

INSTITUTO FEDERAL DE EDUCAÇÃO CIÊNCIA E TECNOLOGIA
CURSO TÉCNICO INTEGRADO EM INFORMÁTICA
CAMPUS CURRAIS NOVOS

Desenvolvimento Web

JavaScript – aula IV –

*Acesso a documentos com Document
Object Model (DOM)*

Professor: Bruno E. G. Gomes

2013

INTRODUÇÃO

- O *HTML* é a interface de operação do *JavaScript*
- Ao código *JavaScript* é permitido:
 - Acessar e manipular o documento HTML
 - Alterar estilos dinamicamente
 - Validar formulários, etc.
- *JavaScript* se utiliza do *Document Object Model (DOM)* do HTML para realizar as manipulações



DOCUMENT OBJECT MODEL (DOM)

- O navegador interpreta cada palavra, letra ou símbolo de um *HTML* e exibe o resultado na tela
 - Esse resultado é uma página visível para o usuário do navegador
- Resultado da interpretação do *HTML* é armazenado em uma estrutura de objetos
 - *Document Object Model (DOM)*
- Cada elemento, atributo e texto *HTML* no *DOM* torna-se um objeto
 - Objetos podem ser acessados de modo independente pelos *scripts*



DOM

- É um padrão definido pelo W3C para acesso a documentos
- DOM do HTML define:
 - *Objetos e propriedades* de todos os elementos em um documento HTML/XHTML
 - Métodos para manipular (obter/modificar/adicionar/apagar) cada elemento



DOCUMENT OBJECT MODEL (DOM)

- Todo documento XHTML é uma hierarquia de elementos
 - Todos são subordinados à *tag* `<html>`
 - `<html>` possui um cabeçalho e um corpo (*body*)
 - Corpo possui outras *tags* ligadas a ele
- Essa hierarquia, em DOM, é representada como uma estrutura de dados de árvore
- DOM define objetos e propriedades para cada elemento
 - Cada objeto de elemento é ligado ao seu objeto “pai” na árvore DOM



EXEMPLO

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Testando DOM</title>
```

```
  </head>
```

```
  <body>
```

```
    <h2>Document Object Model (DOM)</h2>
```

```
    <p>
```

DOM permite o acesso e a manipulação de documentos HTML por meio de funções acessíveis ao *JavaScript*.

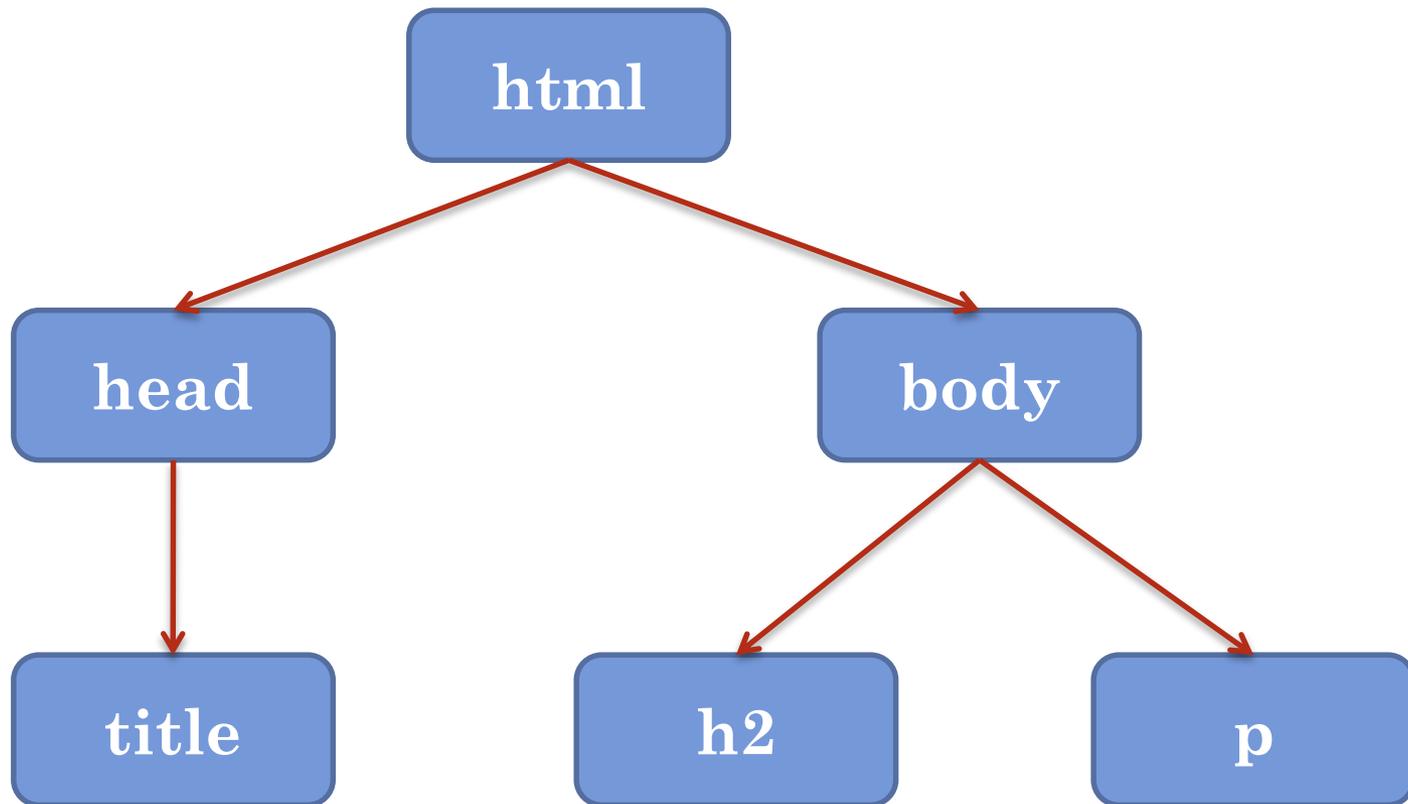
```
  </p>
```

