



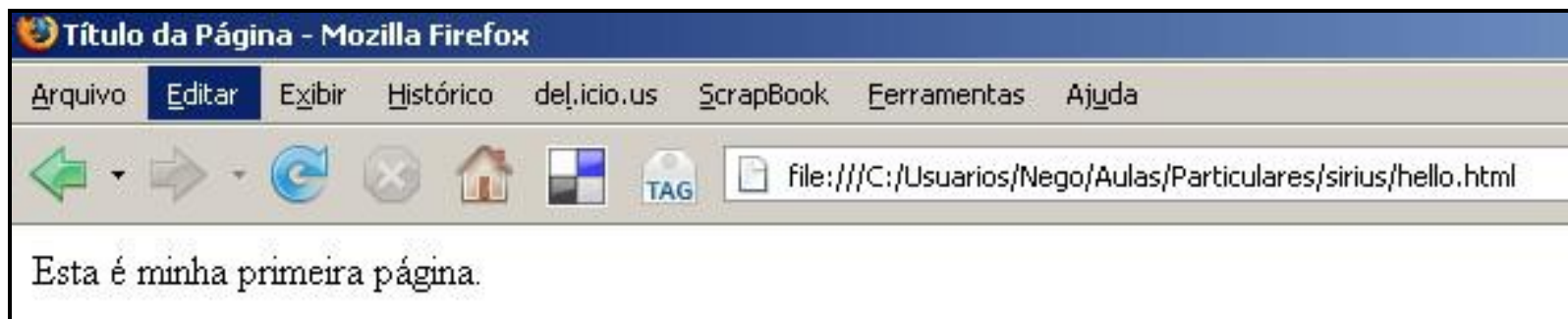
JavaScript



JavaScript

- É a linguagem de script utilizada por milhões de páginas web
- Foi projetada para aumentar interatividade das páginas web:
 - Validação de formulários, interação com o usuário (p.ex., tratamento de cliques de botões), detecção de navegadores, etc
- É reconhecida pela maioria dos navegadores
- **Seu processamento é feito na máquina cliente (browser)**
- **Não há relação com Java**

JavaScript - Exemplo



```
<html>
  <head>
    <title>Título da Página</title>
  </head>
  <body>
    <script type="text/javascript">
      document.write("Esta é minha primeira página.");
    </script>
  </body>
</html>
```

JavaScript – Onde ocorrem

- Uma tag **<script/>** pode ser definida numa seção head, numa seção body e também pode ser definida externamente:
 - Na seção **head**, os scripts são **executados quando são chamados** ou quando algum evento ocorre;
 - Na seção **body**, os scripts são **executados na carga da página web**
 - Para definição externa, um arquivo “.js” precisa ser fornecido com as funções necessárias



JavaScript – Exemplo

```
<html>
  <head>
    <title>Título da Página</title>
    <script src="hello.js"></script>
  </head>
  <body onload="message()">

    <script type="text/javascript">
      document.write("<h1>Esta é minha primeira página.</h1>");
    </script>

  </body>
</html>
```

JavaScript - Sintaxe

- Possui construções existentes na maioria das linguagens de programação (sintaxe similar a C):
 - Declaração de variáveis (*var x;*)
 - Comandos condicionais, repetições, definição de funções de usuário
 - Operadores de atribuição, comparação, ...



JavaScript

Janelas Popup

- Funções para criação de janelas popup:
 - Alerta (*alert("Texto a ser exibido");*)
 - Confirmação (*confirm("Texto a ser exibido em janela Ok");*)
 - Entrada de dados (*prompt("Seu nome");*)
- Janelas que aparecem quando acessamos uma página web

```
<body>  
  <script type="text/javascript">  
    var nome = prompt("Seu nome");  
    if (nome != null && nome != "")  
      document.write("Oi " + nome);  
    else  
      document.write("Oi anônimo!");  
  </script>  
</body>
```

JavaScript – Eventos

- Eventos são ações que podem ser detectadas por um script
- Exemplos de eventos:
 - Clique do mouse, abertura de uma página web ou imagem, envio de um formulário html, uma tecla pressionada, etc
- O tratamento destes eventos pode ser a chamada de funções do script



JavaScript – Eventos

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

```
/* Conteúdo do arquivo event.js */
function mouseOver()
{
  document.imagem.src = "quadrado2.gif";
}
function mouseOut()
{
  document.imagem.src = "quadrado1.gif";
}
```

