

Aula 4

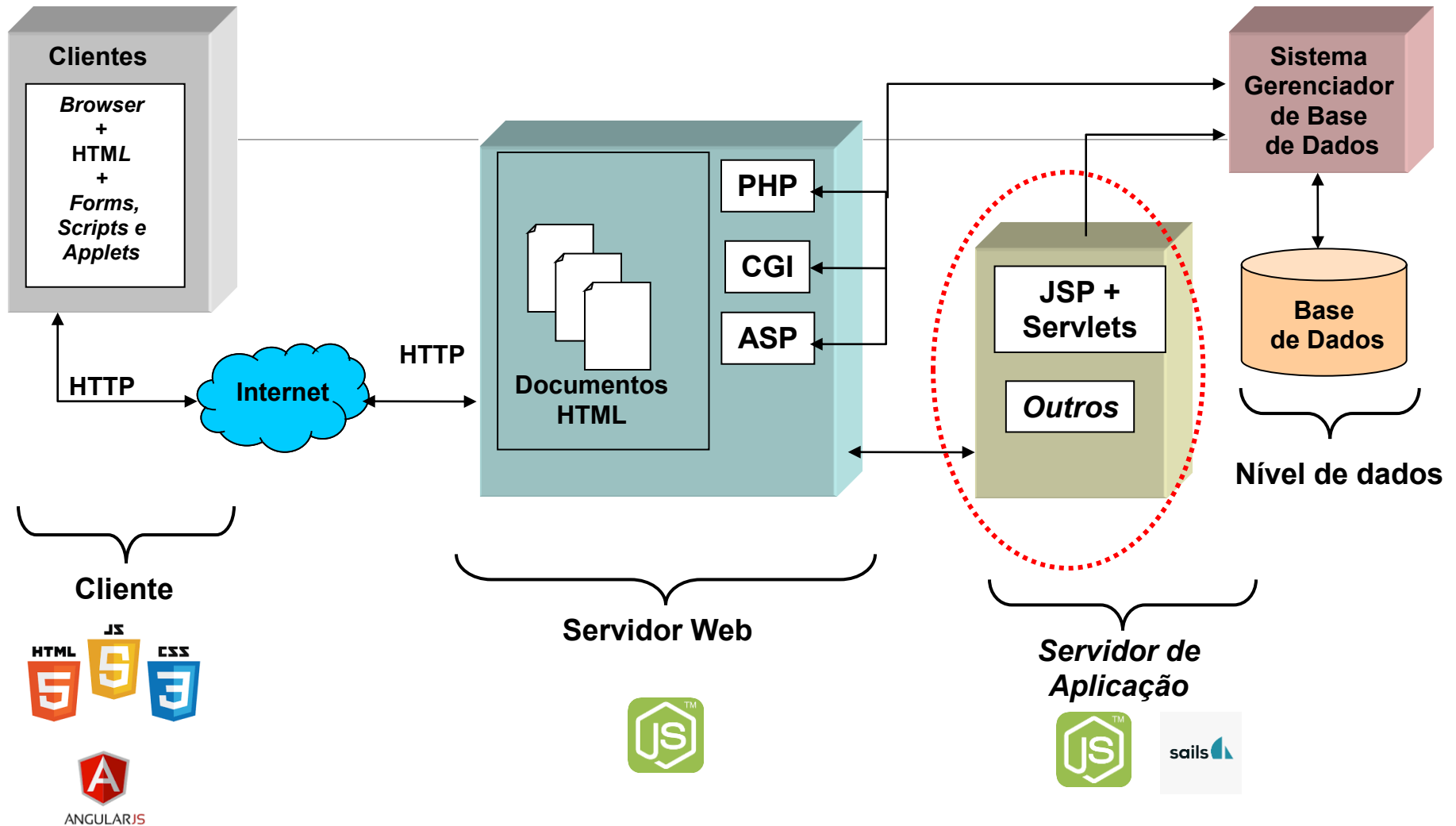
Introdução a JavaScript e DOM

PROF. RUDINEI GOULARTE (RUDINEI@ICMC.USP.BR)

Sumário

- Introdução
- JavaScript
 - Onde colocar o código JavaScript?
 - Funções
 - Eventos, Operadores e Objetos
- Manipulação de Documentos Estruturados
 - Document Object Model

Arquitetura em n-níveis (n-Tier)



Programação no Cliente

- Linguagens de Script
 - Javascript
- Modelos e *Parsers* para Documentos Estruturados
 - DOM & SAX
- Ajax

JavaScript

- JavaScript é uma linguagem de *scripting* multiplataforma e orientada a objetos
 - Adiciona interatividade ao HTML
- Utilizada para design, validar formulários, tratar eventos, detectar navegador, criar cookies, etc
- Linguagem interpretada
- Não é a mesma coisa que Java
 - Derivada de ECMA Scrip

Onde colocar o código JavaScript?

