



# Introdução ao Javascript

Parte 3

Prof. Victor Hugo Lopes



# agenda

Trabalhando com Eventos

DOM na prática: trabalhando com propriedades

<http://goo.gl/tKJhMB>



# Eventos

**Evento** é um acontecimento que ocorre a partir de uma ação do usuário. Por exemplo, quando o usuário move o cursor do mouse por sobre uma div específica, ocorre um evento.

Além deste tipo de evento, alguns eventos também são criados pela ação de outros elementos, como o completo carregamento dos arquivos do website pelo navegador.

Os eventos são identificados por propriedades de tags HTML, mas nem todas as tags possuem eventos.

Importante notar que é o objeto que possui eventos “capturáveis”.

```
<a href="#" onClick="alert('Link foi clicado!');">Clique aqui</a>
```



# Eventos

`onClick`: evento após um click com o mouse no objeto

`onDbClick`: evento do duplo click com o mouse no objeto

`onMouseOver`: evento de passar o mouse sobre o objeto

`onMouseOut`: evento iniciado quando o mouse abandona o objeto

`onMouseDown`: iniciado quando o botão principal é clicado

`onMouseUp`: iniciado quando o botão principal é solto após clicado

`onKeyDown`, `onKeyPress` e `onKeyUp`: baseados nas teclas, quando se clica, mantém pressionada e quando solta, respectivamente.

`onLoad`: quando a página foi carregada

`onBlur`: quando um objeto perde o foco

`onFocus`: quando um objeto ganha o foco

`onSubmit`: quando um formulário é enviado

`onReset`: quando um formulário é resetado

# Eventos

## Exercitando:

Crie um novo arquivo html, com a estrutura básica:

```
<html>  
  <head>  
    <meta charset="utf-8" />  
  </head>  
  <body>  
  </body>  
</html>
```

# Eventos

## Exercitando(localhost/pdi1):

Crie 4 divs de cores diferentes, chamadas caixa1, caixa2, caixa3 e caixa4, com tamanho 100px por 100px, e que tenha o cursor pointer.

```
<html>
<head>
  <meta charset="utf-8">
  <style>
    #caixa1{
      width:100px; height:100px; background-color:red; position: absolute; left:10px; top:10px; cursor:pointer;
    }
    #caixa2{
      width:100px; height:100px; background-color:blue; position: absolute; left:10px; top:120px; cursor:pointer;
    }
    #caixa3{
      width:100px; height:100px; background-color:green; position: absolute; left:120px; top:10px; cursor:pointer;
    }
    #caixa4{
      width:100px; height:100px; background-color:cyan; position: absolute; left:120px; top:120px; cursor:pointer;
    }
  </style>
</head>
<body>
</body>
</html>
```

# Eventos

## Exercitando:

Crie um script JS na seção head:

```
<script type="text/javascript">  
    var controle = "";  
    var cont = 0;  
</script>
```

Crie o método onClick em cada div criada:

```
<div id="caixa1" onClick="controle = controle + ' caixa 1 '; cont++;  
if(cont == 4){ alert(controle); cont=0; controle='';}"></div>
```

# DOM na prática

Bom, agora que já se sabe as técnicas básicas de escrita de um algoritmo em javascript, e que já sabemos manipular minimamente os eventos dos objetos de uma página html, agora se faz necessário obter conhecimentos de como manipular os próprios objetos.



# DOM na prática

Imagine que em diversos tipos de programas, é necessário não somente saber o que ocorre com as ações dos usuários e objetos do programa, mas também é essencial se ter condições de manipular determinados objetos à partir de um script.

De forma análoga, em um jogo eletrônico, a ação de clicar em determinado botão deve ser capaz de iniciar diversas ações, como mover o personagem para determinada posição, movendo seus membros, simulando uma corrida, por exemplo.

# DOM na prática

## Manipulando uma caixa de textos

O elemento input text, que é nossa boa e velha caixa de textos, pode ser manipulada, isto é, podemos alterar suas propriedades.

Um script que queira capturar o valor de um input em um formulário, utiliza objetos e métodos do DOM para tal. Veja a sintaxe:

```
document.getElementById('idDoInput').value
```

Explicando: o objeto básico document é referenciado, onde buscamos o método de capturar um elemento pelo seu ID, onde buscamos a propriedade value deste elemento.

# DOM na prática

Manipulando uma caixa de textos

Exemplificando:

```
<script type="text/javascript">  
    function pegaValor(){  
        V = document.getElementById('valor').value;  
        alert(V);  
    }  
</script>
```

```
<input type="text" name="valor" id="valor" />
```

```
<input type="button" value="Pegar Valor" onClick="pegaValor();" />
```

# DOM na prática

Manipulando uma caixa de textos

Exemplificando:

```
<script type="text/javascript">  
    function preencheValor(val){  
        document.getElementById('valor').value = val;  
    }  
</script>
```

```
<input type="text" name="valor" id="valor" />
```

```
<input type="button" value="Preencher"  
onClick="preencheValor(100);" />
```



# DOM na prática

Exercitando:

E agora, dá pra desenvolver uma calculadora decente??

Tente começar com as 4 operações básicas, e que haja um campo de texto para receber os valores e operadores.

