Introdução a Web JavaScript

Prof: Sergio Hermenegildo
FEMASS

JavaScript NÃO é Java

Diferenças chaves em relação ao Java:

- Java é uma linguagem de programação;
- JavaScript é uma linguagem de script;

- Aplicativos Java são executados pela máquina virtual Java;
- Scripts JavaScript são executados pelos *browsers*;

- Java é compilado;
- JavaScript é texto puro;

JavaScript no HTML

Para inserir códigos JavaScript, iremos fazê-lo em uma Tag HTML apropriada dentro da seção <body> insira o trecho:

```
<script>
      document.write("Hello World!");
</script>
```

- Neste caso, o trecho "*escrito*" pelo JavaScript, será incorporado ao HTML apenas em sua construção;

- O JavaScript é orientado a objetos;
  - Primeira Classe: **document**

JavaScript no HTML

Vejamos uma segunda forma de fazer o mesmo. Coloque a página com o título “Hello World!” e modifiquem o código existente por este

```
<script>
document.write("<h2>"+document.title+"</h2>");
</script>
```

Ou seja, podemos utilizar as propriedades em nossa codificação.

JavaScript: propriedades e Métodos

Propriedades de document:

- title – Define ou Retorna o Título da Página;
- URL – Retorna o URL completo da página;

Métodos de Document:

- write() – Escreve texto no documento;
- createElement(element) – Cria um elemento HTML;
- getElementById(id) – Seleciona um elemento pelo id;
- querySelector(seletor) – Seleciona um elemento (#id, .class, body, etc);

Exemplo:

```
document.getElementById("paragrafo");
document.querySelector("body").setAttribute("bgcolor","blue");
```

Entre outras propriedades e métodos que veremos no decorrer do curso.

JavaScript: arquivo externo

Da mesma forma como nos arquivos CSS, podemos deixar funções e comandos JavaScript em arquivos externos:

– Estes arquivos devem ter a extensão .JS

- Para importar:

```
<head>
      <script type="text/Javascript"
            src="meuScript.js"> </script>
</head>
```

JavaScript: Eventos

É possível disparar scripts a partir de diversos tipos de eventos;

- O primeiro que iremos estudar é o de um clique em um botão:
  - Tag: <button>Clique Aqui!</button>
  - Atributos:
- type="button";
- onclick="alert('Bem vindo!')"

```
<button type="button" onclick=
   "alert('Bem vindo!')"> Clique Aqui!
</button>
```

JavaScript: Funções

Neste exemplo veremos a chamada de uma função, que trocará o conteúdo existente por um novo, após um evento.

```
<body>
<p id="paragrafo">Hello!</p>
<script>
        function mFuncao() {
        var x = document.getElementById("paragrafo");
        x.innerHTML="Olá!";
        }
</script>
<button type="button" onclick="mFuncao();">Traduzir</button>
<body>
```

JavaScript: Funções com argumentos

Mas as funções também podem receber argumentos, vejam:

```
<html>
<head>
    <title>Usando o JavaScript passando value</title>
    <script type="text/javascript" src="value.js"></script>
</head>
<body>
    <h1>Qual sua avaliação do curso de SI da Femass?</h1>
    <form name="area">
        <input type="text" id="campo" /> <br/>
        <input type="button" value="ótimo" onclick="escreve(value)" />
        <input type="button" value="super" onclick="escreve(value)" />
        <input type="button" value="estupendo" onclick="escreve(value)" />
    </form>

</body>
</html>
```

JavaScript: Funções com argumentos

Viram que fizemos a chamada a um arquivo .js no nosso HTML.

Quando o botão é clicado, ele passa o value do próprio botão (que no caso é o que está escrito no botão)

Outra diferença, desta vez tratamos como um form, o que nos permitirá utilizar o dot notation.

```
function escreve(valor){
    document.area.campo.value = valor;
    // Faz o mesmo que
    // document.getElementById("campo").value= valor;
}
```

JavaScript: Variáveis

JavaScript é uma linguagem de tipagem dinâmica e fraca:

– Não é necessário declarar o tipo de uma variável;

– Todas as variáveis são objetos (referência);

– Números são todos reais de 64 bits;

– A variável irá “alterar” o seu tipo de dado conforme os valores

forem atribuídos:

• Tipo de dado dinâmico:

```
var x; // x é indefinido
x = 5; // x é um número
x = "John"; // x é uma string
x = true; // x é um valor lógico
x = null; // x é indefinido
```

JavaScript: Operadores

Os comandos básicos do JavaScript tem a mesma grafia do que no C. Portanto ++, --, %, if, switch, for, while, do while, continue, break são antigos conhecidos, o que nos facilita nos exercícios.

Porém, como o JS não é tipado, temos que ter alguma atenção com os operadores

- == 7==‘7’ retorna verdadeiro
- === 7===‘7’ retorna false por conta do tipo
- != 7 != ‘7’ retorna falso
- !== 7 !== ‘7’ retorna verdadeiro

JavaScript: Campo de entrada

Quando formos obter o valor de um campo existente no HTML, deveremos fazer a conversão para o tipo específico, pois ele será sempre tratado como texto.