- Lista de eventos:
http://www.w3schools.com/jsref/jsref_events.asp



JavaScript – Objetos

- **JavaScript é uma linguagem OO**
- Com isto, algumas classes utilitárias padrões estão disponíveis, as quais possuem métodos e propriedades:
 - String: manipulação de strings no script

```
/* Exemplo de função de script que calcula  
o tamanho de uma string */  
function tamanho(msg)  
{  
    return msg.length;  
}
```

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



JavaScript – Objetos

– Date: manipulação de datas no script

```
/* Exemplo de função de script que manipula  
datas */  
function bug(data)  
{  
    var x = new Date();  
    x.setFullYear(2000,0,0);  
    if (data > x)  
        alert("Bug do milênio!");  
}
```

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

JavaScript – Objetos

- Outras classes disponíveis na linguagem JavaScript:
 - Array: armazenamento de conjuntos de valores
 - http://w3schools.sinsixx.com/js/js_obj_array.asp.htm
 - Boolean: manipulação de valores booleanos
 - https://www.w3schools.com/jsref/jsref_obj_boolean.asp
 - Math: manipulação de valores com operações matemáticas
 - http://www.w3schools.com/jsref/jsref_obj_math.asp

JavaScript – Objetos Exemplo

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<p>Clique no botão</p>
