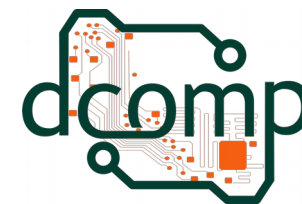




Universidade Federal do Espírito Santo  
Centro de Ciências Agrárias – CCA UFES  
Departamento de Computação



# JavaScript

**Desenvolvimento de Sistemas para WEB**

Site: <http://jeiks.net>

E-mail: [jacsonrcsilva@gmail.com](mailto:jacsonrcsilva@gmail.com)

# Visão Geral

- JavaScript:
  - É uma linguagem em formato *script*;
  - Permite dinamizar funções do site e criar interatividade em páginas web.
- O JavaScript permite:
  - Coletar dados dos formulários HTML e processar suas informações no lado do cliente (sem a necessidade do servidor);
  - Criar e armazenar dados no computador do usuário utilizando *cookies*;
  - Adicionar interatividade a gráficos;
  - Modificar elementos da páginas dinamicamente através da entrada do usuário.

# História

- **A Netscape Corporation inicialmente desenvolveu uma linguagem chamada LiveScript;**
  - Fornecia capacidades de tanto para o navegador quando para o servidor
- **Quando o suporte para Applets Java foi adicionado ao Navigator 2:**
  - Foi assinado um contrato com a Sun Microsystems e nomearam o LiveScript como JavaScript
  - A renomeação de LiveScript para JavaScript deveu-se principalmente ao Java, devido à popularidade do que qualquer semelhança entre as duas linguagens.
  - A versão original do JavaScript, também chamada de LiveScript, foi nomeada JavaScript 1.0.
  - Suportada pelo Navigator 2.0 e Internet Explorer 3.0.
- **A Netscape apresentou o JavaScript 1.1 com o Navigator 3:**
  - Também suportado pelo Internet Explorer 4.0;
  - Características: suporte para mais objetos dos navegadores e funções do usuário.
- **A Netscape apresentou o JavaScript 1.2 com o Navigator 4:**
  - Parcialmente suportado pelo Internet Explorer 4.0;
  - Adicionou novos objetos, propriedades e métodos para trabalhar com folhas de estilos, camadas, expressões regulares e scripts assinados.

# História

- Padronização:
  - A Netscape e a Microsoft submeteram suas linguagens de *script* para serem padronizadas na *European Computer Manufacturers Association* (ECMA);
  - Baseada nas melhores características do JavaScript e do Microsoft Jscript, a ECMA criou o padrão ECMA-262;
  - O padrão ficou conhecido como padrão ECMAScripting.
  - Baseado nesse padrão, a Netscape criou uma nova versão, o JavaScript 1.3, suportado pelo Navigator 4.06 e 4.5.

# JavaScript e a nova Web

- A Web moderna é construída com três tecnologias:
  - HTML:
    - Conteúdo e estrutura;
  - CSS:
    - Apresentação do conteúdo;
  - JavaScript:
    - Interatividade e aplicações avançadas.
- JavaScript é *clientside*:
  - Executa no computador do cliente e não no servidor da web;
- Por ser interpretado, pode ser escrito e lido em um editor de textos, como HTML e CSS.

# JavaScript e a nova Web

- Ele renasceu nos dias atuais, especialmente com o HTML5.
- Os navegadores mais modernos renovaram seus motores para executar o JavaScript mais rapidamente.
- O HTML5 fornece suporte para melhorias que afetam o HTML5:
  - Interatividade do usuário,
  - Bancos de dados,
  - Armazenamento local,
  - Aplicações offline,
  - Geolocalização,
  - Áudio e manipulação de vídeo, e
  - Desenhos com a nova tag <canvas>.

# Segurança

- O JavaScript está disponível do lado do cliente.
- Vulnerabilidade:
  - Se o servidor tiver informações de segurança no código do JavaScript;
- A pessoa pode manipular a informação e modificar campos, pois tudo encontra-se do seu lado.
- Essa é uma característica do JavaScript, por isso, deve-se tomar cuidado ao utilizá-lo.

