JavaScript no Navegador

Programação para Internet I

IFB - Tecnologia em Sistemas para Internet

Marx Gomes van der Linden

JavaScript no navegador

- 3 maneiras de executar um script
  - » Em um arquivo externo:

```
<script src="arquivo.js"></script>
```

  - » Em um bloco script no arquivo HTML

```
<script>...</script>
```

  - » Dentro de eventos em tags HTML

```
<button onclick="...">
```

Executando em uma página HTML

```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>Página de teste</title>
</head>
<body> <h1>Testando Javascript</h1> </body>
<script>
    console.log("Teste");
    console.log([10, 20, 30].map(x => x * x));
    console.log({ a: 'Bla', b: 'Ble'});
</script>
</html>
```

Usando **document.write**

```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>Página de teste</title>
</head>
<body> <h1>Testando Javascript</h1> </body>
<script>
    document.write("Teste<br>");
    document.write(
       [10, 20, 30].map(x => x * x), "<br>"
     );
    document.write({ a: 'Bla', b: 'Ble'});
</script>
</html>
```

Eventos

- É possível fazer com que um código Javascript seja executado apenas quando algo acontecer:
  - Objeto clicado
  - Mouse se movendo
  - Texto de formulário alterado
  - Página carregada
  - ...

Eventos

Evento	Tags	Descrição
`onclick`	Todas as visíveis	Pressiona e solta botão do mouse
`onmousedown` `onmouseup`	Todas as visíveis	Pressiona/solta botão do mouse
`onmousemove`	Todas as visíveis	Seta do mouse se move por cima
`onmouseout`	Todas as visíveis	Seta do mouse sai do elemento
`onkeydown` `onkeyup`	`body` e elementos de formulário	Pressiona/solta tecla
`onselect`	`input`, `textarea`	Seleciona texto
`onload`	`body`, `img`	Carrega o elemento

Exemplo

```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>Página de teste</title>
</head>
<body>
    <h1>Testando Javascript</h1>
    <button onclick="console.log('Teste')">
        Clique aqui
    </button>
 </body>
</html>
```

```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8"><title>Página de teste</title>
</head>
<body>
   <h1>Testando Javascript</h1>
   <button onclick="escreveObjeto()">
   Clique aqui</button>
</body>
<script>
        function soma(a,b){ return a + b;  }

        function escreveObjeto(){
            let obj = {
                descricao: "Texto de exemplo",
                numero: soma(12, soma(3,5)),
                lista: [10, 20, 30].map(x => x * x)
            }

            console.log(obj);
        }
</script></html>
```

Posição da tag **script**

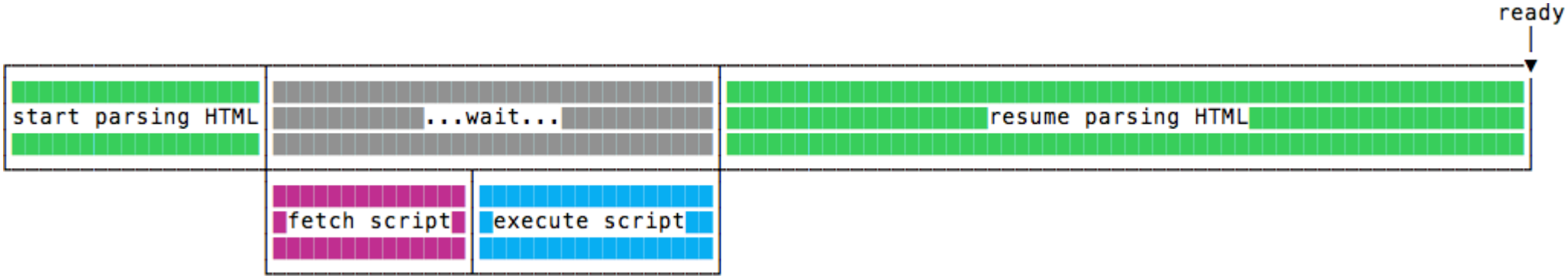
```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>Página de teste</title>
</head>
<body>
    <h1>Testando Javascript</h1>
    <button onclick="escreveObjeto()">
        Clique aqui
    </button>
 </body>
<script src="meucodigo.js"></script>
</html>
```

