

<http://cseweb.ucsd.edu/~lerner/js.jpg>

# JAVASCRIPT

# HTML, CSS, Javascript



- HTML define estrutura
- CSS define estilo
- **Javascript define lógica de operações**

# O que é?



- Linguagem de Programação
  - ▣ Interpretada
  - ▣ Pode ser compilada
  - ▣ "Scripting"
    - (leve, supostamente para pequenos scripts)
  
- Muito utilizada em páginas web
  - ▣ Interatividade
  - ▣ Animações

# O que pode fazer?



- Permite programação numa página HTML
  - ▣ Ciclos, condições, funções, etc...
- Pode reagir a eventos
  - ▣ Ex: carregar num botão do rato
- Pode alterar HTML
  - ▣ Ex: carregar mais comentários
- Etc...

# Origem na Netscape



- Desenvolvido para o Netscape Navigator em 1995
- Maio: Criada como Mocha
- Setembro: Mudou nome para LiveScript
- Dezembro: Netscape Navigator adicionou suporte para Java
  - ▣ O primo mais simples ficou JavaScript
  - ▣ Objectivo: Ganhar notoriedade pela fama do Java

# Onde colocar Javascript

---

- Usando a marca `<script>`
  - ▣ Em qualquer parte da página.
- Incluída na página
  - ▣ Tipicamente dentro da marca `<head>`
- Obtida de um recurso externo
  - ▣ Método preferencial

# JavaScript na Página

```
<html>
  <head>
    <script>
      ...
    </script>
  </head>
  <body>
    ...
    <script>
      ...
    </script>
  </body>
</html>
```

# JavaScript Externo

```
<html>
  <head>
    <script language="javascript" type="text/javascript"
      src="scripts.js"></script>
  </head>
  <body>
  </body>
</html>
```

- ❑ Maior separação do código
- ❑ Possibilidade de partilha entre páginas
- ❑ Cache no browser



# Caraterísticas (relevantes)

---

- Interpretada

- Pode ser compilada "no momento"
- Erros só são detetados na execução

- Sintaxe semelhante ao C/Java

- Usa ;
- Usa chavetas, parêntesis e parêntesis rectos
- Instruções if, for, switch, do while, while, etc..

# Caraterísticas: Semelhante a C/Java

```
for(i=0;i<10;i++){  
    /* Faz qualquer coisa */  
}
```

```
do{  
    /* Faz qualquer coisa */  
}while(i < 10);
```

```
while(i < 10){  
    /* Faz qualquer coisa */  
}
```

# Caraterísticas: Semelhante a C/Java

```
switch(valor){  
    case 0:  
        /* Faz qualquer coisa para 0 */  
        break;  
  
    case "abc":  
        /* Faz qualquer coisa para "abc" */  
        break;  
  
    default:  
        /* Faz qualquer coisa */  
}
```