```

```
<button onclick="myFunction()"> Clique aqui </button>
```

```
<p id="demo"></p>
```

```
<script>
```

```
function myFunction() {  
  var d=new Date();
```

```
  var dias=new Array(7);
```

```
  dias[0]="Domingo";
```

```
  dias[1]="Segunda";
```

```
  dias[2]="Terça";
```

```
  dias[3]="Quarta";
```

```
  dias[4]="Quinta";
```

```
  dias[5]="Sexta";
```

```
  dias[6]="Sábado";
```

```
  document.write("Hoje é " + d.getDay());
```

```
  document.write("Hoje é " + dias[d.getDay()]);
```

```
  document.write("Dia qualquer: " +dias[Math.round(Math.random()*6)]);
```

```
  document.write(navigator.appCodeName);
```

```
}
```

```
</script>
```

```
</body>
```

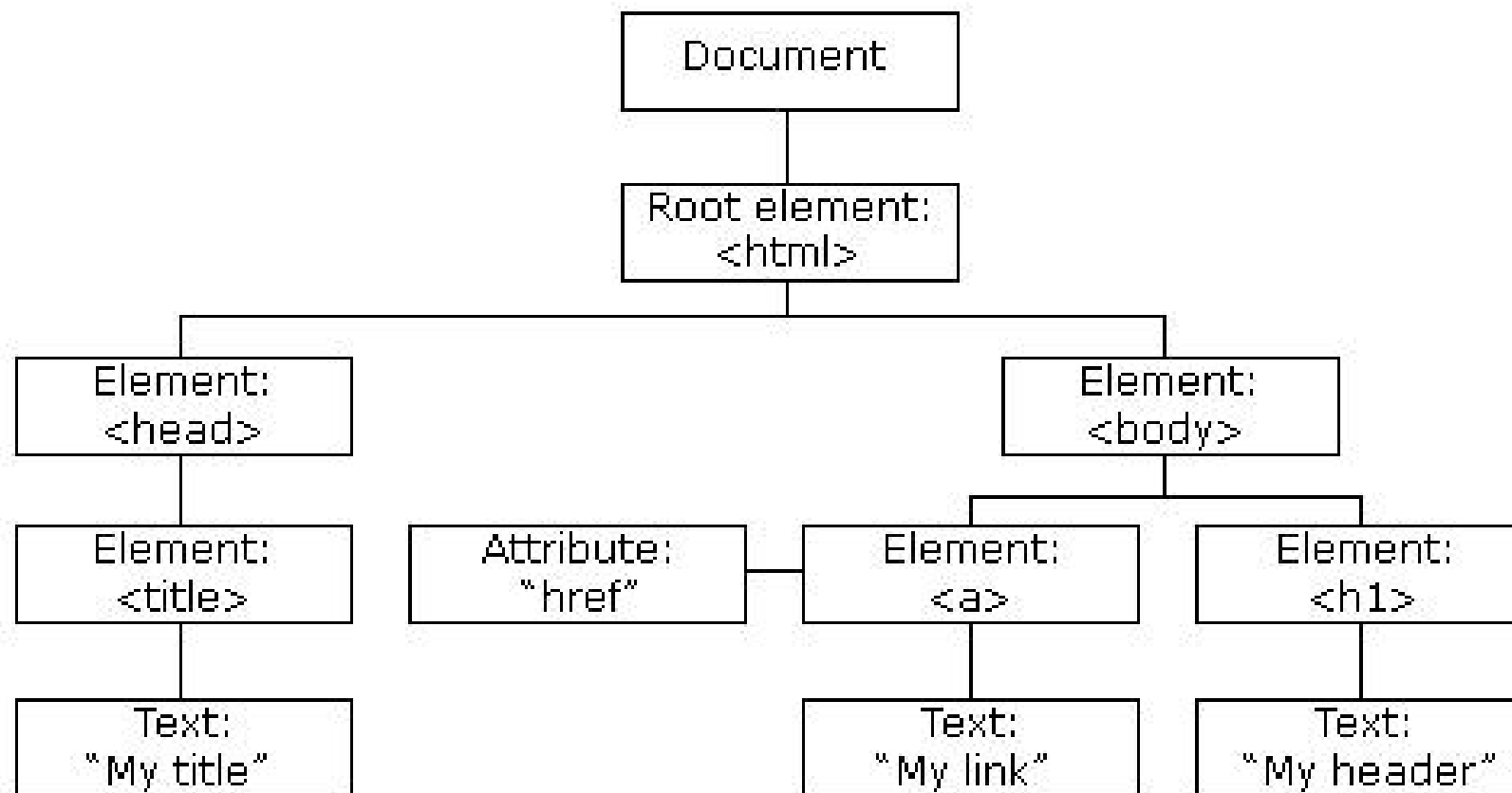
```
</html>
```



JavaScript – Objetos

- Além destas, todos os objetos HTML DOM podem ser acessados através de scripts
 - Window: objeto no topo da hierarquia do JavaScript; representa a janela do browser
 - Navigator: contém informação sobre o browser do cliente
 - Screen: contém informação sobre a tela
 - History: contém as URLs visitadas
 - Location: contém informação sobre a URL corrente
 - https://www.w3schools.com/js/js_htmlDOM.asp

JavaScript – Objetos



JavaScript – Objetos

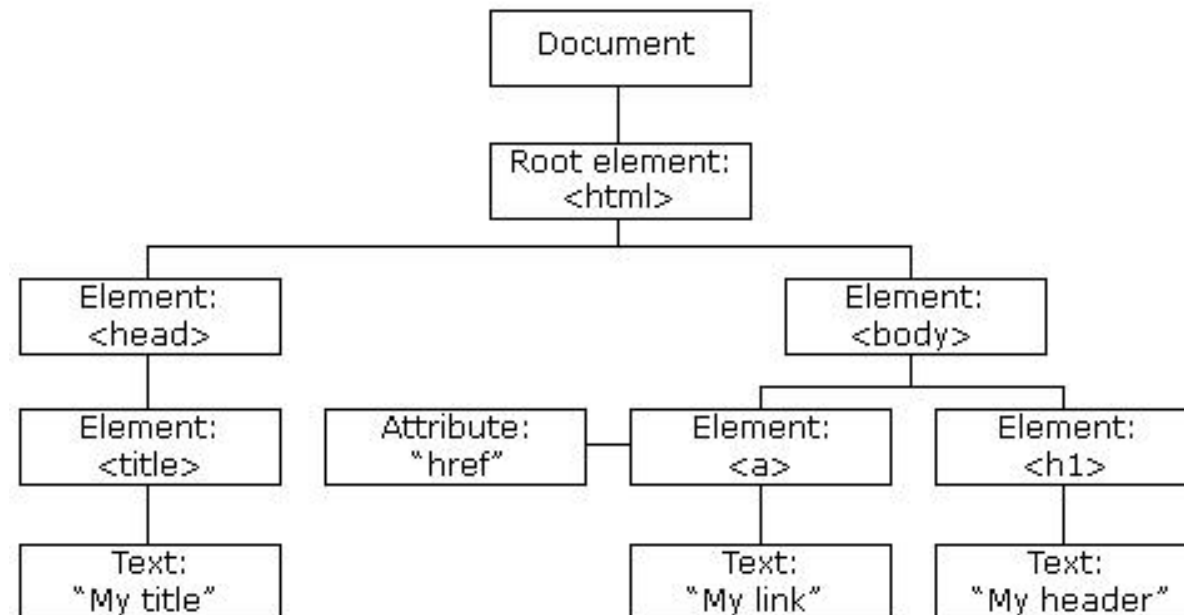
Exemplo com Window