async/defer

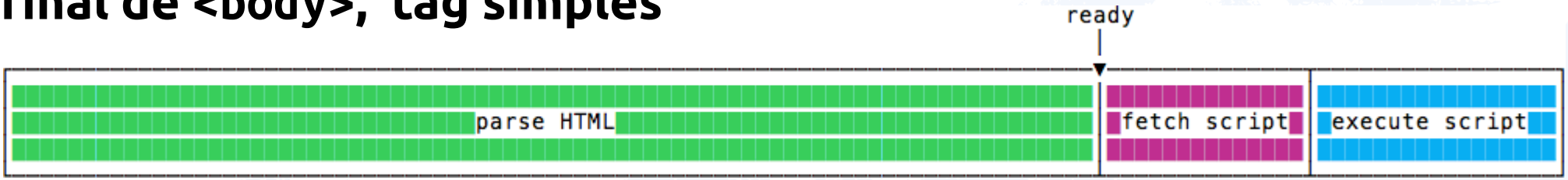
```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>Página de teste</title>
    <script src="meucodigo.js" defer></script>
</head>
<body>
    <h1>Testando Javascript</h1>
    <button onclick="escreveObjeto()">
        Clique aqui
    </button>
 </body>
</html>
```

Onde colocar `<script src=...>` ?

dentro de <head>, tag simples

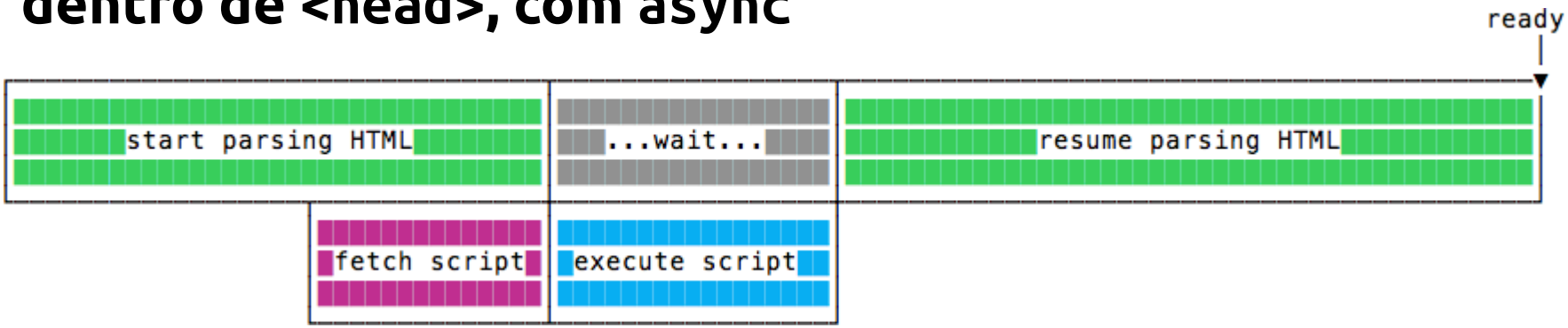


final de <body>, tag simples

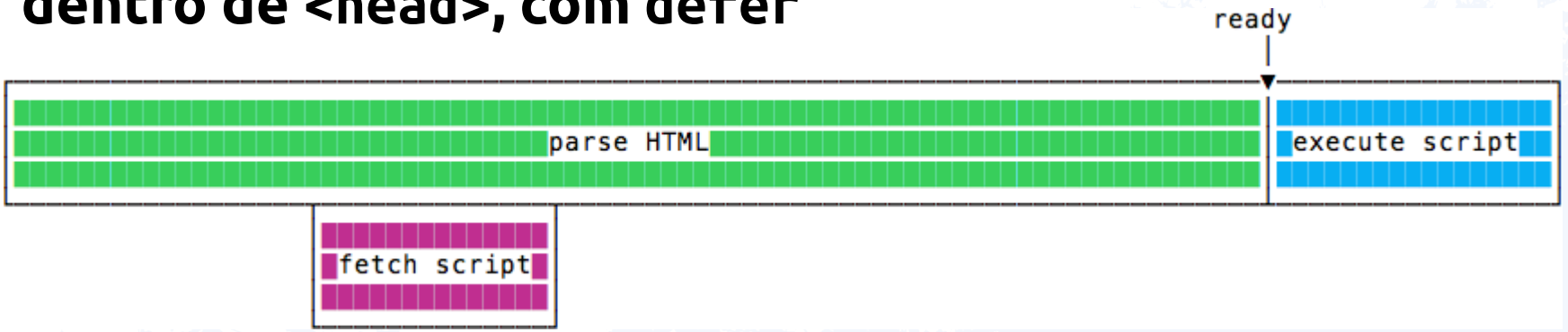


Onde colocar `<script src=...>` ?