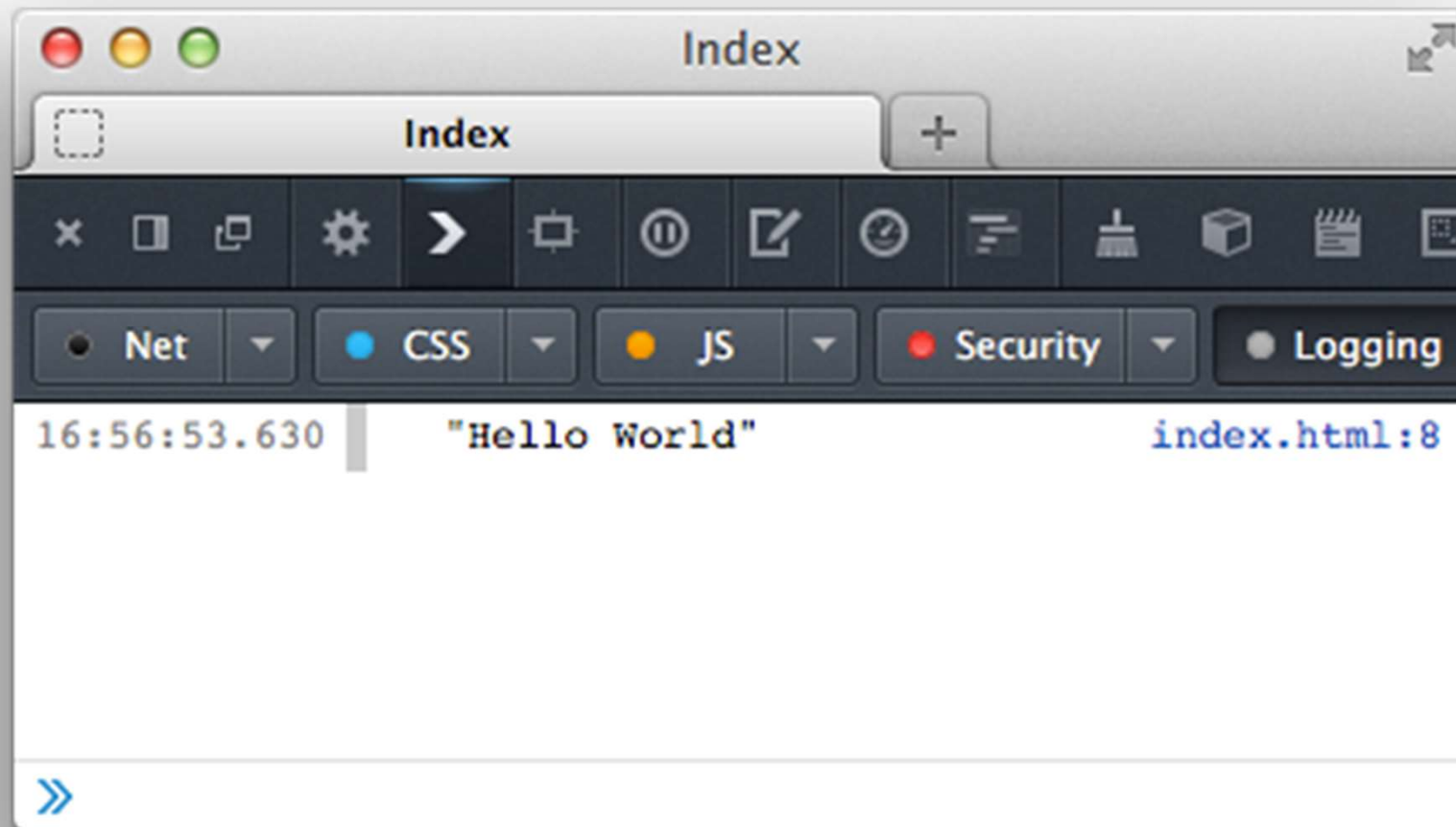
# Caraterísticas: OOP

---

- Orientada a Objetos
  - ▣ Sintaxe no estilo objeto.método()
- Exemplo: Escrita para a consola
  - ▣ Do browser!

```
<script>  
  console.log("Hello World");  
</script>
```

# Caraterísticas: OOP



# Caraterísticas: Tipos Dinâmicos

- Tipo da variável depende do valor
  - ▣ Não é necessário declarar tipo
  - ▣ Tipo pode mudar em execução

var a;

**Declaração**

a = 3;

**Atribuição**

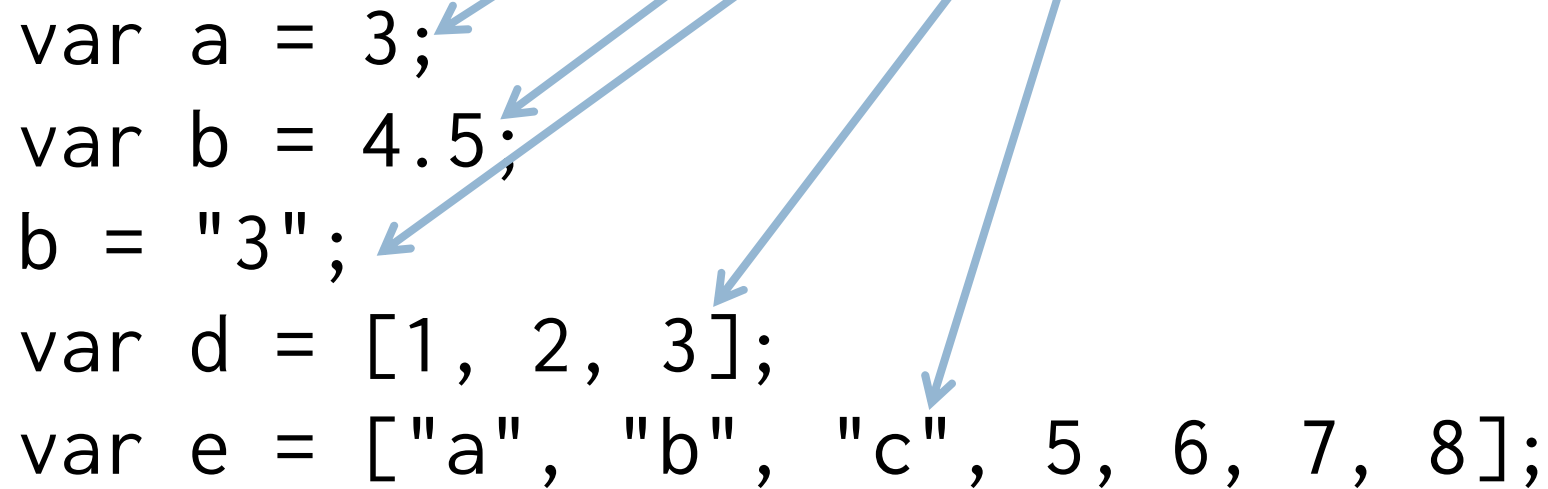
var b = 3;