```
  </body>
```

```
</html>
```



REPRESENTAÇÃO DO EXEMPLO ANTERIOR NA ÁRVORE DOM

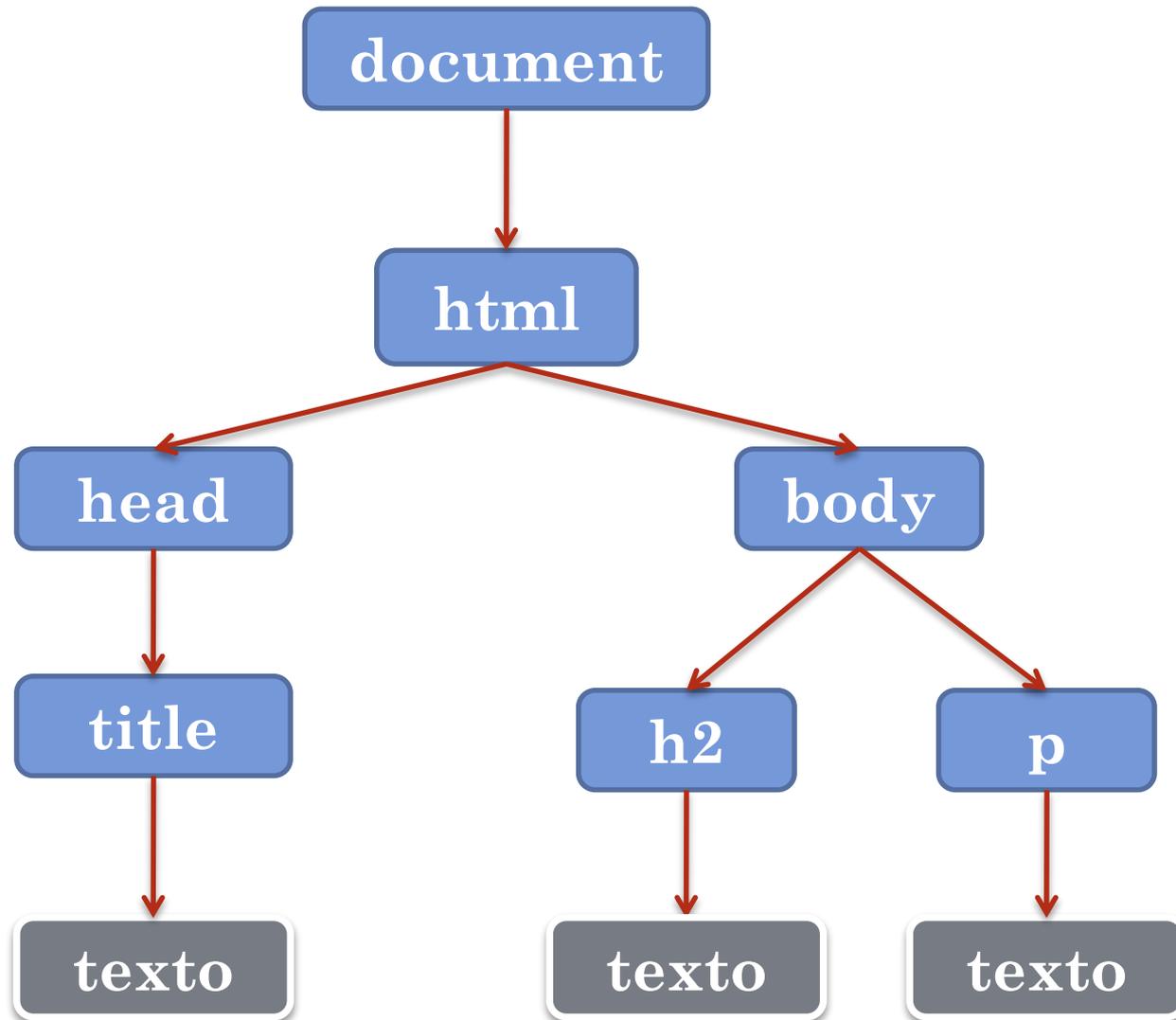


NÓS DA ÁRVORE DOM

- Cada elemento do HTML é representado como um “nó” na árvore DOM
- Tipos de nós:
 - Elemento `<a>`
 - Atributo `href, id, class, etc.`
 - Texto `“http://www.abc.com”`
 - Comentário `<!-- texto ->`
- *Nó de documento (document):*
 - É a raiz de toda a representação DOM
 - Ponto de entrada para obter os outros nós



EXEMPLO ANTERIOR COM INFORMAÇÕES COMPLEMENTARES



EXEMPLO

- O valor de um nó de atributo é uma referência ao próprio atributo
 - como um apontador para o atributo