dentro de <head>, com async



dentro de <head>, com defer

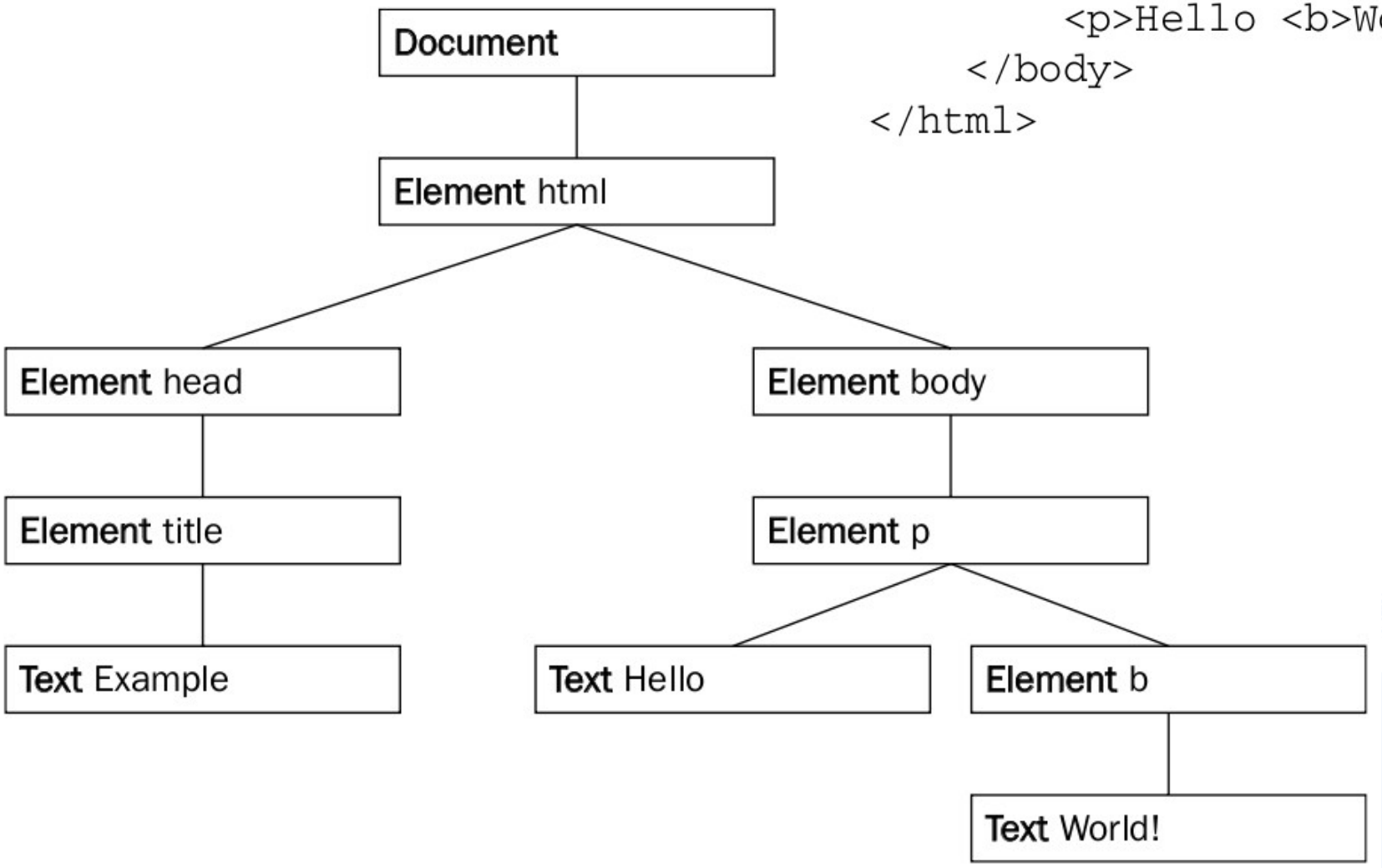


Document Object Model (DOM)