- Pode-se incorporar o código JavaScript no <head>
- Pode-se incorporar o código JavaScript no <body>

```
<script>  
    ...  
</script>
```

- Pode-se utilizar um documento externo com seu código JavaScript.

```
<head>  
    <script src="scripts.js"></script>  
</head>
```

JavaScript no HTML

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="utf-8">
    <title>Exemplo Simples de JavaScript</title>
  </head>
  <body>
    <p>Após carregar o primeiro parágrafo, um script JavaScript
      é executado.</p>

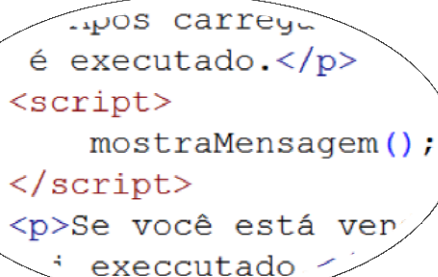
    <script>
      alert('Interrompemos o carregamento desta página com
        um alerta JavaScript!');
    </script>

    <p>Se você está vendo isso, é porque o script já
      foi executado.</p>
  </body>
</html>
```

Utilizando Funções

- O exemplo anterior mistura código JavaScript e marcações HTML
 - Para manter a organização, deve-se utilizar funções

```
<head>
  <meta charset="utf-8">
  <title>Exemplo Simples de JavaScript</title>
  <script>
    function mostraMensagem() {
      alert('Interrompemos o carregamento desta página com
      um alerta JavaScript!');
    }
  </script>
</head>
```



```
    <p>Após carregamento
    é executado.</p>
    <script>
      mostraMensagem();
    </script>
    <p>Se você está vendo
    esta mensagem, o carregamento
    não foi interrompido.</p>
```


Funções

- Algumas vantagens de se utilizar funções em JavaScript:
 - A maior parte do código está fora da marcação
 - Reuso do código e melhoria do desempenho (cache)
 - Uso de arquivos JavaScript externos
 - Pode-se utilizar eventos

Movendo o código para um arquivo externo

- Criar o arquivo `script.js` com o seguinte código:

```
function mostraMensagem() {  
    alert('Interrompemos o carregamento desta página com  
    um alerta JavaScript!');  
}
```

- Referenciar o arquivo externo no código HTML:

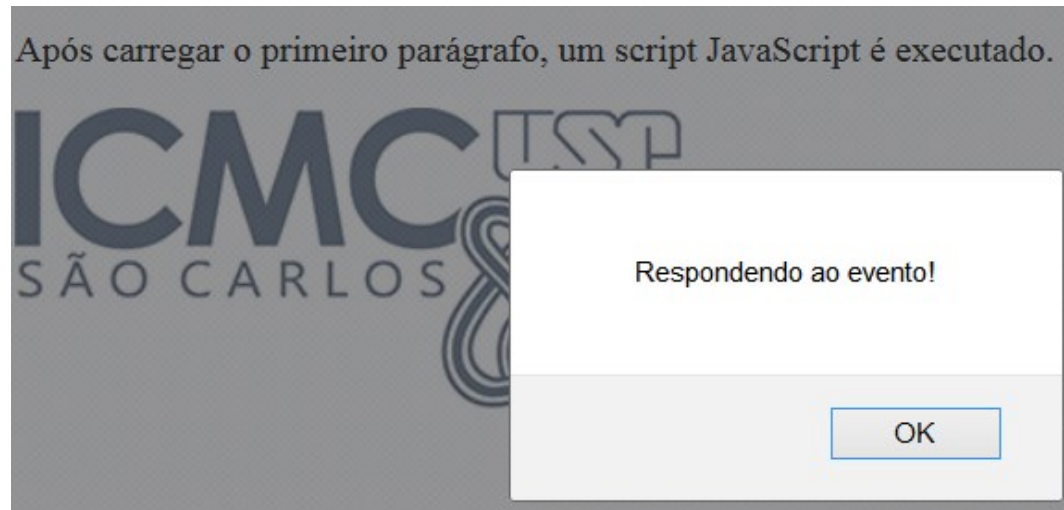
```
<script src="script.js"></script>
```

Eventos

- Eventos JavaScript são notificações que um elemento HTML envia quando coisas específicas acontecem
 - Mover o mouse sobre um elemento (`onmouseover`), clicar em um botão (`onClick`), etc.

