão reatribuídas, e apresentam escopo de bloco.

Variáveis

- Existem duas maneiras de se declarar uma variável em JavaScript, com e sem o atributo var:
  - usando ‘var’ a variável é criada no escopo da função (sendo liberada quando a função encerra).
  - sem ‘var’, a variável é criada no escopo global que no browser é o objeto *window* (sendo liberada quando a página é fechada) - não é boa prática de programação!



Variáveis

- Uma variável pode assumir diversos valores primitivos
  - Number - 10, 10.5, 1.5e3, 0xff (representado 255)
  - String - “teste”, ‘teste’
  - Boolean - true ou false
  - undefined - quando não existe ou não foi inicializada
  - null - permite definir explicitamente que não tem valor
  - valores não primitivos (Object, Function, Array)

Variáveis

```
var n1 = 10;          // Number
var n2 = 10.5;
var n3 = 1.5e3;
var n4 = 0xff;

var s1 = "Teste";     // String
var s2 = 'Teste';

var b1 = true;        // Boolean

var u = undefined;
var n = null;

var obj = {    // Object
  id: 30,
  nome: "Fulano"
};
var arr = [ "lista", "de", "elementos" ]; //Array
var func = function(x, y) { return x + y }; // Function
```

Condicionais

```
var numero = 1;
// Comparação normal (compara apenas os valores)
if(numero == "1") { // resultado: true
  alert("== compara apenas os valores");
}

// Comparação restrita (compara valores e tipos das variáveis)
var mensagem;
if(numero === "1") { // resultado: false
  mensagem = "Nunca executa";
} else {
  mensagem = "Number é diferente de String com o ===";
}

/*
  Operador ternário (versão reduzida do IF)
  condição ? se_verdadeiro : se_falso ;
*/
var x = (numero === "1") ? "Nunca executa" : "Number é diferente de String com o ===" ;
```

Condicionais – Ex4

Index.html:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemplo de Variáveis Condicionais em JavaScript</title>
</head>
<body>
  <h1>Exemplo de Variáveis Condicionais em JavaScript</h1>

  <script>
    // Declaração da variável
    var numero = 1;

    // Verificação da condição utilizando ==
    if(numero == '1') {
      alert("== compara apenas os valores");
    }
    // Declaração da variável mensagem
    var mensagem;

    // Verificação da condição utilizando ===, compara valores e tipos
    if(numero === "1") {
      mensagem = "Nunca executa";
    } else {
      mensagem = "Número é diferente de String com o ===";
    }

    // Utilização do operador ternário para atribuir valor à variável x
    var x = (numero === "1") ? "Nunca executa" : "Número é diferente de String com o ===";

    // Exibição dos resultados na página
    document.write("<p>Resultado da verificação com '===' utilizando if...else: " + mensagem + "</p>");
    document.write("<p>Resultado da verificação com '===' utilizando operador ternário: " + x + "</p>");
  </script>
</body>
</html>
```

Condicionais

Resultado:

Exemplo de Variáveis Condicionais em JavaScript

Resultado da verificação com '===' utilizando if...else: Número é diferente de String com o ===

Resultado da verificação com '===' utilizando operador ternário: Número é diferente de String com o ===

```
// Declaração da variável mensagem
                /*==: Este operador compara apenas os valores, realizando uma conversão de tipo, se necessário.
                Isso significa que ele tenta igualar os dois valores, mesmo que sejam de tipos diferentes.
                Por exemplo, no código, a expressão numero == '1' retorna true porque o valor numérico 1 é
                igual à string '1' após a conversão de tipo.