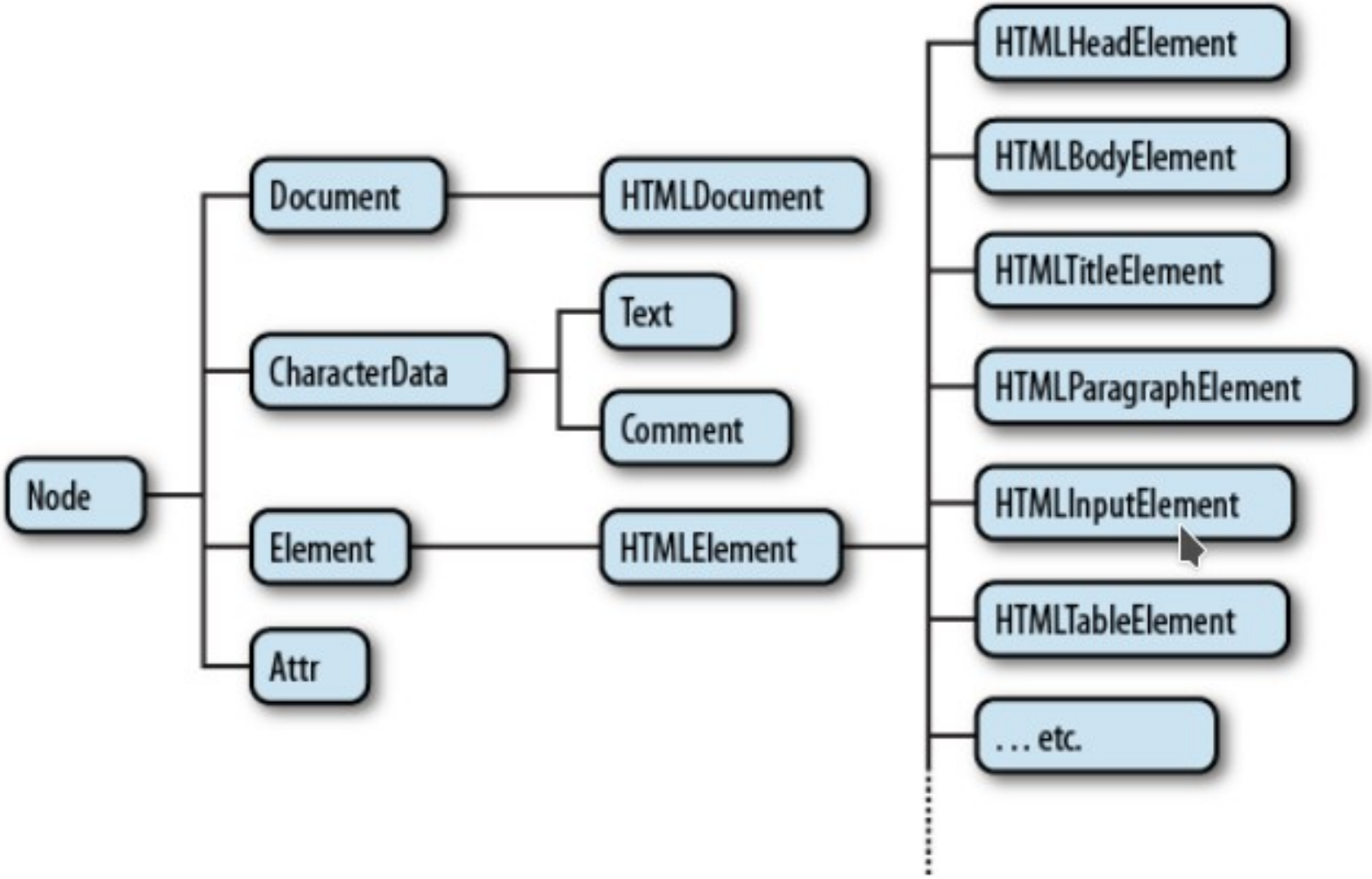
- Javascript vê o documento HTML como uma árvore que contém os elementos da página em hierarquiva
- Na raiz da árvore, está `Document` e, abaixo dele, o elemento que representa a tag `<html>`
- Todos os elementos podem ser manipulados