**Declaração  
e  
Atribuição**

# Caraterísticas: Tipos Dinâmicos

**a é um número inteiro**  
**b é um número real**  
**b passa a ser uma string**  
**d é um array de inteiros**  
**e é um array misto**

```
var a = 3;  
var b = 4.5;  
b = "3";  
var d = [1, 2, 3];  
var e = ["a", "b", "c", 5, 6, 7, 8];
```

A diagram illustrating dynamic typing. It shows five lines of code in a box. Five blue arrows originate from a text box on the right and point to specific parts of the code: the first arrow points to the value '3' in 'var a = 3;', the second points to '4.5' in 'var b = 4.5;', the third points to the string '3' in 'b = "3";', the fourth points to the array '[1, 2, 3]' in 'var d = [1, 2, 3];', and the fifth points to the string 'c' in 'var e = ["a", "b", "c", 5, 6, 7, 8];'.

# Caraterísticas: Funções

## □ Funções sem tipo definido

### ▣ parâmetros opcionais

```
function mostraMensagem(mensagem, valor){  
    console.log(mensagem+" "+valor);  
}
```

```
mostraMensagem("Temperatura", 23);  
mostraMensagem("Hello");
```

**Ambas  
utilizações  
válidas**

**valor será null**



# Caraterísticas: Âmbito da função

- Variáveis podem:
  - ▣ Ser globais
  - ▣ Pertencer a uma função

```
var a = "global";  
function mostraMensagem(b){  
    var c = "local";  
    console.log(a+" "+b+" "+c);  
}  
  
mostraMensagem("argumento");
```

# Popups



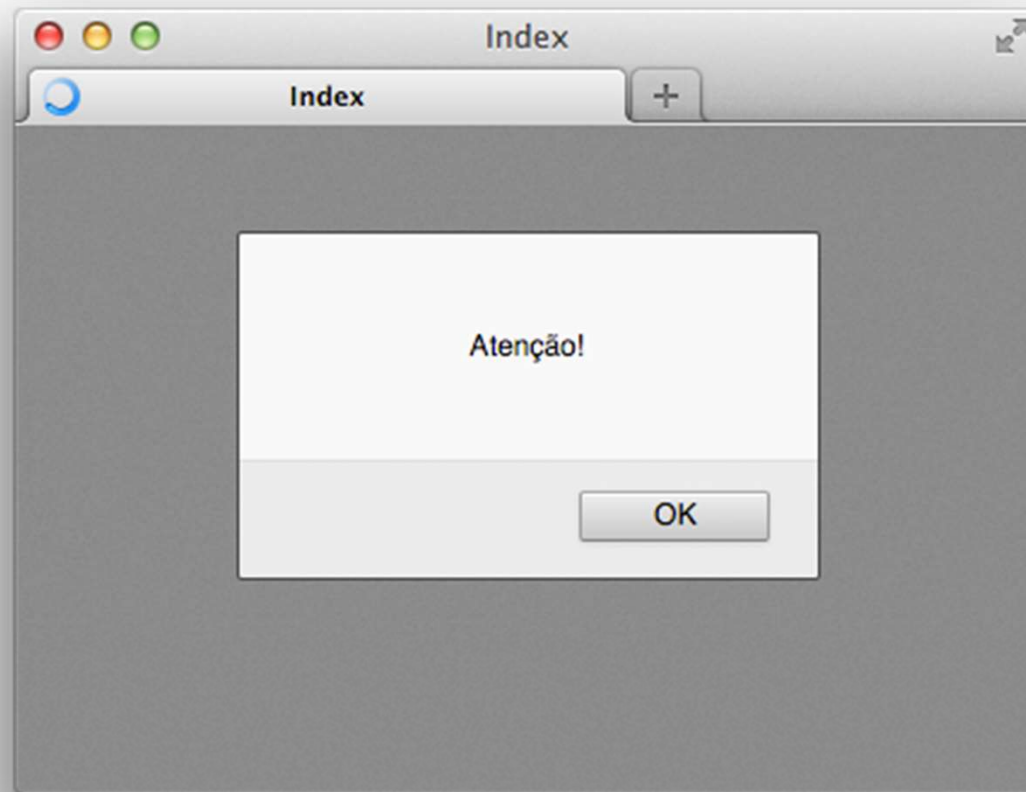
- ❑ JavaScript pode modificar qualquer conteúdo da página!
- ❑ Pode mostrar alertas/popups

```
alert("Atenção!");
```