  ===: Este operador, conhecido como estritamente igual, compara tanto os valores quanto os tipos.
                Ele não realiza nenhuma conversão de tipo, então, para retornar true, os dois operandos devem
                ser do mesmo tipo e ter o mesmo valor. No código, a expressão numero === '1' retorna false
                porque numero é um número (1) e a string '1' é de um tipo diferente.
  */
```

Condicionais SwitchCase – Ex5

Index.html:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Exemplo de Switch em JavaScript</title>
</head>
<body>
 <h1>Exemplo de Switch em JavaScript</h1>

 <label for="laranjaInput">Digite a quantidade de laranjas (1,2 ou 3):</label>
 <input type="text" id="laranjaInput">
 <button onclick="calcularSuco()">Calcular</button>
<script>
    function calcularSuco() {
      var laranjaInput =
document.getElementById("laranjaInput").value;
      var laranja = parseInt(laranjaInput);

      if (isNaN(laranja) || laranja <= 0) {
        alert("Digite um número válido de laranjas.");
        return;
      }
```

```
  // Estrutura de controle switch
    switch(laranja) {
      case 1:
        alert("Fazer um copo de suco");
        break;
      case 2:
        alert("Fazer dois copos de suco");
        break;
      case 3:
        alert("Fazer uma jarra de suco");
        break;
      default:
        alert("Quantidade de laranjas não suportada");
    }
  }
 </script>
</body>
</html>
```

OBS: isNaN (NaN-Not a Number)

Condicionais SwitchCase – Ex5

Antes de clicar 'calcular'

Depois de clicar 'calcular'

Laços de repetição

a)

```
// FOR - executa um laço um determinado numero de vezes
var pessoas = ["João", "José", "Maria", "Sebastião", "Antônio"];

for (var i = 0; i < pessoas.length; i++) {
  alert(pessoas[i]);
}
```

b)

```
// FOR IN - executa um laço nas propriedade de um objeto
var nome = "";
var pessoa = { nome:"Iara", sobrenome:"Almeida"};
for (x in pessoa) {
  nome += pessoa[x] + " ";
}
alert(nome);
```

Laços de repetição (Laço for) – Ex6

Index.html:

a)

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Exemplo de Laço For em JavaScript</title>
</head>
<body>
 <h1>Exemplo de Laço For em JavaScript</h1>

 <button onclick="exibirNomes()">Exibir Nomes</button>

 <script>
  function exibirNomes() {
   var pessoas = ["João", "Ana", "Maria", "Sebastião", "Fernanda"];

   // Laço for para percorrer o array de pessoas
   for (var i = 0; i < pessoas.length; i++) {
    alert(pessoas[i]); // Exibe o nome atual em um alerta
   }
  }
 </script>
</body>
</html>
```

Laços de repetição (Laço for) – Ex6

Resultado antes de clicar no botão:

a)

Vai mostrar todos os nomes a cada clique no botão, irá mudar o nome

Laços de repetição (Laço for) – Ex6

Index.html:

b)

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemplo de Laço For...in em JavaScript</title>
</head>
<body>
  <h1>Exemplo de Laço For...in em JavaScript</h1>
  <button onclick="exibirPessoa()">Exibir Pessoa</button>
  <script>
    function exibirPessoa() {
      var pessoa = {nome: "Iara", sobrenome: "Almeida"};
      var nomeCompleto = "";

      // Laço for...in para percorrer as propriedades do objeto
      for (var x in pessoa) {
        nomeCompleto += pessoa[x] + " ";
      }
      alert(nomeCompleto); // Exibe "Iara Almeida"
    }
  </script>
</body>
</html>
```

Laços de repetição (Laço for) – Ex6

Resultado antes de clicar no botão:

b)

Vai mostrar todos os nomes a cada clique no botão, irá mudar o nome

Laços de repetição – Exercícios em aula

Laco_repetição 01

```
var resultado="";
var numero = 1;
while (numero<=10){
    resultado += numero + " ";
    numero++;
}
alert(resultado);
```

Implementar esses exemplos para entregar.
Exemplo Laco_repetição 01 e 02

Laco_repetição 02

```
var resultado="";
var numero = 1;
do{
    resultado += numero + " ";
    numero++;
}while (numero<=10);
alert(resultado);
```

Funções - Exercícios em aula

- É um recurso que permite efetuar tarefas repetitivas ou funcionalidades que só devem ser executadas após um evento, como o clique de um botão, o uso da função é a solução.