Exemplo

```
<html>
    <head>
        <title>Example</title>
    </head>
    <body>
        <p>Hello <b>World!</b></p>
    </body>
</html>
```



Hierarquia de classes do DOM



Acessando um elemento da DOM

- O objeto **`document`** fornece quatro métodos que retornam elementos da árvore:
  - » `document.getElementsByTagName(tag)`
  - » `document.getElementsByClassName(classe)`
  - » `document.getElementsByName(name)`
  - » `document.getElementById(id)`
- Para recuperar o elemento <body>, use simplesmente:
  - » `document.body`

Exemplo

```
<!DOCTYPE html>
<head>
    <style>.verde{ color:green; }</style>
    <script src="meucodigo.js" defer></script>
</head>
<body>
    <p id="primeirao" class="verde">Esse é o primeiro
parágrafo</p>
    <p>Esse é o segundo paragrafo</p>
    <h4 onclick="teste()" style="cursor:
pointer">Formulário:</h4>
    <form class="verde">
        Login: <input type="text" name="login"><br>
        Senha: <input type="password" name="senha">
        <input type="submit" value="Enviar">
    </form>
 </body>
</html>
```

meucodigo.js

```
function teste(){

   let primeiro = document.getElementById('primeirao');
   let paragrafos = document.getElementsByTagName('p');
   let verdes =
       document.getElementsByClassName('verde');
   let login = document.getElementsByName('login')[0];

   console.log('primeiro:', primeiro);
   console.log('login:', login);
   console.log('paragrafos:', paragrafos);
   console.log('verdes:', verdes);
 }
```

Fonte do evento

- Para obter informações sobre a fonte do evento, passe **event** como argumento para a função

```
<button onclick="fazalgo(event)">Faz algo</button>
```

```
function fazalgo(e){
    console.log(e.type, e.target);
}
```

"click"

Elemento do botão

Manipulando eventos

- É possível manipular as próprias chamadas de eventos diretamente do código Javascript
  - » Eventos são propriedades que recebem funções
  - » `this` se refere ao objeto que disparou o evento (igual à propriedade `target` do evento)
- Melhora a separação entre HTML e JS

Exemplo

```
<button id="meubotao">Faz algo</button>
```

```
document.getElementById('meubotao').onclick = function(e)
{
    console.log(e.type);
    console.log(e.target == this);
}
```

Criando novos nós

- Para criar novos nós, use os seguintes métodos de **`document`**:
  - `createElement(tag)`
  - `createTextNode(texto)`
- Para adicionar um nó a um elemento existente, use (no elemento-pai):
  - `appendChild()`

Exemplo

```
<!DOCTYPE html>
<head>
    <script src="meucodigo.js" async></script>
</head>
<body>
    <div id="paragrafos">
        <p id="primeirao" class="verde">
            Esse é o primeiro parágrafo</p>
        <p>Esse é o segundo paragrafo</p>
    </div>

    <button onclick="adiciona()">
        Adiciona!</button>
 </body>
</html>
```

Exemplo

```
function adiciona(){

    let p = document.createElement('p');
    let texto =
        document.createTextNode('Um novo parágrafo');
    p.appendChild(texto);

    let div = document.getElementById('paragrafos');
    div.appendChild(p);
}
```

Exemplo - Alternativa

```
function adiciona(){

    let p = document.createElement('p');
    // p.innerText =
    //       "Um novo parágrafo";

    p.innerHTML =
          "Um <strong>novo</strong> parágrafo";

    let div = document.getElementById('paragrafos');
    div.appendChild(p);
}
```

jQuery

write less, do more.

