



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

AULA:

Introdução à Java Script

Disciplina: Programação de Sistemas para Internet

Alba Lopes, Profa.

<http://docentes.ifrn.edu.br/albalopes>
alba.lopes@ifrn.edu.br

Agenda

- ▶ Introdução
- ▶ Primeiro Exemplo
- ▶ Segundo Exemplo
- ▶ Conceitos introdutórios
 - ▶ Código JavaScript em página HTML
 - ▶ Funções e eventos
 - ▶ Scripts Externos
- ▶ Comandos de Saída
 - ▶ window.alert()
 - ▶ document.write()
 - ▶ innerHTML()
 - ▶ console.log()
- ▶ Sintaxe do JavaScript
 - ▶ Variáveis
 - ▶ Atribuição
 - ▶ Exemplos
- ▶ Exercícios



Introdução

- ▶ O Java Script permite detectar qualquer coisa que ocorre em uma página web como:
 - ▶ Clique na página
 - ▶ Redimensionamento da janela do navegador
 - ▶ Fornecimento de dados em um campo de texto
 - ▶ Foco em um elemento
- ▶ Sendo assim, Java Script é uma linguagem de programação de script que identifica eventos ocorridos em uma página web.
- ▶ É possível escrever código para responder a essas interações do usuário, como por exemplo
 - ▶ Mudar o conteúdo um elemento
 - ▶ Alterar o estilo dos elementos
 - ▶ Validar dados
 - ▶ Exibir e ocultar elementos
 - ▶ etc



Primeiro exemplo (parte 1/2)

▶ JavaScript pode modificar o conteúdo de um element HTML

- ▶ Um dos muitos métodos existente (e largamente utilizado) em Javascript é o `getElementById()`.
- ▶ Esse método busca um elemento com determinado id e realiza alguma operação sobre ele.
- ▶ Seja uma página HTML com a seguinte estrutura

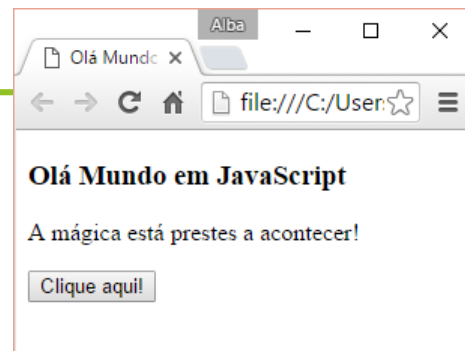
```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"/></head>
<body>

  <h3>Olá Mundo em JavaScript</h3>

  <p id="olamundo">A magia está prestes a acontecer!</p>

  <button type="button">Clique aqui!</button>

</body>
</html>
```



Primeiro exemplo (parte 2/2)

- ▶ Para modificar o do parágrafo de id “olamundo”, deve-se buscar o parágrafo com o método `getElementById()` e alterar o seu conteúdo HTML.

```
document.getElementById('olamundo').innerHTML = 'Olá Mundo!';
```

- ▶ Essa ação deve ser realizada quando o botão for clicado (onclick):

```
<body>

<h1>Olá Mundo em JavaScript</h1>

<p id="olamundo">A mágica está prestes a acontecer!</p>

<button type="button" onclick="document.getElementById('olamundo').innerHTML = 'Olá Mundo!'">
Clique aqui!</button>

</body>
```



Segundo exemplo (parte 1/1)

► Alterar o estilo de um element

- O JavaScript consegue alterar as propriedades de estio (CSS) dos elementos

```
document.getElementById('olamundo').style.backgroundColor = 'yellow';
```

- Assim como no exemplo anterior, a ação será realizada ao clicar no botão:

```
<body>

  <h1>Olá Mundo em JavaScript</h1>

  <p id="olamundo">A mágica está prestes a acontecer!</p>

  <button type="button" onclick="document.getElementById('olamundo').style.backgroundColor = 'yellow'">
    Clique aqui!</button>

</body>
```



Conceitos introdutórios

▶ Código JavaScript em página HTML

- ▶ Existem algumas formas de se inserir código JavaScript em uma página HTML
- ▶ A utilização das tags `<script></script>` dentro do body é uma delas
- ▶ O atributo `type="text/javascript"` indica ao browser que o código contido dentro das tags é do tipo “JavaScript”, porém esse atributo pode ser omitido porque JS é a linguagem de script padrão de HTML.

```
<body>
  <h3> Olá Mundo em JavaScript </h3>

  <p id="olamundo">A magia está prestes a acontecer!</p>

  <button type="button" onclick="document.getElementById('olamundo').style.backgroundColor = 'yellow'">
  Clique aqui!</button>

  <script>
    document.getElementById('olamundo').innerHTML = 'A magia aconteceu!';
  </script>
</body>
```



Conceitos introdutórios

► Funções e eventos

- Uma função JavaScript é um bloco de código que pode ser executado quando solicitado
- Por exemplo, uma função é executada quando um evento ocorre (como o clique de um botão)
- As funções podem ser declaradas tanto no <head>, quanto no <body>
 - Funções e eventos serão abordados ainda em maiores detalhes