```

```



Eventos

Nome do Evento	Disparado ao	Pode ser aplicado em
<code>onClick</code>	Clicar em um elemento	Todos elementos
<code>onMouseOver</code>	Passar o cursor do mouse sobre um elemento	Todos elementos
<code>onMouseOut</code>	Tirar o cursor do mouse de um elemento	Todos elementos
<code>onKeyDown</code>	pressionar uma tecla	<code><select></code> , <code><input></code> , <code><textarea></code> , <code><a></code> , <code><button></code>
<code>onKeyUp</code>	soltar uma tecla pressionada	<code><select></code> , <code><input></code> , <code><textarea></code> , <code><a></code> , <code><button></code>
<code>onFocus</code>	receber o foco	<code><select></code> , <code><input></code> , <code><textarea></code> , <code><a></code> , <code><button></code>

Eventos

Nome do Evento	Disparado	Pode ser aplicado em
onBlur	Ao perder o foco	<select>, <input>, <textarea>, <a>, <button>
onChange	Quando o valor em um campo é alterado. Só é acionado após mover o foco pra outro campo	<select>, <input type="text">, <textarea>
onSelect	Quando uma porção do texto em um campo é selecionado	<input type="text">, <textarea>
onError	Quando o navegador falha ao carregar uma imagem	
onLoad	Após carregar completamente uma nova página	, <body>
onUnload	Quando o navegador descarrega uma página	<body>

Variáveis

- Variáveis JavaScript são criadas com o uso da palavra chave **var** seguida pelo nome da variável
 - JavaScript permite declarar variáveis mesmo sem o var, mas isso não é recomendado

```
var mensagem;
```

- Diferencia maiúsculas de minúsculas (*case sensitive*)
- Atribuição: deve-se utilizar o sinal =

```
var mensagem = "Minha mensagem";  
alert(mensagem);
```

Tipos de Dados

- É possível armazenar diferentes tipos de dados, como texto, números inteiros, *arrays*, objetos, etc.
 - Não é possível definir o tipo de dado da sua variável
- Pode-se substituir o valor de uma variável que possui um texto com um número por exemplo (dinamicamente tipada):

```
var mensagem = "Minha mensagem";  
mensagem = 30.2;
```

- Requer mais atenção!

```
variavel = inputElement.value;  
variavel = inputElement;
```

Operadores Aritméticos

Operador	Operação	Exemplo
+	Adição	$x+y$
-	Subtração	$x-y$
*	Multiplicação	$x*y$
/	Divisão	x/y
%	Módulo (resto da divisão inteira)	$x\%y$
-	Inversão de sinal	$-x$
++	Incremento	$x++$ ou $++x$
--	Decremento	$x--$ ou $--x$

Operadores de Comparação

Operador	Função	Exemplo
==	Igual a	(x == y)
!=	Diferente de	(x != y)
===	Idêntico a (igual e do mesmo tipo)	(x === y)
!==	Não Idêntico a	(x !== y)
>	Maior que	(x > y)
>=	Maior ou igual a	(x >= y)
<	Menor que	(x < y)
<=	Menor ou igual a	(x <= y)

Operadores Bit a bit

Operador	Operação	Exemplo
&	E (AND)	(x & y)
	OU (OR)	(x y)
^	Ou Exclusivo (XOR)	(x ^ y)
~	Negação (NOT)	~x
>>	Deslocamento à direita (com propagação de sinal)	(x >> 2)
<<	Deslocamento à esquerda (preenchimento com zero)	(x << 1)
>>>	Deslocamento à direita (preenchimento com zero)	(x >>> 3)

Operadores de Atribuição

Operador	Exemplo	Equivalente
=	x = 2	Não possui
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
&=	x &= y	x = x & y
=	x = y	x = x y
^=	x ^= y	x = x ^ y
>>=	x >>= y	x = x >>= y
<<=	x <<= y	x = x <<= y
>>>=	x >>>= y	x = x >>>= y

Operadores Lógicos

Operador	Função	Exemplo
&&	E Lógico	(x && y)
 	OU Lógico	(x y)
!	Negação Lógica	! x

Exercício 1

Desenvolva uma aplicação Web (HTML5+ JS) em que:

- ao clicar em um botão (“Clique aqui”), inserido em uma página HTML, uma função JS é chamada;
- a função JS deve receber a resposta de confirmação ou não da operação
 - Objeto JS “confirm”;
- a resposta (confirmado ou não confirmado) deve ser apresentada no HTML corrente.
 - Objeto JS “document”.

Arrays

- Para declarar um array, deve-se utilizar o seguinte formato:

```
var colorList = [];
```

- Pode-se adicionar elementos em um array utilizando índices:

- colorList[0] = “vermelho”;
- colorList[1]=“verde”;
- colorList[2]=“azul”;

```
for (var i = 0; i < colorList.length; i++) {  
    alert("A cor é: " + colorList[i]);  
}
```

Funções que recebem e retornam dados

```
function mostraMensagem(mensagem) {  
    alert(mensagem);  
}  
mostraMensagem("Minha mensagem");  
  
function multiplica(numA, numB) {  
    return numA * numB;  
}  
  
// passa dois numeros como parametro e pega a resposta.  
var resultado = multiplica(3202, 23405);  
// combina o resultado com a mensagem.  
var msg = "O produto de 3202 por 23405 é " + resultado;  
// mostra a mensagem.  
mostraMensagem(msg);
```

Manipulação de Documentos Estruturados

- **Documentos Estruturados**
 - **HTML**
 - **XML & cia**

Técnicas de Manipulação de Documentos Estruturados

Criação de analisadores léxicos, nos quais letra por letra é examinada obtendo assim as informações desejadas.

muito trabalhoso....

- *Simple API for XML (SAX).*
- *Document Object Model (DOM).*
 - *HTML DOM*
 - *XML DOM*



Simple API for XML (SAX)

- Desvantagens:

- ✓ O SAX é recomendado para a leitura (*read-only*).
- ✓ Não possui muitas operações de manipulação dos dados.

- **Vantagens:**

- ✓ Não necessita armazenar ou carregar todo o conteúdo do arquivo em memória.
- ✓ É recomendado para arquivos grandes.

Document Object Model (DOM)

- **Vantagem:**

- Possui API robusta para a manipulação dos dados.

- **Desvantagens:**

- Consumo de memória, pois armazena todo o documento em memória em forma de árvore.
- Não recomendado para arquivos grandes.

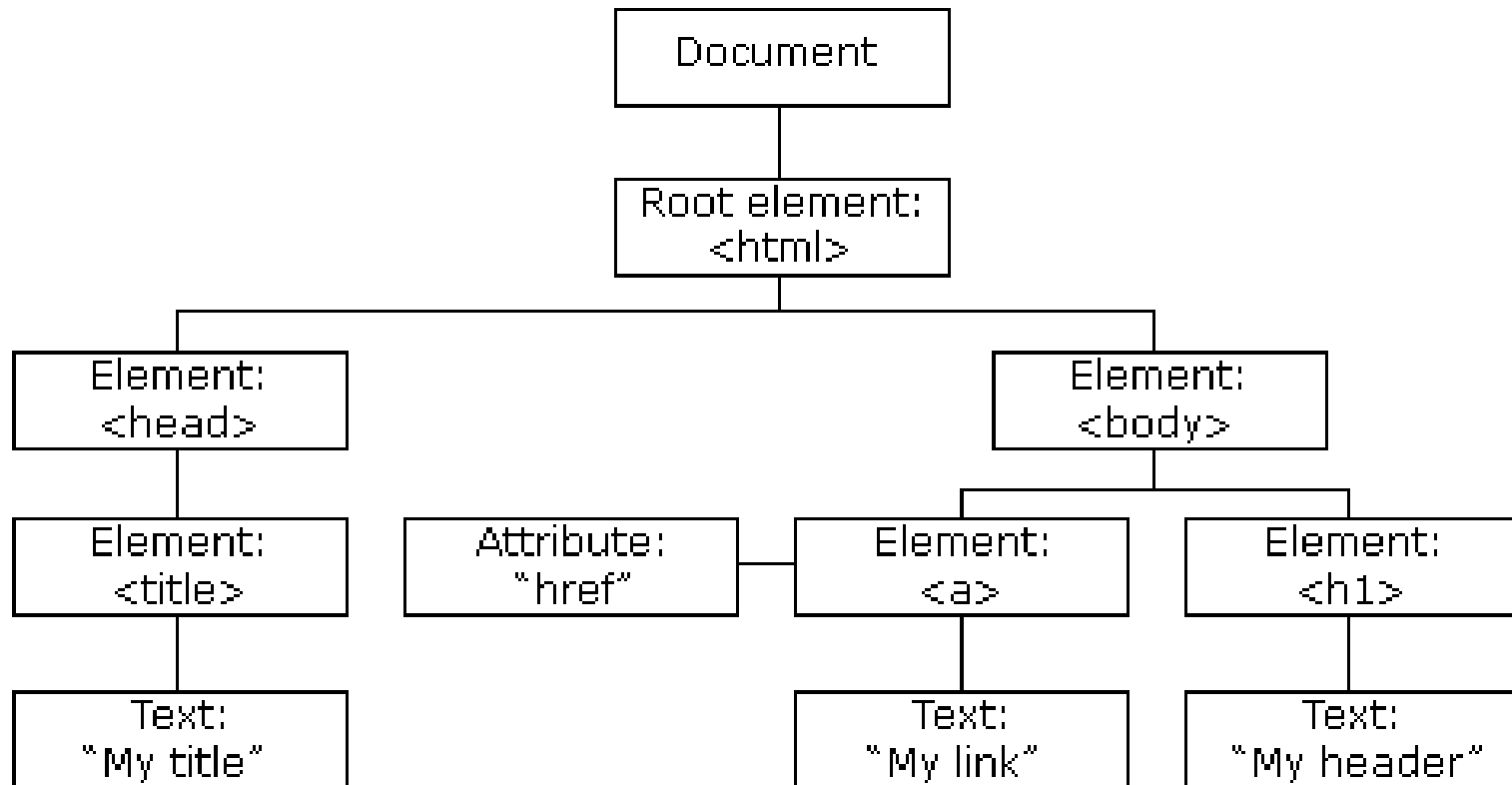
DOM

- Encontra-se na terceira versão.
- **Pode ser utilizado para HTML também.**
- É um padrão W3C.
- XML DOM é um modelo de objeto de documento para XML.
- Independente de linguagem e plataforma.
- Define um padrão para a manipulação de documentos XML.
- Define um padrão para acessar objetos.
“Parser de arquivos estruturados”.

DOM e Árvore

- DOM é utilizado para manipulação de arquivos.
- Carrega os arquivos em forma de árvore na memória.
- Todos os elementos são manipulados como nós da árvore.
- Relação hierárquica entre os nós