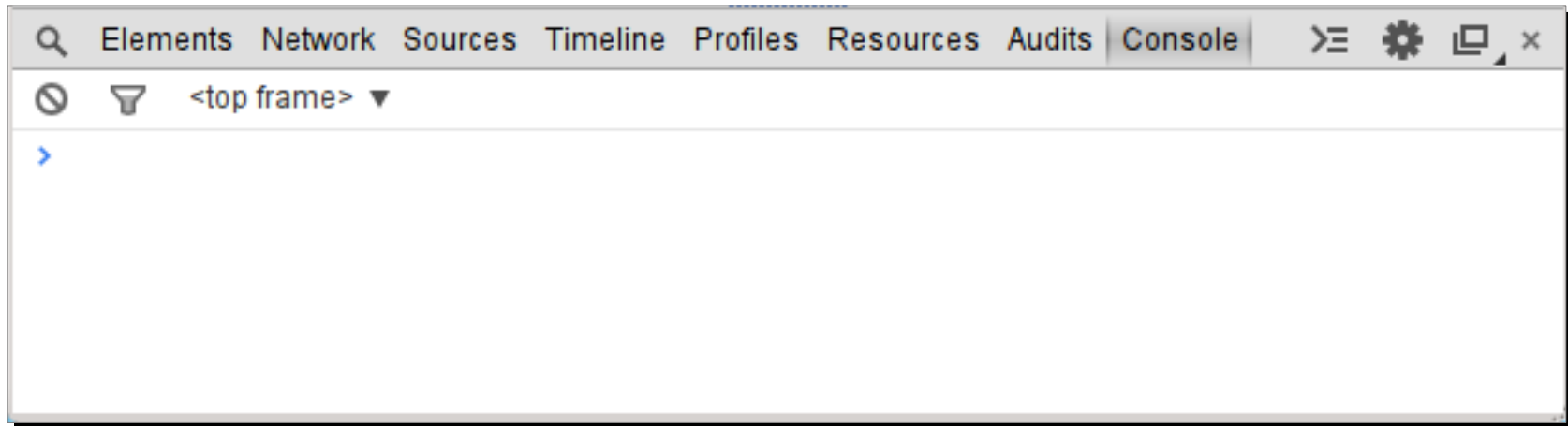
# O JavaScript permite

- Pode detectar o tipo de navegador utilizado.
- Pode mudar propriedades da página.
- Pode-se obter valores de formulários.
- Obter valores do usuário (com prompt).
- Criar animações.
- Efetuar operações do lado do cliente.
- Utilizar Ajax.

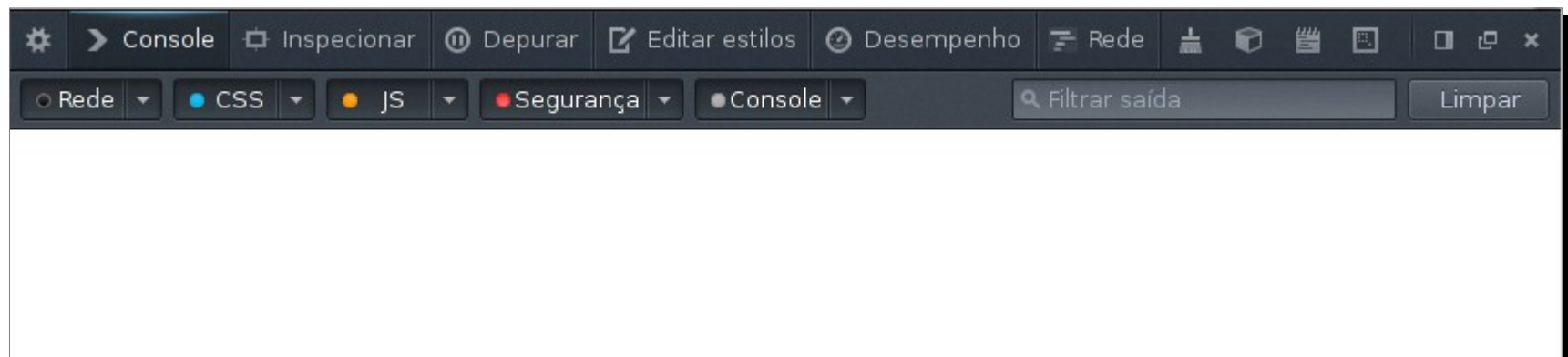


# JavaScript direto no Navegador

- No Chromium ou no Google Chrome:  
CTRL+SHIFT+J



- No Firefox: CTRL+SHIFT+K



# Primeiros comandos

```
alert('Oi Mundo');
```

*Cuidado ao utilizar:  
o JavaScript é case-sensitive.*

```
var nome = 'Jeiks';
```

```
alert("Olá " + nome + ", seja bem vindo!");
```

```
var idade = window.prompt('Qual sua idade?');
```

```
if (idade >= 18) alert('Você é de maior!');
```

```
else alert('Você é de menor!');
```