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8"/>
  <script>
    function minhaFuncao() {
      document.getElementById('p1').innerHTML = 'Parágrafo modificado!';
    }
  </script>
</head>
<body>
  <h1>Minha página web</h1>
  <p id="p1">Um parágrafo</p>
  <button type="button" onclick="minhaFuncao()">Teste sua função</button>
</body>
</html>
```



Conceitos introdutórios

▶ Scripts externos

- ▶ Scripts também podem ser colocados em arquivos externos, com extensão .js
- ▶ Arquivo externos contendo scripts são úteis quando o mesmo trecho de código é invocado em diversas páginas html

```
function minhaFuncao() {  
    document.getElementById('p1').innerHTML = 'Parágrafo modificado!';  
}
```

meuScript.js

```
<!DOCTYPE html>  
<html>  
<head><meta charset="utf-8"/></head>  
<body>  
<h1>Minha página web</h1>  
<p id="olamundo">Um parágrafo</p>  
    <button type="button" onclick="minhaFuncao()">Teste sua função</button>  
    <script src="meuScript.js"></script>  
</body>  
</html>
```

meuArquivo.html



Conceitos introdutórios

▶ Scripts externos

▶ Vantagens

- ▶ Separa HTML e Javascript
- ▶ Faz o código tanto HTML quanto JavaScript mais fácil de ler e manter
- ▶ Código JavaScript que já tenham sido carregados em cache aumentam a velocidade do carregamento da página



Scripts externos **NÃO** podem conter a tag `<script>`



É uma boa prática colocar scripts imediatamente antes da tag de encerramento do body (`</body>`)
Melhora o carregamento da página, pois muito código javascript pode acarretar lentidão na exibição



Comandos de Saída

- ▶ JavaScript pode exibir dados de diferentes formas
 - ▶ Escrevendo em uma caixa de alerta, usando `window.alert()`.
 - ▶ Escrevendo no próprio HTML, usando `document.write()`.
 - ▶ Escrevendo em um element HTML, usando `innerHTML`.
 - ▶ Escrevendo no console do navegador, usando `console.log()`.

▶ `window.alert()`

```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"/></head>
<body>
  <h3>Minha página web</h3>
  <p>Exemplo de saída com alert</p>
  <script>window.alert('Mundo Mágico do JavaScript');</script>
</body>
</html>
```



Comandos de Saída

▶ document.write()

```
<!DOCTYPE html>
<html>
  <head><meta charset="utf-8"/></head>
  <body>
    <h3>Minha página web</h3>
    <p>Exemplo de saída com document.write.</p>
    <button onclick="document.write('Mundo mágico do JavaScript')">Teste seu código!</button>
  </body>
</html>
```



Usar `document.write()` após o carregamento completo do HTML irá apagar todo o HTML existente.



Comandos de Saída

▶ innerHTML()

- ▶ Para acessar um element HTML, JavaScript pode usar o método `document.getElementById(id)`.
- ▶ O atributo `id` define o element HTML. A propriedade `innerHTML` define o conteúdo HTML :

```
<!DOCTYPE html>
<html>
  <head><meta charset="utf-8"/></head>
  <body>
    <h3>Minha página web</h3>
    <p id="p1">Exemplo de saída com innerHTML.</p>
    <button onclick="document.getElementById('p1').innerHTML = 'Mundo mágico do JavaScript'">
      Teste seu código!</button>
  </body>
</html>
```

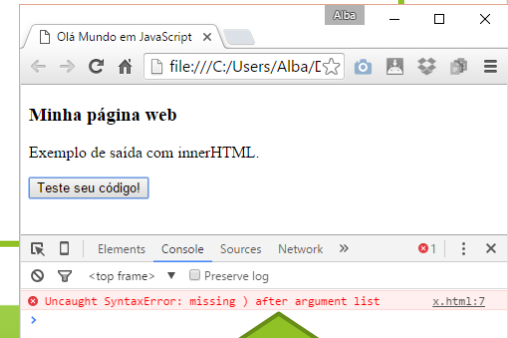


Comandos de Saída

▶ console.log()

- ▶ No seu navegador, você pode usar o método `console.log()` para exibir dados.
- ▶ Ative o console do desenvolvedor no seu navegador com F12 e selecione "Console" no menu.