 - Pais, filhos, vizinhos, raiz e nós folhas
- Manipular árvores e nós já é de conhecimento dos programadores → Estrutura de Dados.

Árvore DOM HTML (exemplo)



Árvore DOM

Objetivo: Determinar e mostrar como os dados serão representados e manipulados em memória.

DOM e JavaScript

- O objeto JavaScript `document` permite o acesso à árvore DOM do documento HTML apresentado no navegador Web.
- JavaScript e DOM permite mover, criar e remover itens em uma página Web.
- As propriedades CSS também podem ser alteradas.
- Não é necessário recarregar a página para alterar a sua exibição.

Propriedades DOM

* innerHTML – contém o texto (conteúdo)

nodeName – contém o nome de um nó

value – contém o valor (conteúdo) de um nó

parentNode – o pai de um nó

childNodes – os nós filhos de um nó

attributes – os atributos de um nó

...

Uso: no.propriedade

Métodos DOM

`getElementById(id)`

`getElementsByTagName(name)`

`appendChild(node)`

`removeChild(node)`

...

Uso: no.método

Referência:

- <http://www.w3schools.com/jsref/default.asp>

Buscar Elementos HTML

Método	Descrição
<code>document.getElementById()</code>	Busca um elemento pelo seu ID
<code>document.getElementsByTagName()</code>	Busca elementos pelo nome da Tag
<code>document.getElementsByClassName()</code>	Busca elementos pela classe
<code>element.parentNode</code>	Retorna o nó pai de um elemento

Alterar Elementos HTML

Método	Descrição
<code>element.innerHTML=</code>	Altera o conteúdo HTML de um elemento
<code>element.attribute=</code>	Altera o atributo de um elemento HTML
<code>element.setAttribute(attribute,value)</code>	Altera o atributo de um elemento HTML
<code>element.style.property=</code>	Altera o estilo de um elemento HTML

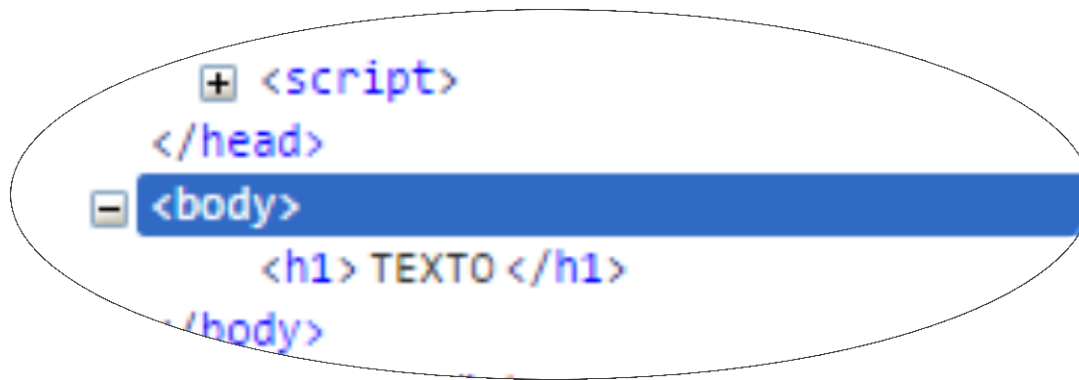
Adicionar/Remover Elementos HTML

Método	Descrição
<code>document.createElement()</code>	Cria um elemento HTML
<code>document.removeChild()</code>	Remove um elemento HTML
<code>document.appendChild()</code>	Adiciona um elemento HTML
<code>document.replaceChild()</code>	Substitui um elemento HTML
<code>document.write(<i>text</i>)</code>	Adiciona texto no documento HTML
<code>document.createAttribute()</code>	Cria um atributo que pode ser inserido em um elemento

Exemplo1 – Criar Elemento