# Formas de adicioná-lo ao HTML

- Pode ser colocado em um arquivo separado e inserido no HEAD da página:

```
<script src="seu_script.js"
      type="text/javascript"
      charset="utf-8"></script>
```

- Ou pode ser adicionado dentro da própria página:

```
<script type="text/javascript" charset="utf-8">
    //código JavaScript bem escrito aqui
</script>
```

- Ao ser adicionado dentro da própria página, o ideal é adicioná-lo dentre as *tags* HEAD.
  - Alguns navegadores também aceitam sua inserção dentro do BODY.

# Primeira página com JS

```
<html>
  <head>
    <script type="text/javascript">
      document.writeln(
        "<h1>Iniciando com JavaScript!</h1>");
    </script>
  </head>
  <body></body>
</html>
```

# Obtendo valores de elementos com JavaScript

```
<body>
```

```
<input id="nome" value="digite seu nome" />
```

```
<input type="button"
```

```
onclick="javascript:alert('oi '+nome.value) "
```

```
value="clique aqui"
```

```
/>
```

```
</body>
```

# Exemplos

- Utilizando o JavaScript para obter a data:

```
<h1>Hoje é  
  <script type="text/javascript">  
    document.write(Date());  
  </script>  
</h1>
```

# O que fazer quando o navegador não suporta JavaScript

`<h1>Hoje é:`

```
<script type="text/javascript">  
document.write(Date())  
</script>
```

**`<noscript>`**

Não disponível, pois o JavaScript está desabilitado em seu navegador. Por favor, habilite-o e recarregue essa página para ver a hora e o tempo.

**`</noscript>`**

`</h1>`

# Considerações sobre variáveis

- Exemplos:
  - **function test(){  
    var msg = "Oi"; // variável local  
}**  
**alert(msg); // dará erro!**
  - **function test(){  
    msg = "hi"; // variável global  
}**  
**alert(msg); // assim funciona**
- Em mais de uma variável, utiliza-se vírgula:  
**var msg = "oi", encontrado = false, idade = 27;**



# Considerações sobre variáveis

- Tipos de dados:
  - "undefined" valor de undefined;
  - "boolean" valor de Booleanos;
  - "string" valor de String;
  - "number" valor de números;
  - "object" valor de objeto ou null;
  - "function" valor de uma função.

```
var msg = "some string";  
alert( typeof msg );           // "string"  
alert( typeof(msg) );         // "string"  
alert( typeof 95 );           // "number"
```

# Estruturas

## Condicionais:

```
if (variavel > 5)
{ //código }
else if (variavel < 0)
{ //código }
else
{ //código }
```

## Laços:

### while

```
var i=0;
while(i < 5)
{ alert(i++); }
```

### for

```
for (var i=0; i<5;i++)
{ alert(i); }
```

# Estruturas

## Seleção Múltipla:

```
switch( variavel )
{
    case "1":
        //comandos
        break;
    case "2": //comando
        break;
    default:
        //comandos
}
```

## Laços:

### do ... while

```
var i=0;
do{
    alert(i++);
}while(i < 5);
```

As instruções **break** e **continue** também podem ser utilizadas para alterar o fluxo de controle.

# Arrays

- Como criar e iniciar um array:

```
var vet = new Array( 5 );  
for (var i = 0; i<5 ; i++)  
    vet[ i ] = i; //atribuindo valores
```

Os Arrays do JavaScript podem receber qualquer tipo de variável.