```
<html>
<head>
  <script type="text/javascript">
    function currLocation() {
      alert(window.location);
    }
    function newLocation() {
      window.location="http://www.w3schools.com";
    }
  </script>
</head>
<body>
  <input type="button" onclick="currLocation()" value="Show current URL">
  <input type="button" onclick="newLocation()" value="Change URL">
</body>
</html>
```


HTML DOM

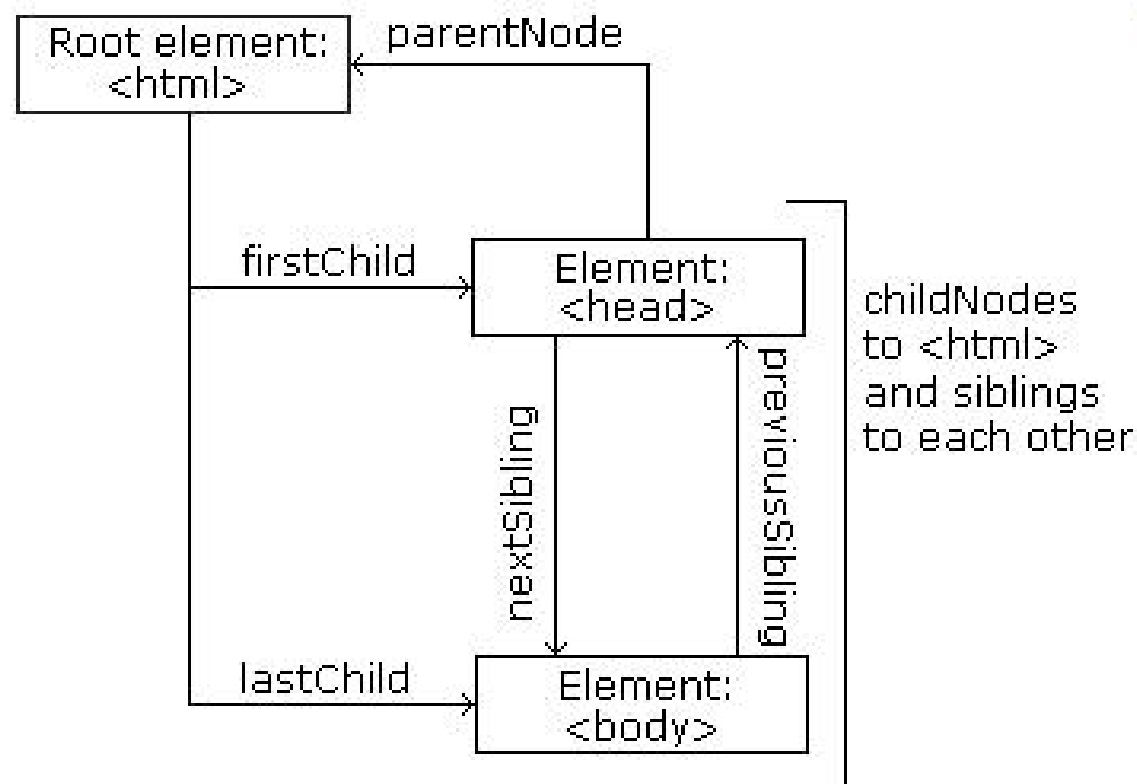
- Define um padrão para acesso a elementos HTML
- O DOM apresenta um documento HTML como uma estrutura em árvore

```
<html>
<head>
<title>My title</title>
</head>
<body>
  <a href="">My link</a>
  <h1>My header</h1>
</body>
</html>
```



HTML DOM

- Relacionamento entre nós numa árvore DOM



HTML DOM - API

- É definida por um conjunto de propriedades e métodos
- Algumas propriedades DOM
 - x.**innerHTML**: o valor text de x
 - x.**nodeName**: o nome do elemento x
 - x.**nodeValue**: o valor de x
 - x.**nodeType**: o tipo de x (*1 – elemento; 2 – atributo; 3 – texto; ...*)
 - x.**parentNode**: o nó pai de x
 - x.**childNodes**: os nós filhos de x
 - x.**attributes**: os nós atributos de x

HTML DOM - API

- Alguns métodos DOM
 - x.**getElementById(*id*)**: obtém o elemento com o *id* fornecido
 - x.**getElementsByTagName(*name*)**: obtém todos os elementos com a tag *name*
 - x.**appendChild(*node*)**: insere um nó filho *node* em x
 - x.**removeChild(*node*)**: remove o nó filho *node* de x



HTML DOM – API

Exemplo