```
<html>
  <head>
    <script>
      // executa esta função quando o documento está carregado
      window.onload = function() {
        // cria um novo elemento h1 em
        // página HTML vazia
        heading = document.createElement("h1");
        heading_text = document.createTextNode("TEXT0 ");
        heading.appendChild(heading_text);
        document.body.appendChild(heading);
      }
    </script>
  </head>
  <body>
  </body>
</html>
```

Exemplo1 - Resultado



TEXTO

Exemplo2 – Remover Elemento

```
<html>
<head>
  <script>
    removeElemento = function() {
      var parent = document.getElementById("div1");
      var child = document.getElementById("p1");
      parent.removeChild(child);
    }
  </script>
</head>
<body>
  <div id="div1">
    <p id="p1">Esse é o primeiro parágrafo.</p>
    <p id="p2">Esse é o segundo parágrafo.</p>
    <a href="#" onclick="removeElemento();">Remover primeiro parágrafo</a>
  </div>
</body>
</html>
```

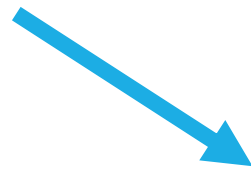

Exemplo2 - Resultado



Esse é o primeiro parágrafo.

Esse é o segundo parágrafo.

[Remover primeiro parágrafo](#)



Esse é o segundo parágrafo.

[Remover primeiro parágrafo](#)

Exercício 1

**** Produzam um HTML bem-formado! ****

```
<body>
  <h1>Título?</h1>
  <br/>
  <h3>
    <form>
      Digite um Título: <input type="text"
        id="query" size="70"/>
      <input type="button" value="Atribuir
        Título" onclick="busca()" />
    </form>
  </h3>
</body>
```

Exercício 2

`** Idem **`

```
<body>
  <h1>Troca</h1>
  <h1>Título</h1>
  <h1>Troca</h1>
  <br/>
  <h3>
    <form>
      ...
    </form>
  </h3>
</body>
```

JQuery

jQuery

- **O quê você já deve(ria) saber: básico de HTML, CSS e JS**
- jQuery é uma biblioteca JavaScript que facilita o desenvolvimento fornecendo soluções prontas para diversas tarefas comuns.
 - Relativamente pequena
 - Amigável aos web designers
 - Amplamente testado
 - É gratuito
 - Grande comunidade de desenvolvimento
 - Plugins

Utilizando jQuery

- Existem duas formas para referenciar o jQuery na página HTML:
- Para o download do arquivo, visite jquery.com/download
 - Inclua o caminho para o arquivo na página HTML