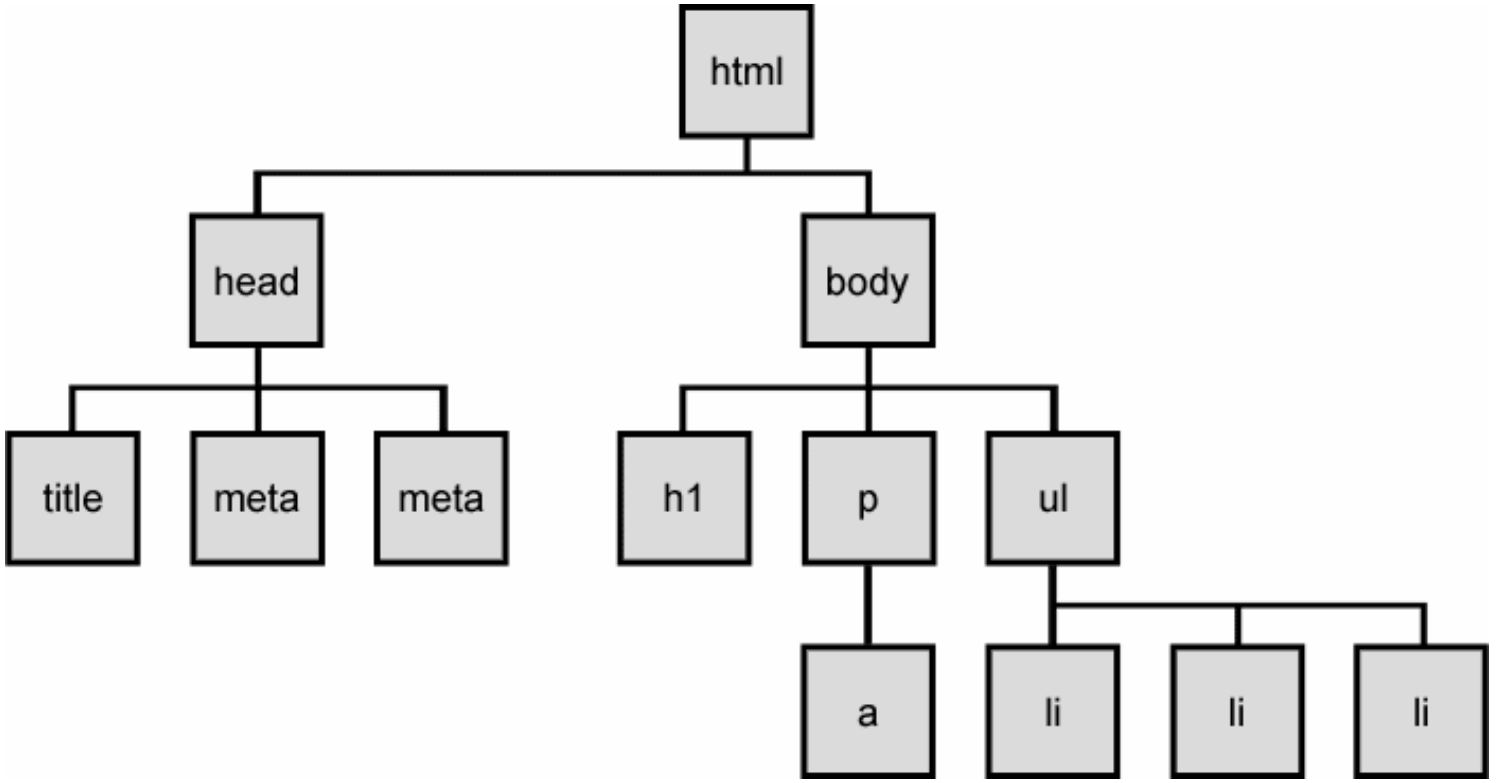
- Exemplo_funcoes: “Executar função ao clique do botão”: executar um alerta cada vez que um botão seja acionado até, no máximo, 5 alertas. A cada iteração, apresentar o número do alerta.