```
<html>
<body>
  <p id="intro">Hello World!</p>

  <script type="text/javascript">
    txt=document.getElementById("intro").innerHTML;
    document.write("<p>Texto do parágrafo intro: " + txt + "</p>");
  </script>
</body>
</html>
```

HTML DOM – API

Exemplo

```
<html>
<body>
  <p id="intro">Hello World!</p>

  <script type="text/javascript">
    txt=document.getElementById("intro").childNodes[0].nodeValue;
    document.write("<p>Texto do parágrafo intro: " + txt + "</p>");
  </script>
</body>
</html>
```



JavaScript – Mais Usos

- Criação de cookies
- Validação de entrada
- Disparo de funções com retardo (tempo)
- Criação de objetos próprios

HTML – Como funciona o envio desta mensagem?

```
<body>
```

```
<form action="processaForm.jsp" >
```

```
Nome: <input type="text" name="firstname"><br />
```

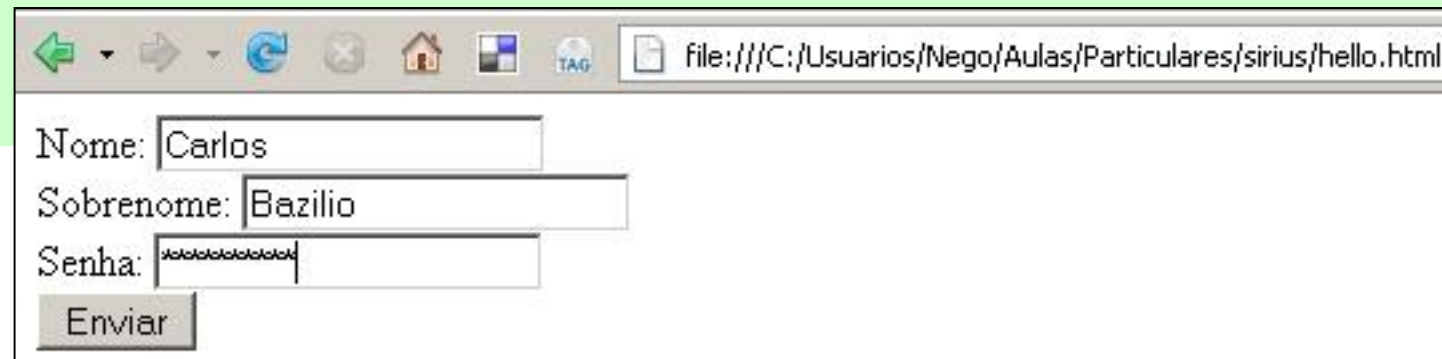
```
Sobrenome: <input type="text" name="lastname"><br />
```

```
Senha: <input type="password" name="senha">
```

```
<input type="submit" value="Enviar">
```

```
</form>
```

```
</body>
```



The screenshot shows a web browser window with the address bar displaying 'file:///C:/Usuarios/Nego/Aulas/Particulares/sirius/hello.html'. The browser's toolbar includes back, forward, home, and search buttons. The main content area displays a form with three input fields: 'Nome:' containing 'Carlos', 'Sobrenome:' containing 'Bazilio', and 'Senha:' containing masked characters. Below the fields is a button labeled 'Enviar'.

- URL após clicar no botão: .../sirius/processaForm.jsp?firstname=Carlos&lastname=Bazilio&senha=abcdefg



Tipos de requisições HTTP

- GET
 - Parâmetros são enviados na url da solicitação
 - <http://localhost:8080/sirius/processaForm.jsp?firstname=Carlos&lastname=Bazilio&senha=abcdefg>
 - Envio de parâmetros *firstname* e *lastname* e *senha* para um recurso no servidor (neste caso, uma página jsp – processaForm.jsp)
 - Estas urls podem ser armazenadas como favoritos
 - Tipo padrão (*default*)
 - Não recomendado para o envio de informações sigilosas



Referências

- <http://www.w3schools.com/>
 - Site com tutoriais on-line rápidos e com muita qualidade
- <http://www.csszengarden.com/>
 - Site que demonstra as potencialidades de CSS
- <http://del.icio.us/carlosbazilio/{css+html}>
 - Meus favoritos sobre o assunto
- <http://www.w3.org/>
 - Site do consórcio W3C