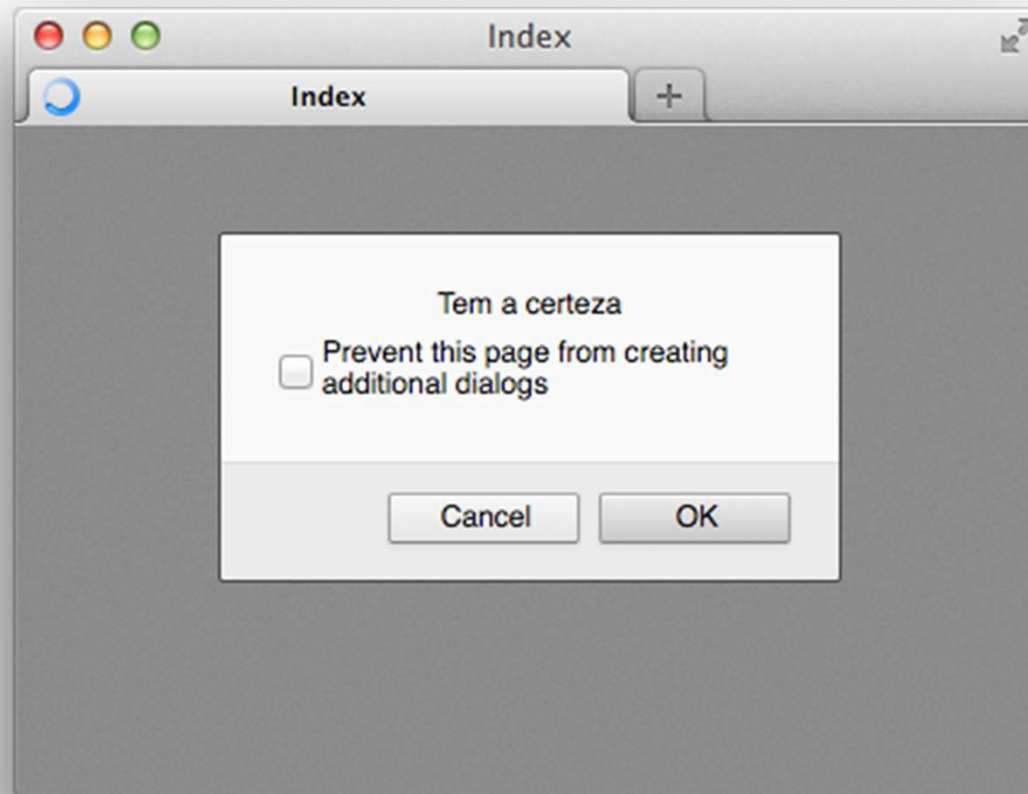
```
confirm("Tem a certeza");
```

```
prompt("Insira o seu nome");
```