Tarefa de aula, implementar este exemplo:
- Exemplo_funcoes

DOM

- Quando uma página HTML é carregada, o browser cria o DOM da página.
- Ou seja, o HTML resultante, o que pode ser visto pressionando Ctrl + U nos browsers!

DOM

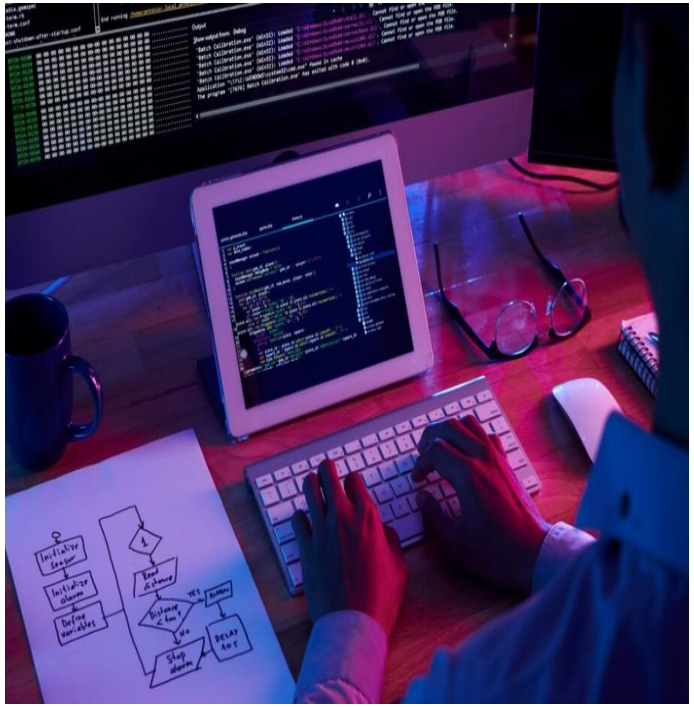


Alteração do DOM

- Após a pagina carregada no cliente, existe a possibilidade de alterar o DOM sem efetuar requisições ao servidor.
- Para isso, utilizamos Javascript e/ou bibliotecas javascripts, tais como jQuery.
- Javascript é executado nativamente pelos browsers.

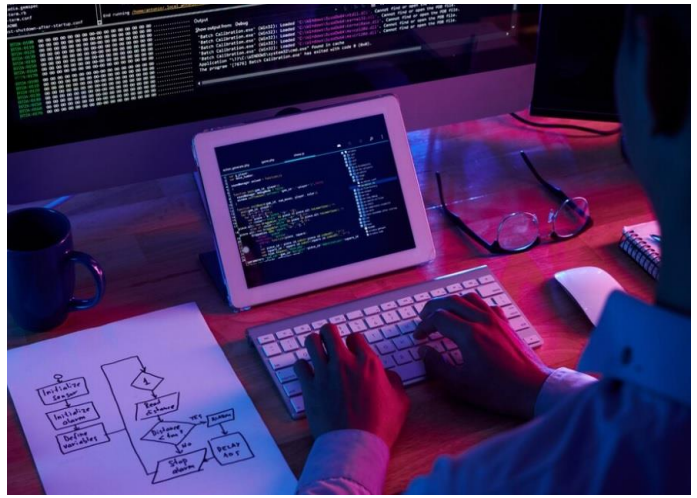
Eventos

onclick	Evento disparado quando se clica com o mouse em algum elemento
oncontextmenu	Clique com o direito do mouse
ondblclick	Dois cliques do mouse em um elemento
onmousedown	Ocorre quando o usuário pressiona o mouse (antes do clique)
onmouseenter	Executa quando o mouse entra na área do elemento.
onmouseleave	Executa quando o mouse sai da área do elemento.
onmousemove	Executado enquanto o usuário esta movendo o mouse em cima de um elemento.
onmouseover	Executado quando o mouse entra na área de um elemento ou dos seus filhos.
onmouseout	Executado quando o mouse sai da área de um elemento ou de seus filhos.
onmouseup	Executado quando o botão do mouse para de ser pressionado.