789+1*4				
7	8	9		
6	5	4		
3	2	1		
0	+	-	*	/
total				

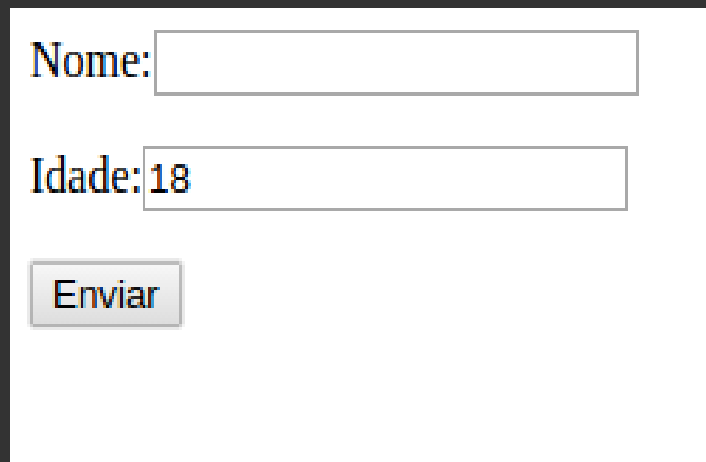
# DOM na prática

## Trabalhando com formulários

Um bom uso para JavaScript é no controle de formulários, garantindo, por exemplo, a validação do preenchimento de campos.

Imagine o exemplo de um simples formulário onde se deseja garantir que se digitem valores de forma correta.

Para tal, precisamos saber o que fazer em determinados eventos disparados pelo usuário.



A screenshot of a simple web form. It contains two text input fields. The first field is labeled 'Nome:' and is empty. The second field is labeled 'Idade:' and contains the number '18'. Below the input fields is a button labeled 'Enviar'.

Nome:

Idade:

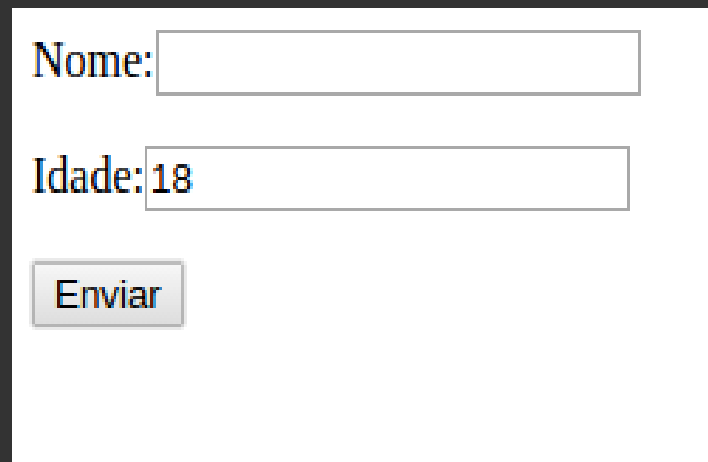
# DOM na prática

## Trabalhando com formulários

Neste caso, precisamos de um evento que indique que o usuário terminou de digitar, para que possamos realizar a validação do valor informado.

Para isso, temos o evento `onBlur`, que indica que o usuário está saindo da caixa de texto.

Neste exemplo, vamos simplesmente garantir que os campos sejam preenchidos.



A screenshot of a web form with a white background. It contains two text input fields. The first field is labeled 'Nome:' and is empty. The second field is labeled 'Idade:' and contains the number '18'. Below the fields is a button labeled 'Enviar'.

Nome:

Idade:

# DOM na prática

## Trabalhando com formulários

O formulário:

```
<form name="formulario" id="formulario" method="post" action="">
```

```
  <p>Nome:<input type="text" name="nome" id="nome" /></p>
```

```
  <p>Idade:<input type="text" name="idade" id="idade" /></p>
```

```
  <input type="button" value="Enviar" id="btnEnviar" />
```

```
</form>
```



# DOM na prática

## Trabalhando com formulários

Alterando cada campo de texto, criando o evento onBlur:

```
<form name="formulario" id="formulario" method="post" action="">
```

```
    <p>Nome:<input      type="text"      name="nome"      id="nome"
onBlur="" /></p>
```

```
    <p>Idade:<input      type="text"      name="idade"      id="idade"
onBlur="" /></p>
```

```
    <input type="button" value="Enviar" id="btnEnviar" />
</form>
```

# DOM na prática

## Trabalhando com formulários

Indicando o comando ou a função que será chamada no evento onBlur:

```
<form name="formulario" id="formulario" method="post" action="">
```

```
    <p>Nome:<input    type="text"    name="nome"    id="nome"
onBlur="checaNome();" /></p>
```

```
    <p>Idade:<input    type="text"    name="idade"    id="idade"
onBlur="checaIdade();" /></p>
```

```
    <input type="button" value="Enviar" id="btnEnviar" />
</form>
```

# DOM na prática

## Trabalhando com formulários

Por fim, basta criar as funções indicadas nos eventos:  
Pensando um pouco, estas funções irão precisar de algum parâmetro?

```
<script type="text/javascript">  
    function checaNome()  
  
    }  
    function checaIdade()  
  
    }  
  
</script>
```

# DOM na prática

## Trabalhando com formulários

Considerando as funções sem parâmetro, vamos construir seu corpo:

```
<script type="text/javascript">
    function checaNome(){
        nome = document.getElementById('nome').value;
        if (nome == ""){
            alert('Preencha corretamente o nome');
            document.getElementById('nome').focus();
        }
    }

    function checaIdade(){
        idade = document.getElementById('idade').value;
        if (idade == ""){
            alert('Preencha corretamente a idade');
            document.getElementById('idade').focus();
        }
    }
</script>
```



# DOM na prática

## Alterando propriedades CSS com javascript

Como considerado anteriormente, o DOM permite alterar e capturar propriedades diversas dos objetos HTML inseridos nos documentos, inclusive as propriedades de estilo.

```
document.getElementById("caixa3").style.backgroundColor="pink";
```

```
document.getElementById("caixa3").style.width="300px";
```

# DOM na prática

## EXERCITANDO:

Construir uma página com um álbum fotográfico, sendo uma grande e as demais em miniatura, onde seja possível clicar na foto miniatura e seja aberto a foto grande.