----- OU -----

```
var vet = new Array();  
// criando os elementos e atribuindo os valores:  
for (var i = 0; i<5 ; i++)  
    vet[ i ] = i;
```

----- OU -----

```
var vet = new Array( 1, 2, 3, 4, 5 );
```

# Funções

- O formato de uma definição de função é:

```
function nome da função (lista de parâmetros)
{
    declarações e instruções
    return valor; //opcional
}
```

- Nome da função: qualquer identificador válido;
- Lista de parâmetros: lista separada por vírgula que contém os nomes dos parâmetros recebidos pela função;
- Declarações e instruções: fazem parte do escopo da função;
- Retorno: sendo opcional, retorna um valor da função.
- Observação: termine cada instrução com um ponto-e-vírgula.

# Exemplo

```
<head>
```

```
  <script type="text/javascript">
```

```
    function escreverSequencia() {
```

```
      var max = window.prompt("Digite o Max:");
```

```
      max = parseInt(max);
```

```
      for ( var i=1 ; i <= max ; i++)
```

```
        sequencia.innerHTML += ' ' + i;
```

```
    }
```

```
  </script>
```

```
</head>
```

```
<body>
```

```
  <input type="button" value='Escrever'
```

```
    onclick="escreverSequencia()" />
```

```
  <div id="sequencia"></div>
```

```
</body>
```

# Exemplo

- Adicionar no head:

```
<script type="text/javascript">  
function obterDiaDeHoje() {  
    var data = new Date();  
    var mes = data.getMonth()+1;  
    var dia = data.getDate();  
    var ano = data.getFullYear();  
    var dataCompleta = dia + "/" + mes + "/" + ano;  
    CampoData.value = dataCompleta;  
}  
</script>
```

- E no body:

```
<body onload="obterDiaDeHoje()">  
    <input id="CampoData" />  
</body>
```

# Eventos

- Outros manipuladores de eventos existem além do onload.
- Esses manipuladores adicionais dependem da ação do usuário
- Os manipuladores de eventos principais incluem:
  - onclick;
  - onmouseover;
  - onmouseout;
  - onblur;
  - onfocus;
  - Entre outros...