Eventos

onkeydown	Executado quando o usuário esta pressionando uma Tecla.
onkeypress	Executado quando o usuário pressiona uma Tecla.
onkeyup	Executado quando o usuário solta uma Tecla.



Eventos

onabort	Executa quando o usuário aborta o carregamento de algum elemento ou recurso.
onbeforeunload	Executado antes do documento html ser descarregado
onerror	Executado quando ocorre erro no carregar um arquivo externo.
onpagehide	Executado quando o usuário “sai” do site. (troca de aba)
onscroll	No ato do usuário utilizar a barra de rolagem.

Listagem completa de eventos

- http://www.w3schools.com/jsref/dom_obj_event.asp

Seletores

- Para efetuar uma alteração com javaScript, antes precisamos identificar o elemento desejado.

Seletores

- Exemplos:
  - document.getElementsByTagName("div")
  - document.querySelectorAll(".my-class")
  - document.getElementsByClassName("my-class")
  - document.getElementById(“meuID”);

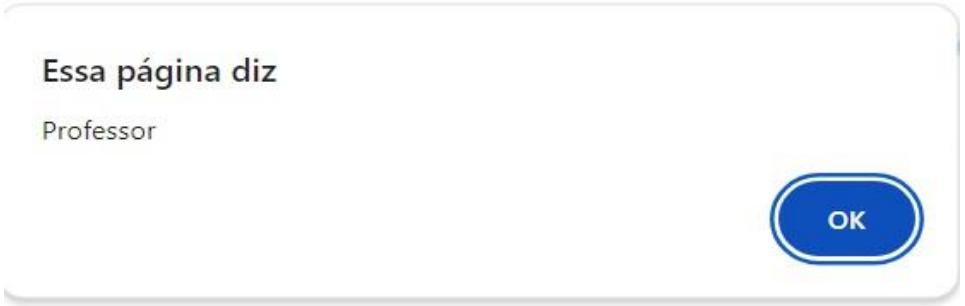
Exemplo 1 - Seletores

Index.html:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<title>Exemplo de onfocusout</title>
<style>
 body {
  font-family: Arial, sans-serif;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
 }
 #nome {
  padding: 10px;
  font-size: 16px;
  border: 1px solid #ccc;
  border-radius: 5px;
  margin-right: 10px;
 }
</style>
</head>
<body>
 <label for="nome">Informe o seu nome:</label>
 <input type="text" id="nome" onfocusout="minhaFuncao()">

 <script>
  function minhaFuncao() {
   var x = document.getElementById("nome");
   alert(x.value);
  }
 </script>
</body>
</html>
```

Exemplo 1 - Seletores

Resultado ao clicar fora da caixa após preencher nome:



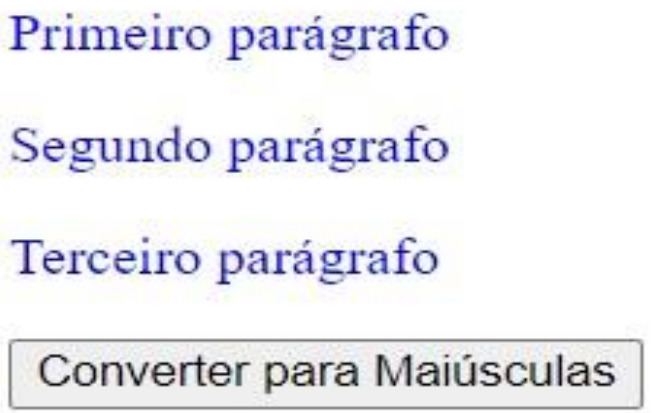
Exemplo 2 - Seletores

Index.html (deixa o que está em <p> maiúscula):

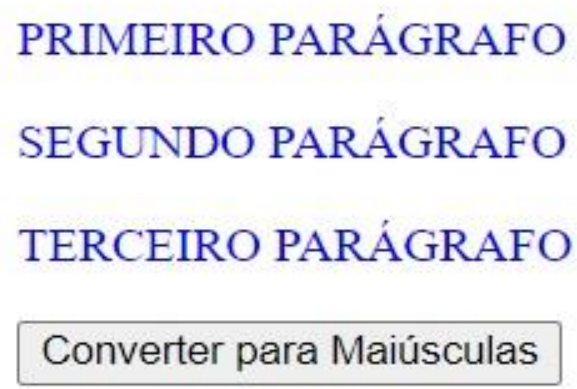
```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-
width, initial-scale=1.0">
<title>Exemplo de
getElementsByClassName</title>
<style>
  .my-class {
    color: blue;
  }
</style>
</head>
<body>