- O texto dentro de um atributo pode ser acessado com a propriedade **innerHTML**
- Exemplo:

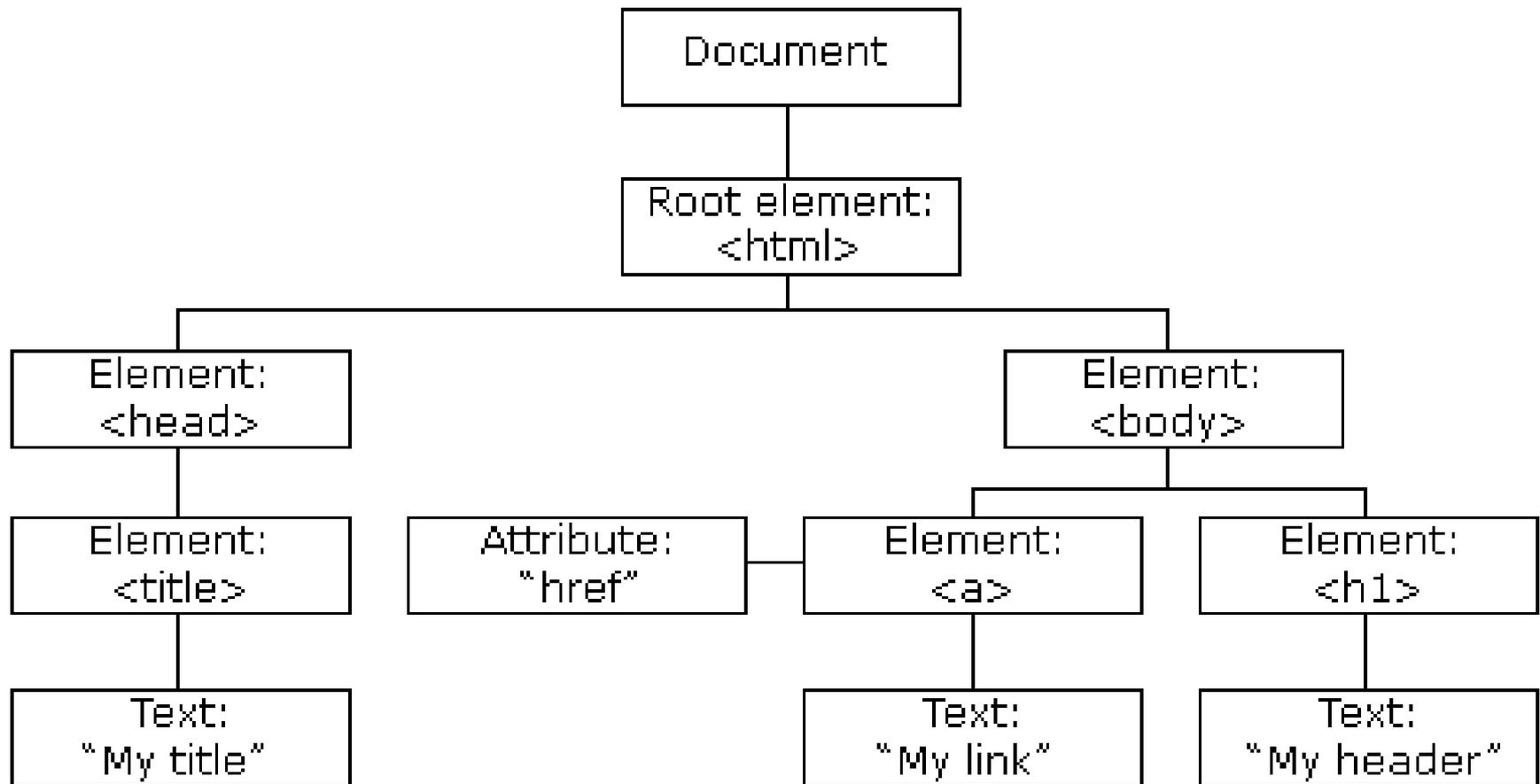
```
<title>Testando DOM</title>
```

→ Elemento nó é *<title>*

→ Texto “Testando DOM” é “filho” do elemento *<title>*

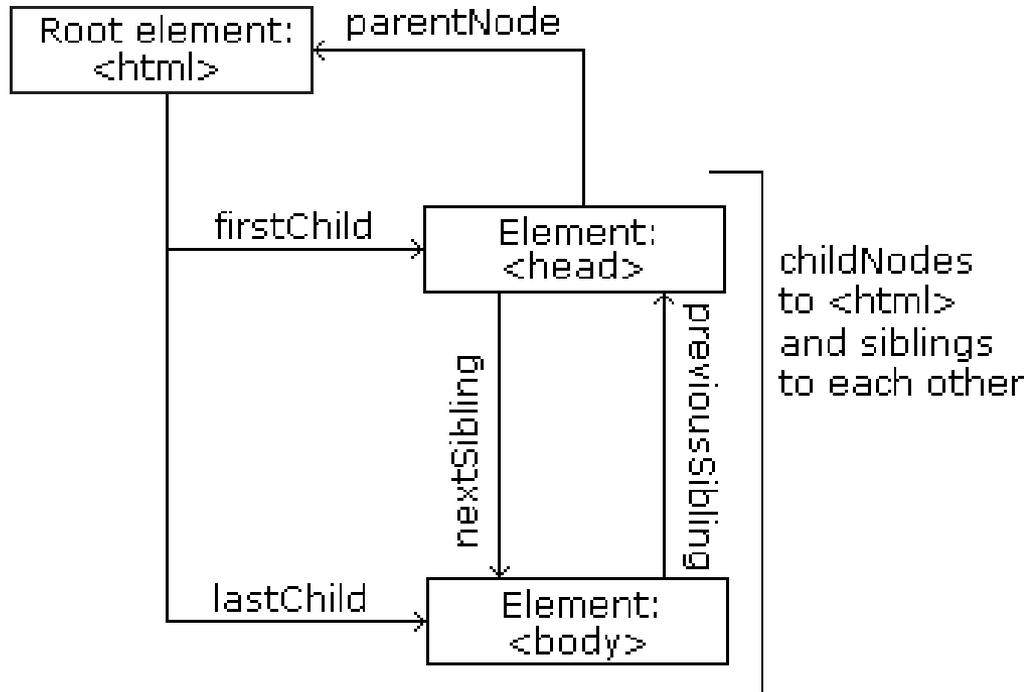


ESQUEMA GERAL DE NÓS DOM



RELACIONAMENTO ENTRE OS NÓS: PAIS, FILHOS E IRMÃOS

- Em uma árvore:
 - Nó topo é chamado de **principal** (*root*)
 - Todo nó possui um **pai** (exceto o *root*)
 - Um nó pode ter qualquer número de **filhos**
 - **Irmãos** (*siblings*) são nós com o mesmo pai
 - Um nó sem filhos é chamado de **folha**



EXEMPLO