No caso de um campo numérico, utilizaremos o comando parseInt(campo.value)

JavaScript: Campo de entrada

```
<!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">
    <style type="text/css">
        #campo{border: black solid 2px; font-size: 20px; text-align: center; width: 70px; height: 40px; }
    </style>
    <title>parseInt</title>
</head>
<body>
    <h3> Informe um numero entre 1 e 9:  </h3>

    <form name="area">
        <input type="text" id="campo"/><br><br>
        <input type="button" id="bt" value="Verificar" onclick="gerador()"/>
        <input type="text"  id="resultado" disabled /><br><br>
    </form>
    <script>
        function gerador(){
            var numero=parseInt(document.area.campo.value);
            if(numero<1||numero>9)
                alert("Numero inválido,tente novamente.");
            else
                if(numero%2==0)
                    document.area.resultado.value="PAR";
                else
                    document.area.resultado.value="IMPAR";
        }
    </script>
</body>
</html>
```

JavaScript: Criando Elementos

É possível adicionar novos elementos HTML;

- Qualquer tipo de elemento, definindo qualquer propriedade;
- Tudo através do JavaScript

```
<body id="corpo">
<h1>Adicionar Elementos</h1>
<p>Digite o texto: <input type="text" id="texto"></p>
<p><button onclick="adicionar()">Adicionar</button></p>
<script>
function adicionar() {
var texto = document.getElementById("texto").value;
var para = document.createElement("p");
para.innerHTML = texto;
var corpo = document.getElementById("corpo");
corpo.appendChild(para);
}
</script>
</body>
```

JavaScript: Removendo Elementos

É possível remover elementos HTML;

- Qualquer tipo de elemento, com a condição de que conheçamos também o seu **pai**;
- Tudo através do JavaScript;

```
<body id="corpo">
<h1>Remover Elemento</h1>
<p><button onclick="remover()">Remover</button></p>
<p id="texto">Texto que será removido...</p>
<script>
function remover() {
var pai = document.getElementById("corpo");
var filho = document.getElementById("texto");
pai.removeChild(filho);
}
</script>
</body>
```

JavaScript: Exercício

1. Criem uma página que mude a cor de fundo para azul quando clicar no botão
2. Criem uma página que obtenha seu nome e sobrenome e de um alerta com eles.
3. Criem uma página que mude a cor de um h1 de acordo com o botão escolhido (Azul ou vermelho)
4. Criem uma página que verifica se um campo texto está vazio
5. Criem uma página que dado um número de entrada, gere uma pirâmide de 1 ao número entrado (exemplo: 4)

   1

   22

   333

   4444
6. Criem uma página que contenha dois botões, um irá criar um parágrafo, e o outro, apagar um parágrafo (não necessariamente os mesmos)

JavaScript: Vetores

Para finalizarmos, vejamos como trabalhar com vetores no JavaScript

O vetor irá funcionar como no C, porém, com a declaração utilizando [] ao invés de {}

```
carros = ["fiat", "ford", "citroen", "bmw"];
```

Para inserirmos um item em um vetor existente basta fazermos:

```
carros.push("pegeout");
```

Mas também podemos acessar um item especifico do vetor de forma direta

```
carros[3]
```

JavaScript: Vetores

Nos vetores do JavaScript poderemos aplicar ordenações, buscas, etc de forma direta com os comandos

Ordenação:

```
carros.sort()
```

Reverso:

```
carros.reverse()
```

Última posição no vetor:

```
carros.lastIndexOf("citroen")
```

Tamanho do vetor:

```
carros.length
```

JavaScript: Vetores

```
<body>
    <h2 id="saida"></h2>
    <script>
    carros = ["fiat", "ford", "citroen", "bmw"];
    a = document.getElementById("saida").innerHTML = "Vetor original <br>";
    a = document.getElementById("saida").innerHTML = a + carros + "<br><br>";
    a = document.getElementById("saida").innerHTML = a + "Adicionando a pegeout <br>";
    carros.push("pegeout");
    a = document.getElementById("saida").innerHTML = a + carros + "<br><br>";
    a = document.getElementById("saida").innerHTML = a + "Vetor reverso <br>";
    a = document.getElementById("saida").innerHTML = a + carros.reverse()+ "<br><br>";
    a = document.getElementById("saida").innerHTML = a + "Vetor orndenado <br>";
    a = document.getElementById("saida").innerHTML = a + carros.sort()+ "<br><br>";
    a = document.getElementById("saida").innerHTML = a + "Achando a última posição do citroen
<br>";
    a = document.getElementById("saida").innerHTML = a + carros.lastIndexOf("citroen")+
"<br><br>";
    a = document.getElementById("saida").innerHTML = a + "Verificando quem está no índice 3 do
vetor <br>";
    a = document.getElementById("saida").innerHTML = a + carros[3]+ "<br><br>";
    a = document.getElementById("saida").innerHTML = a + "Usando o length vemos que o vetor
possui "+carros.length+" carros <br>";
    </script>
</body>
```

JavaScript: Exercício

1. Criem uma página que contenha um campo para receber entradas de frutas, um botão para adicionar a uma lista na ordem em que foram digitadas, e outra lista demonstrando as frutas ordenadas ao contrário.