<p class="my-class">Primeiro parágrafo</p>
<p class="my-class">Segundo parágrafo</p>
<p class="my-class">Terceiro parágrafo</p>

<button
onclick="converterParaMaiusculas()">Converter
para Maiúsculas</button>

<script>
function converterParaMaiusculas() {
  var elementos =
document.getElementsByClassName("my-class");
  for (var i = 0; i < elementos.length; i++) {
    elementos[i].textContent =
elementos[i].textContent.toUpperCase();
  }
}
</script>

</body>
</html>
```

Exemplo 2 - Seletores

Resultado (antes):



Resultado (depois de clicar no botão):



Exemplo 2 - Seletores

Agora faça a função converterParaMinusculas() para converter para minúscula, aproveitando o mesmo código que já existe para a Maiúscula. Utilize dois botões para ficar alternando entre os modos

Resultado (antes):

Resultado (depois de clicar no botão):

EXERCÍCIOS PARA ENTREGAR:

- Além dos exercícios feitos em aula, entregar os exercícios da lista que esta no **arquivo PDF anexo a aula de hoje**.
- Para todos os exercícios utilizar das boas práticas de programação e colocar o código CSS (styles.css) e o de javaScript (nome.js) separado do arquivo HTML.

- **Colocar todos os códigos nas suas respectivas pastas e compactar num arquivo (zip ou rar) e submeter no 'Minha UFN' no exercício da aula.**

Recomendação de leitura

- http://tableless.com.br/tenha-o-dom/
- http://www.fititnt.org/padrao/seletores.html
- http://desenvolvimentoparaweb.com/javascript/javascript-jquery-equivalentes-funcoes-nativos/
- https://www.caelum.com.br/apostila-html-css-javascript/jquery/#12-7-exercicios-jquery-na-home

Referências

HOGAN, Brian P. HTML 5 e CSS3: Desenvolva hoje com o padrão de amanhã . Rio de Janeiro (RJ): Ciência Moderna Ltda., 2012 282 p. ISBN 978-85-399-0260-6

Fábio Flatschart. HTML 5 - Embarque Imediato, 2011. (Biblioteca Digital)

Deitel, Paul J.; Deitel, Harvey M.. Ajax, Rich Internet Applications e Desenvolvimento Web para Programadores, 2008. (Biblioteca Digital)

SILVA, Maurício Samy. HTML 5: a linguagem de marcação que revolucionou a web. 2. ed. São Paulo, SP: Novatec, 2014. 335 p. ISBN 978-85-7522-403-8.

Denilson Bonatti. Desenvolvimento de Jogos em HTML5, 2014. (Biblioteca Digital)

W3SCHOOL. The world’s largest web development site. Acessado em 2018. Disponível em: http://www.w3schools.com/.

W3C. World Wide Web Consortium. Acessado em 2018. Disponível em: http://www.w3.org

MORRISON, Michael. Use a cabeça JavaScript. Rio de Janeiro (RJ): Alta Books, 2008. 606 p.

https://br.freepik.com/fotos-vetores-gratis/desenvolvimento-web

Material do Professor Fabrício Tonetto Londero, 2023.

Thank you for your attention!!

Email: **andre.flores@ufn.edu.br**

Santa Maria – RS 2025