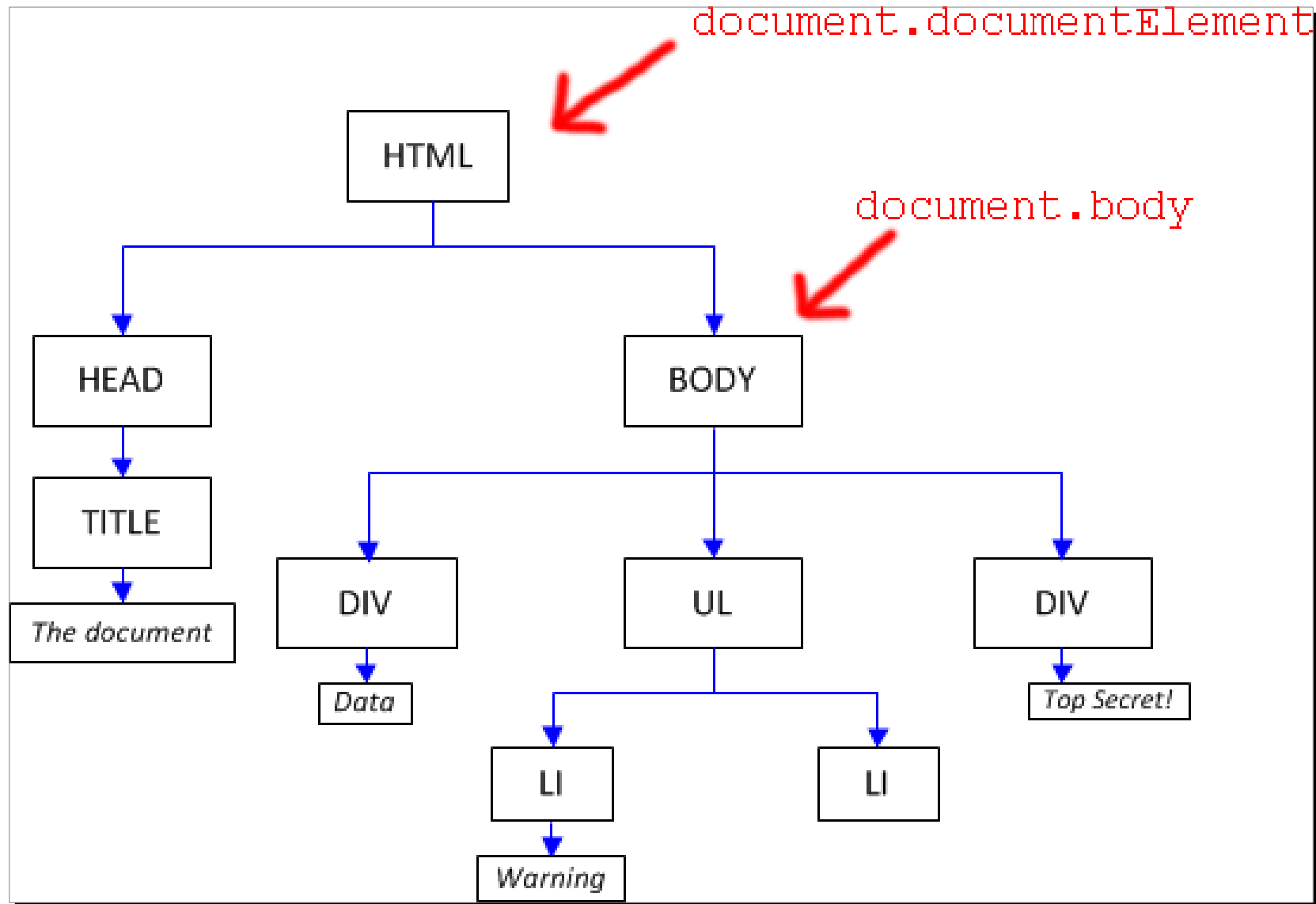
# Objetos

- O JavaScript também possui objetos como:
  - Math (cálculos matemáticos comuns);
  - String (operações com cadeias de caracteres);
  - Date (manipulações de data e hora);
  - Boolean e Number;
  - document (manipulador da página aberta);
  - window (manipulador da janela do navegador).
- Visualize seus métodos/componentes no próprio navegador:
  - Abra o console do navegador, digite o **nome** do *objeto* e um **ponto** logo em seguida.

# Modelo de Objeto de Documento DOM

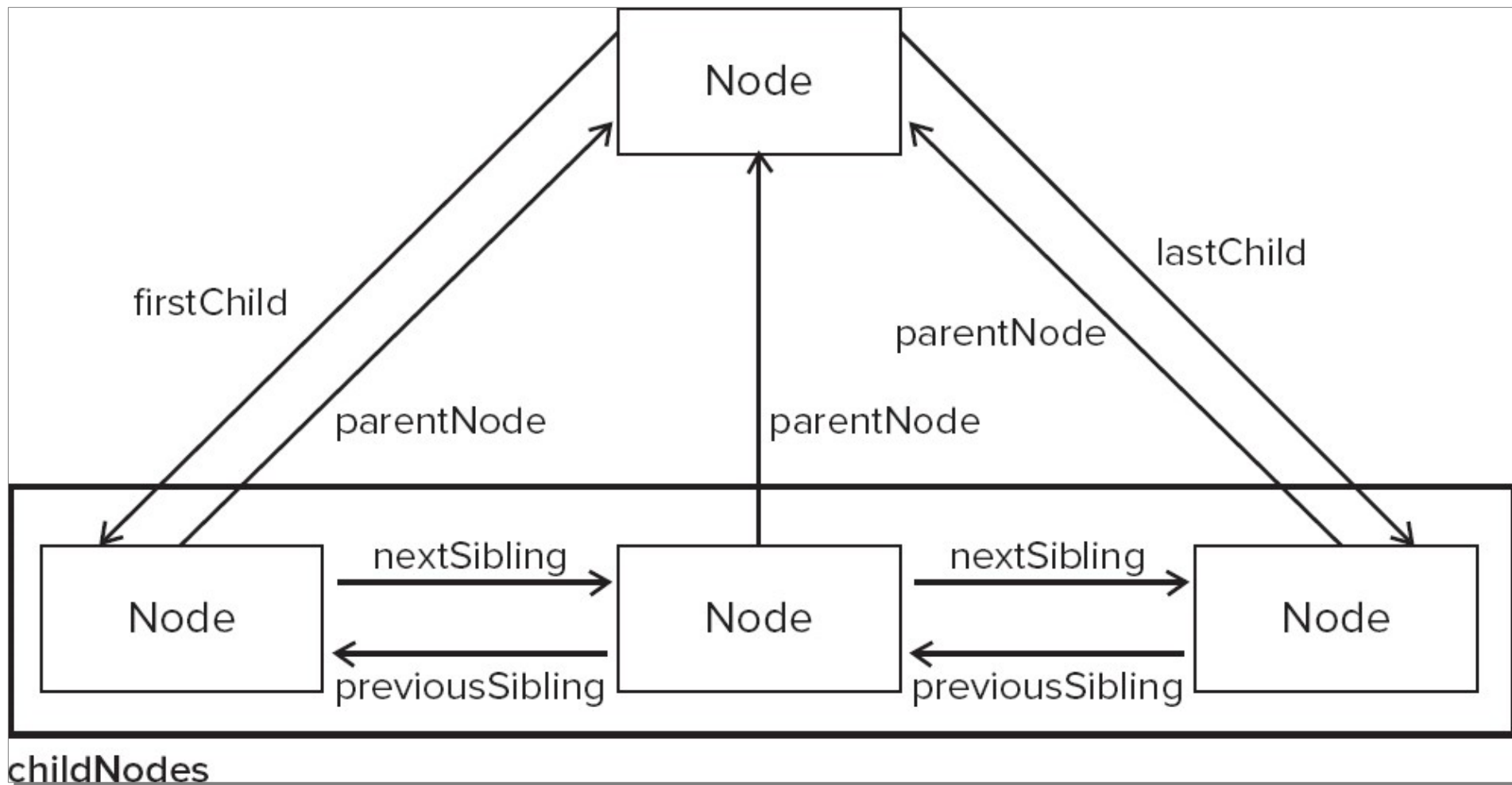
- Dentro de um navegador, a página Web inteira é representada por uma **hierarquia de objetos**.
- Com JavaScript, o programador:
  - Tem acesso a todos os elementos da página;
  - Pode criar, modificar e remover dinamicamente os elementos da página.
- A W3C criou o DOM para ser padronizado e utilizado por todos os navegadores.
- Como a hierarquia do DOM é em árvore, cada um dos seus nós é relacionado como um **Node**.