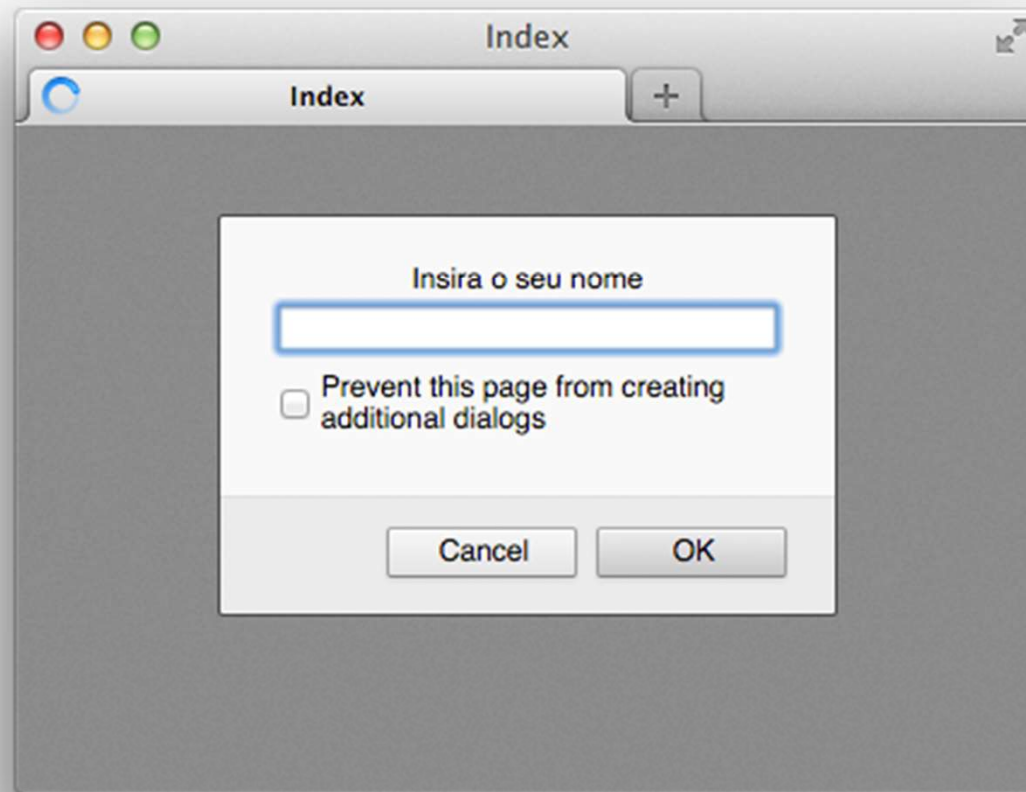
# Alert



# Confirm



# Prompt



# DOM



- Document Object Model
  - ▣ Representa elementos HTML em JS
- Permite que o JS acesse/altere qualquer elemento HTML
- Propriedade **id** é extremamente útil
  - ▣ Permite localizar elementos

[http://www.w3schools.com/jsref/dom\\_obj\\_document.asp](http://www.w3schools.com/jsref/dom_obj_document.asp)

# DOM: getElementById

```
<body>
```

```
  <h1 id="texto">HTML</h1>
```

```
  <script>
```

```
    var e = document.getElementById("texto");
```

```
    e.innerHTML = "JS";
```

```
    e.style.backgroundColor = "#444";
```

```
    e.style.color = "orange";
```

```
  </script>
```

```
</body>
```