```
<html>
  <head> <title>DOM Tutorial</title> </head>
  <body>
    <h1>DOM Lesson one</h1>
    <p>Hello world!</p>
  </body>
</html>
```

→ `<html>` - nó principal

→ `<head>` e `<body>` são filhos do nó principal

→ Nó de texto “*DOM Lesson one*” tem como pai o nó de elemento `<h1>`

OBTENDO OS NÓS DESEJADOS

- Cada nó DOM possui informações que podemos usar para identificá-lo
- Podemos localizar um elemento por
 1. ID
 2. Nome da *tag*
 3. Navegação na árvore DOM, utilizando o relacionamento entre elementos (pai, filho, etc.)
- Manipulação de um elemento pode meio do DOM normalmente segue um padrão:
 1. Obter o elemento ou grupo de elementos que desejamos realizar algo
 2. Especificar o “efeito” que desejamos aplicar nele(s)



LOCALIZANDO PELO ID

- Localizar o elemento pelo ID é bastante fácil
 - ID identifica unicamente o elemento
- É importante verificar se o ID realmente existe, antes de fazer alguma coisa com o resultado da busca
- Utilizamos o método **getElementById**, do objeto (nó) *document*
 - Retorna uma referência para o nó ou **null** se ele não for encontrado



EXEMPLO – ACESSANDO O ELEMENTO DE ÂNCORA PELO ID