```
<!DOCTYPE html>
<html>
  <head><meta charset="utf-8"/></head>
  <body>
    <h3>Minha página web</h3>
    <p id="p1">Exemplo de saída com console.log</p>
    <button onclick="console.log('Mundo mágico do JavaScript')">
      Teste seu código!</button>
  </body>
</html>
```



O console do desenvolvedor é uma ferramenta muito útil. Apresenta, inclusive, erros ocorridos na interpretação do código. Teste, por exemplo, retirar o parêntese final do seu código.
Ex: `<button onclick="console.log('Mundo mágico do JavaScript')">`

Sintaxe de JavaScript

▶ Variáveis

- ▶ Variáveis são espaços reservados da memória RAM que guardam o dado recebido
- ▶ O nome de uma variável é utilizado para sua identificação para posterior uso no programa
- ▶ Regras para nomes de variáveis
 - ▶ O nome de uma variável pode ter um ou mais caracteres
 - ▶ O primeiro caractere deve ser uma letra ou o underscore “_”.
 - ▶ Não é permitido utilizar espaços em branco
 - ▶ **CERTO**: n, nome, n1, nome1, _nome1, nome_1
 - ▶ **ERRADO**: 1nome, 1, 1 nome
 - ▶ Para declaração de uma variável, é utilizada a palavra reservada **var**
 - ▶ Ex: var nome = “Maria”.
 - ▶ Entretanto, não é obrigatório o uso da palavra reservada var antes da declaração.



A linguagem JavaScript é sensível ao caso (case sensitive). Faz diferença escrever NOME, nome, Nome, nOME. Cada uma dessas representam variáveis diferentes.



Sintaxe de JavaScript

▶ Atribuição

- ▶ Para atribuir um valor a uma variável, utiliza-se o operador =
 - ▶ `var numero_um = 1;`
 - ▶ `var Salario = 545.55;`
 - ▶ `var sobrenome = "Lopes"`
 - ▶ `var vazio = false;`
 - ▶ `var hoje = Date();`
 - ▶ `var soma = numero_um + 32;`
 - ▶ `var nomeCompleto = "Alba" + " " + sobrenome;`
 - ▶ `var dado = document.getElementById("texto").innerHTML;`



Sintaxe de JavaScript

▶ Exemplo de variável e atribuição

```
<!DOCTYPE html>
<html>
  <head><meta charset="utf-8"/></head>
  <body>
    <p id="p1"></p>
    <p id="p2"></p>
    <script>
      var nome = "Alba";
      var sobrenome;
      document.getElementById("p1").innerHTML = nome;
      document.getElementById("p2").innerHTML = sobrenome;
    </script>
  </body>
</html>
```



Utilizar uma variável declarada, porém não inicializada acarretará no valor **undefined**.



Sintaxe de JavaScript

▶ Exemplo de variável e atribuição

```
<!DOCTYPE html>
<html>
  <head><meta charset="utf-8"/></head>
  <body>
    <p id="p1">Alba</p>
    <button onclick="exibirNoAlerta()">Alerta!</button>

    <script>
      function exibirNoAlerta() {
        var nome = document.getElementById("p1").innerHTML;
        window.alert(nome);
      }
    </script>
  </body>
</html>
```



Assim como é possível alterar o conteúdo HTML de um elemento, também é possível recuperar esse conteúdo!



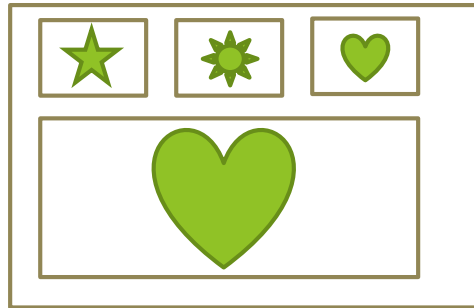
Exercícios

1. Crie uma página HTML com um cabeçalho h1, um botão. Ao clicar no botão, altere o texto do cabeçalho para “A mágica continua!”.
2. Crie uma página HTML com um parágrafo. Ao clicar no parágrafo, altere a cor do texto para verde.
3. Crie uma página com um parágrafo e dois botões: um “Ocultar” e um “Mostrar”. Ao clicar no botão Ocultar, modifique a propriedade `style.display` do parágrafo para `none`. Ao clicar no botão “Mostrar”, altere a propriedade `style.display` do parágrafo para `block`.
4. Crie uma página com uma imagem de uma criança (ex: “crianca.jpg”) e um botão “Crescer”. Crie uma função chamada `trocarImagem`. Nessa função, altere a propriedade `src` da imagem para “adulto.jpg”. Chame a função `trocarImagem` no `onclick` do botão Crescer. Escreva ainda um log no console com a mensagem “Imagem trocada com sucesso!”.



Exercícios

5. Crie uma página contendo 3 botões (Pequena, Média e Grande) e uma imagem. Ao clicar no botão Pequena, altere o tamanho da imagem para 100x100px. Ao clicar no botão Média, altere o tamanho da imagem para 300x300px. Ao clicar no botão Grande, altere o tamanho da imagem para 500x500px.
6. Crie uma página contendo 3 miniaturas de imagens e uma div. Ao clicar em cada imagem, a imagem clicada deve aparecer no seu tamanho original na div. Ex:



REFERÊNCIAS

- ▶ [1] W3C School. JavaScript Tutorial. Disponível em: <http://www.w3schools.com/js/>
- ▶ [2] MORISSON, Michael. Java Script - Use a Cabeça. Ed. 2. Rio de Janeiro: Altabooks
- ▶ [3] Manzano, José; Toledo, Suely. Guia de Orientação e Desenvolvimento de Sites - HTML, XHTML, CSS e JavaScript / Jscript. 2a. Edição