# DOM: getElementById (JS desligado)

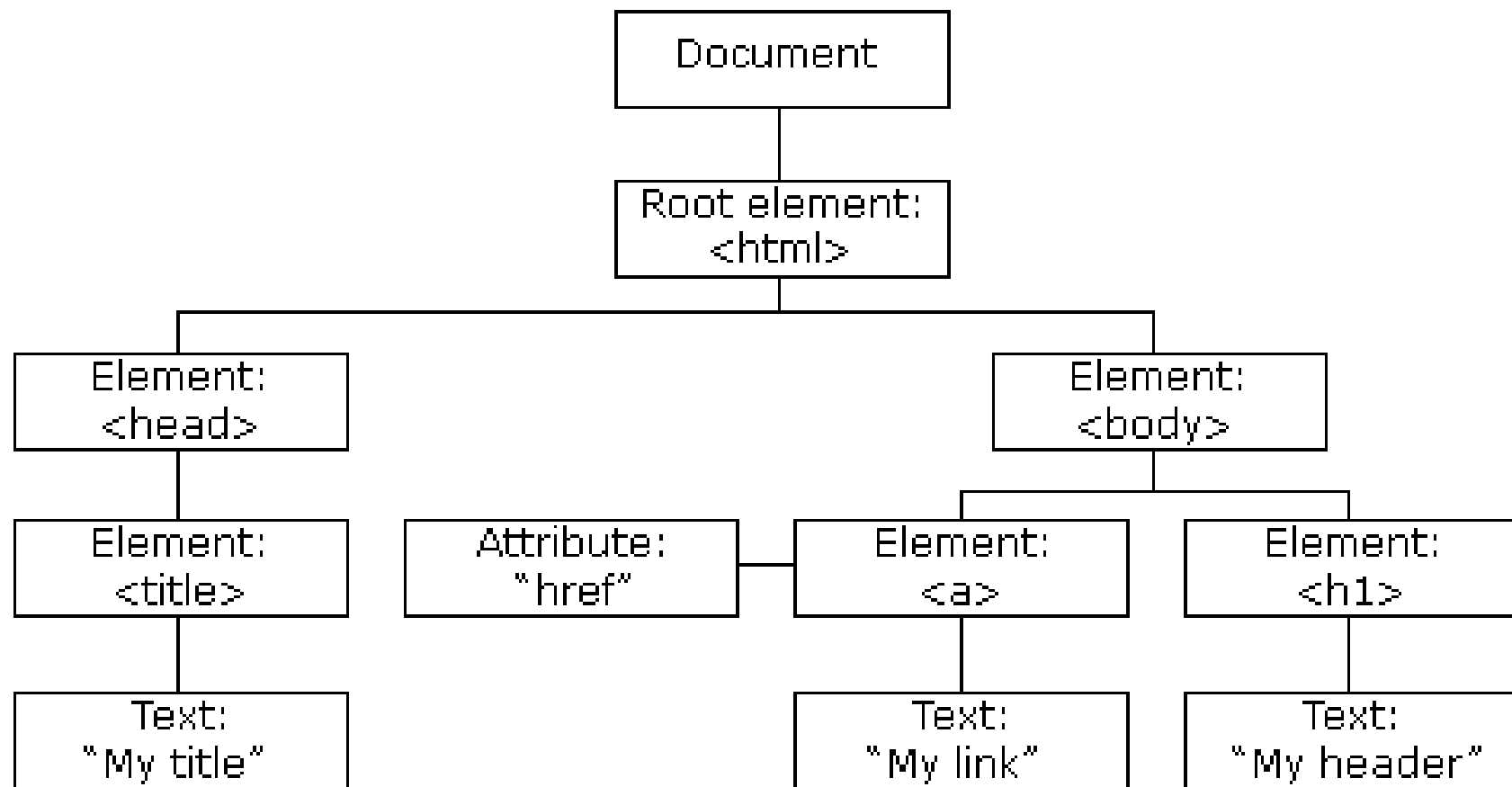




# DOM: getElementById (JS ligado)



# DOM



# Eventos

- Possível ligar funções a eventos

- Exemplo:

- Página terminada de carregar

- Clique do apontador

- Redimensionamento do browser

```
<h1 nome_evento="chamar-função()"></h1>
```

# Eventos

```
<body>
```

```
  <h1 id="texto">HTML</h1>
```

```
  <script>
```

```
    var e = document.getElementById("texto");
```

```
    e.innerHTML = "JS";
```

```
    e.style.backgroundColor = "#444";
```

```
    e.style.color = "orange";
```

```
  </script>
```

```
</body>
```

**Script declarado  
depois do <h1>**

**Funciona!**

# Eventos

```
<body>
```

```
  <script>
```

```
    var e = document.getElementById("texto");
```

```
    e.innerHTML = "JS";
```

```
    e.style.backgroundColor = "#444444";
```

```
    e.style.color = "orange";
```

```
  </script>
```

```
  <h1 id="texto">HTML</h1>
```

```
</body>
```

**Script declarado  
antes do <h1>**

**h1 não existe  
quando código  
é executado**

**Não funciona!**

# Eventos



# Eventos: OnLoad

```
<head>
  <script>
    function change(){
      var e = document.getElementById("texto");
      e.innerHTML = "JS";
      e.style.backgroundColor = "#444";
      e.style.color = "orange";
    }
  </script>
</head>

<body onload="change()">
  <h1 id="texto">HTML</h1>
</body>
```

**Função chamada  
depois da página  
ter sido carregada.**

# Eventos: Apontador

---

- onmouseover: apontador passou por cima do elemento
- onclick: apontador clicou no elemento
- onmouseout: apontador saiu de cima do elemento
- Etc...

[http://www.w3schools.com/jsref/dom\\_obj\\_event.asp](http://www.w3schools.com/jsref/dom_obj_event.asp)



# Exemplo

<http://jsfiddle.net/q2Qaa/>

# Para Referência



- Validador de JS, <http://www.jshint.com>
- JSFiddle, <http://jsfiddle.net>
  - ▣ Exemplo eventos: <http://jsfiddle.net/q2Qaa/>
- W3 Schools Javascript Tutorials, <http://www.w3schools.com/js/>