# DOM – Hierarquia de Objetos



Abra uma página na Web e acesse seu DOM pelo console do navegador

# Percorrendo os Nodes



Percorra a hierarquia de uma página pelo console do navegador.  
`document.childNodes`

# Um pouco de HTML5 e JavaScript

# Classes

- Obter todos os elementos que sejam da classe “escolhido”:

```
var osEscolhidos =  
    // retorna um Array:  
    document.getElementsByClassName("escolhido");
```

- Obter todos os elementos que sejam da classe “escolhido”, porém que também estejam dentro do elemento com id “meuDIV”:

```
var osEscolhidos =  
    // retorna um Array:  
    document.getElementById("meuDiv").  
    getElementsByClassName("escolhido");
```

# Tags

- Obter os elementos pelo nome da tag

```
var elementos = document.getElementsByTagName( 'p' );  
// retorna um Array com os elementos que são <p ...>
```

- Obtendo valores dos elementos, como sua classe:

```
elementos[0].className
```

# Característica de Foco

- Definindo o foco:

```
var botao = document.getElementById("botao");  
botao.focus();
```

- Confirmando qual elemento tem o foco:

```
document.activeElement
```

- Verificando se o documento ou um elemento dele possui foco:

```
document.hasFocus()
```

- E definindo a visibilidade de um elemento:

```
botao.scrollIntoView()
```



# Verificando a codificação

```
document.charset = "latin1";  
// ou: document.charset = "utf-8";  
alert( document.charset )  
alert( document.defaultCharset )
```

# Adicionando novos atributos

- Novos atributos podem ser adicionados no HTML5. Para isso, deve-se utilizar o prefixo “data-”. No HTML, faça:

```
<div id="meuDiv"  
    data-Identificacao="123456"  
    data-meuNome="Jacson">  
  
</div>
```

- Para obter os valores com JavaScript:

```
var identificacao = meuDiv.dataset.Identificacao;  
var meuNome = meuDiv.dataset.meuNome;  
meuDiv.dataset.Identificacao = 23456;  
meuDiv.dataset.meuNome = "Jeiks";
```