```
<script src="js/jquery-2.1.3.min.js"></script>
```
- Para utilizar um CDN (*Content Distribution Network*)
 - Google CDN

```
<script  
src="//ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.m  
in.js"> </script>
```
- Deve-se incluir o caminho para a biblioteca antes de qualquer outro código jQuery!

Utilizando jQuery

- Sintaxe: baseada em “seleção” + “ação” em elementos HTML:

\$(seletor).ação()

- \$ para definir/acessar jQuery
- (*seletor*) para buscar elementos HTML
- Uma ação() jQuery para ser executada em um elemento

Utilizando jQuery

- Para começar a utilizar o jQuery, inclua um **segundo** script:

```
<script> $(document).ready(function() {  
// o código deve ser escrito aqui  
});  
</script>
```

- Ou `<script src="myFunctions.js"></script>`
- A função `.ready()` determina que todo código só será executado após o carregamento completo da página
 - Boa prática!

Exemplo

teste.html

teste.js

Versões

- Atualmente existem duas versões de jQuery: 1 e 2
- Hoje a funcionalidade das versões 1.11 e 2.1 são as mesmas.
- jQuery 2 removeu o suporte para os navegadores Internet Explorer 6,7 e 8.
 - Suportar esses navegadores requer mais código, aumentando o tamanho do arquivo.
- Enquanto os navegadores IE 6,7 e 8 continuam a ser utilizados, recomenda-se o uso da versão 1.1 do jQuery.
- Nesta disciplina usaremos jQuery 2

Selecionando elementos

- jQuery permite identificar e alterar elementos com base nos seletores (nomes de TAGs)
 - `$ ("seletor")`
- Por exemplo, para selecionar uma tag "p"
 - `$ ("p")`
- Para alterar o HTML dentro deste elemento
 - `$ ("p").html ("<h1>Novo HTML</h1>");`

Seletores Básicos

- Seletores de ID “#”

```
<p id="mensagem"> Uma mensagem </p>  
var paraMensagem = $("#mensagem");
```

- Seletores de Elemento (ou Tag)

```
var todosLinks = document.getElementsByTagName("a");  
var todosLinks = $("a");
```

- Seletores de Classe “.”

```
$(".menu")  
$(".menu").hide();
```

Percurso (*Traversing*) jQuery

- Usado para selecionar/encontrar elementos HTML baseado em suas relações hierárquicas (árvore DOM).
- Métodos para
 - Ancestrais: `parent()`, `parents()`, `parentsUntil()`
 - Descendentes: `children()`, `find()`
Ex.:

```
$(document).ready(function(){  
    $("div").children(); /* $("div").find("p"); */  
});
```
 - Irmãos: `siblings()`, `next()`, `nextAll()`, `nextUntil()`, `prev()`, `prevAll()`, `prevUntil()`

Filtros jQuery

- Permite filtrar os elementos selecionados com base em certas características.
- `first()`, `last()`, `eq()`, `filter()`, `not()`, ...
- ```
$(document).ready(function(){
 $("div p").first();
});
```

# Entendendo a seleção

---

- **Loops Automáticos** – Como criar loops para percorrer uma coleção de objetos é muito comum, essa característica é padrão em jQuery
  - Ao aplicar uma função em uma seleção de elementos jQuery, automaticamente é aplicado o loop
  - Ex: `$ ("p") .hide () ;`
- **Funções encadeadas** – jQuery utiliza um princípio de encadeamento, que permite que sejam adicionadas funções uma após a outra (ligadas por um “.”)
  - Exemplo:  
`$ (' #popUp') .width (300) .height (300) .text ('Oi!') .fadeIn (10) ;`

# Adicionando conteúdo em uma página

---

- `.html()` – Permite ler ou alterar o conteúdo HTML de um elemento:
  - `alert($("#p").html());`
  - `$("#p").html("<b>Campo obrigatório!</b>"); /*negrito*/`
- `.text()` – Permite ler ou alterar o conteúdo de um elemento:
  - `$("#p").text("<b>Campo obrigatório!</b>"); /*<b> aparece!*/`
- `.append()` – Adiciona o conteúdo HTML como o **último** filho do elemento selecionado:
  - `$('#erros').append("<p>Campo obrigatório </p>");`
- `.prepend()` – Adiciona o conteúdo HTML como o **primeiro** filho do elemento selecionado



# Adicionando conteúdo em uma página

---

- `.before()` ou `.after()` – Adiciona HTML antes ou depois do elemento selecionado. Exemplo:

```
<input type="text" name="usuario" id="usuario">
$('#usuario').after('Texto');
$('#usuario').before('obrigatório');
```

- Resultado:

- `<span class="erro">Obrigatório</span>`
- `<input type="text" name="usuario" id="usuario">`
- `<span class="erro">Texto</span>`

# Substituindo e Removendo

---

- `.remove()` – remove um único elemento ou uma seleção de elemento. Exemplo:

```
$ ('#popup') .remove () ;
```

```
$ ('span.erros') .remove () ;
```

- `.replaceWith()` – substitui a seleção com um novo conteúdo. Ex:

```
$ ('#produto101') .replaceWith (<p>Adicionado ao carrinho</p> ') ;
```

# Definindo e Lendo atributos

---

- `.addClass()` – adiciona uma classe específica a um elemento. Exemplo:

```

$('a[href^="http://"]').addClass('linkExterno');

```

- `.removeClass()` – é o **oposto** do `.addClass()`. Exemplo:

- `$( '#alertBox' ).removeClass('highlight');`

- `.toggleClass()` – A funcionalidade é alternada. **Remove** a classe se ela já existe ou **Adiciona** a classe caso ela ainda não exista.

```
$('#changeStyle').click(function() {
 $('body').toggleClass('altStyle');
});
```

# Eventos

---

Evento	Descrição
<code>\$ ('seletor') .<b>click</b> (function)</code>	Invoca uma função ao clicar no elemento
<code>\$ ('seletor') .<b>dblclick</b> (function)</code>	Invoca uma função após um duplo clique no elemento
<code>\$ ('seletor') .<b>focus</b> (function)</code>	Invoca uma função quando o elemento recebe o foco
<code>\$ ('seletor') .<b>mouseover</b> (function)</code>	Invoca uma função ao passar o mouse sobre o elemento
<code>\$ ('seletor') .<b>keypress</b> (function)</code>	Invoca uma função quando uma tecla é pressionada
<code>\$ ('seletor') .<b>submit</b> (function)</code>	Invoca uma função ao submeter o formulário

Para uma lista completa dos eventos, consulte [Events | jQuery API Documentation](#)

# Eventos – Exemplo de uso

---

1) Selecione o elemento

```
$ ('#menu')
```

2) Inclua o evento

```
$ ('#menu') .mouseover ();
```

3) Crie uma função anônima

```
$ ('#menu') .mouseover (function () { });
```

4) Inclua as ações desejadas

```
$ ('#menu') .mouseover (function () {
 $ ('#submenu') .show ();
}) ;
```

# Efeitos

Função	Descrição
<code>\$('seletor').hide()</code>	Esconde os elementos selecionados
<code>\$('seletor').show()</code>	Apresenta os elementos selecionados
<code>\$('seletor').toggle()</code>	Alterna (entre apresentar e esconder) os elementos
<code>\$('seletor').slideDown()</code>	Desliza para baixo e apresenta o elemento selecionado
<code>\$('seletor').slideUp()</code>	Desliza para cima e esconde o elemento selecionado
<code>\$('seletor').slideToggle()</code>	Alterna entre <code>slideDown</code> e <code>slideUp</code>
<code>\$('seletor').fadeIn()</code>	Gradualmente esconde o elemento
<code>\$('seletor').fadeOut()</code>	Gradualmente apresenta o elemento
<code>\$('seletor').fadeTo()</code>	Gradualmente esconde o elemento até determinada opacidade

# Referências

---

BOOKMAN, Ano de Edição: 2002, Capítulo 4, Document Object Model (DOM), disponível em: <http://www.inf.ed.ac.uk/teaching/courses/ec/miniatures/dom-up.pdf>, acessado em 04 de Março de 2015.

W3Schools

- <http://www.w3schools.com/html/dom/default.asp>
- <http://www.w3schools.com/jsref/default.asp>

---

Este material tem sido produzido e atualizado colaborativamente pelos professores do grupo de Sistema Web e Multimídia Interativos do ICMC-USP.