```
<html>
  <head><title>Testando DOM</title> </head>
  <body>
    <h2>Document Object Model (DOM)</h2>
    <p>
      Veja a especificação DOM em:
      <a id="ref1" href="http://www.w3.org/DOM/">DOM</a>
    </p>
    <script type="text/javascript" src="ex1.js">
    </script>
  </body>
</html>
```

OBS.: Script no próximo slide

EXEMPLO – ACESSANDO O ELEMENTO DE ÂNCORA PELO ID

```
var target = document.getElementById("ref1");

if (target != null) {
  window.alert (
    "Nome do nó: " + target.nodeName + "\n" +
    "HTML dentro do nó: " + target.innerHTML + "\n" +
    "ID: " + target.id + "\n" +
    "Tem filhos? " + target.hasChildNodes();
  );
} else {
  window.alert("Elemento não encontrado.");
}
```



LOCALIZANDO PELO NOME DA TAG

- Para recuperar um grupo de elementos pelo nome da *tag* utilizamos o método
getElementsByTagName(<nome_da_tag>)
- Pode ser chamado a partir de qualquer nó
 - não apenas do nó *document*, como acontece com *getElementById*
- Retorna uma lista de nós:
 - Todos os nós que tenham o nome especificado a partir do nó de origem onde o método foi chamado



EXEMPLO

- O XHTML abaixo será manipulado pelo script do próximo *slide*

```
<body>
  <p>
    No corpo desta página há um parágrafo
    e uma lista não-ordenada.
  </p>
  <ul>
    <li> Primeiro item da lista</li>
    <li> Segundo item da lista</li>
    <li> Terceiro item da lista</li>
  </ul>

  <script type="text/javascript" src="ex2.js">
  </script>
</body>
```

JAVASCRIPT UTILIZANDO MÉTODO *GETELEMENTSBYTAGNAME*

```
var listItems = document.getElementsByTagName("li");  
  
document.write("<p> Há " + listItems.length + " elementos 'li'</p>");  
document.write("<p> Texto dentro dos elementos li: </p>");  
  
for (var i = 0; i < listItems.length; i++) {  
    document.write("<p>" + listItems[i].innerHTML + "</p>");  
}
```



EXERCÍCIO

- Inclua uma outra lista no exemplo anterior e faça um *script* para obter a referência aos itens apenas dessa segunda lista que você adicionou.
- Dica:
 - Para obter os elementos da segunda lista, você deve ler o “segundo” elemento da lista de itens retornados por *document.getElementsByTagName*