- A biblioteca Javascript mais usada no mundo
- Simplificar várias tarefas comuns:
  - » Manipulação da DOM, CSS, animações, Ajax
- Elimina variações entre navegadores
- Software livre

Integrando jQuery à sua página

- Duas maneiras:
  - 1. Localmente:

```
<script src="jquery-3.5.1.min.js"></script>
```

    - Baixar do site oficial: https://jquery.com/

  - 2. Utilizar versão online

```
<script
  src="http://code.jquery.com/jquery-3.5.1.min.js"
</script>
```

    - Ver última versão em: https://code.jquery.com/

Seletor jQuery

- Caractere: **$**
  - » Quase tudo em jQuery começa com o seletor
- Usado como função para selecionar elementos com a sintaxe do CSS:
- Exemplos:

```
$('#nome')
```
← Seleciona por id

```
$('p')
```
← Seleciona por tag

```
$('.destaque')
```
← Seleciona por classe CSS

Mais de um elemento

- Quando mais de um elemento é selecionado (tags, classes), use **`eq(n)`**, **`first()`** e **`last()`** para retornar um elemento específico:

```
$('p').eq(2)
```

```
$('.destaque').eq(1)
```

```
$('.destaque').first()
```

```
$('.destaque').last()
```

Iterando

- Para iterar por todos os elementos, use **each(*função*)**

```
$('p').each( function(i) {
      console.log( i , $(this) );
});
```

Modificando o texto

- **text(), text(novo)**
  - » Retorna/modifica o texto de um elemento
- **html(), html(novo)**
  - » Retorna/modifica o html interno de um elemento

```
console.log( $('#nome').text() );

$('p').eq(2).html('Testando, <em>Funciona!</em>');
```

Criando e adicionando

- Para criar um elemento, basta escrever a tag dentro do seletor:

```
let novop = $('<p>Novo parágrafo</p>');
```

- Para adicionar um elemento:
  - » append(…) → Ao final
  - » prepend(…) → Ao início

```
$('#campo').append(novop);
```

Navegando pela DOM

- **parent()** → Retorna o elemento pai
- **children()** → Retorna todos os elementos filhos
- **next()** → Retorna o próximo elemento
- **prev()** → Retorna o elemento anterior

```
$('.destaque').parent().text('Novo texto');
```

Modificando um atributo

- Para modificar o valor de um atributo qualquer de uma tag, use o método:
- **`attr(atributo, novovalor)`**

```
$('#minhaimagem').attr('src', 'foto.jpg');
```

Removendo um elemento

- Para remover um elemento da árvore, use o método:

  `remove()`

```
$('#minhaimagem').remove();
```

Modificando estilos CSS

- **`css(propriedade)`** → Retorna o valor de uma propriedade CSS
- **`css(propriedade, novovalor)`** → Modifica o valor de uma propriedade CSS
- **`css({propriedade: novovalor,`**
  `propriedade: novovalor,`
  `...})` → Modifica o valor de uma várias propriedade CSS

Exemplos

```
console.log( $('#txt').css('margin-left') );
```

```
$('#txt').css('color', 'green');
```

```
$('#txt').css(
   {
     'color': 'green',
     'font-weight': 'bold',
     'font-family' : 'sans'
   }
);
```

Lidando com classes CSS

- **`addClass(nome)`** → Adiciona uma classe CSS
- **`removeClass(nome)`** → Remove uma classe CSS
- **`toggleClass(nome)`** → Alterna entre adicionar ou remover a classe CSS

- Exemplo:

```
$('#txt').addClass('citacao');
```

Exibindo e escondendo elementos

- jQuery tem vários efeitos visuais embutidos para exibir ou esconder um elemento.
- Simples:
  - **`hide()`, `show()`, `toggle()`** → Esconde, exibe, alterna

```
$('#txt').toggle();
```

Efeito “fade”

- Usado para exibir e esconder o elemento gradualmente
  - » **`fadeOut(velocidade, callback)`** → Desaparecer
  - » **`fadeIn(velocidade, callback)`** → Aparecer
  - » **`fadeToggle(velocidade, callback)`** → Aparecer
- Parâmetros (opcionais):
  - » **`velocidade`** → milissegundos ou string (`'slow'`, `'def'`, `'fast'`)
  - » **`callback`** → função ao ser executada ao terminar

```
$('#txt').fadeToggle('slow', function(){
    console.log('Terminou!');
});
```

Efeito “slide”

- **slideUp(velocidade, callback)** → Desaparecer
- **slideDown(velocidade, callback)** → Aparecer
- **slideToggle(velocidade, callback)** → Alternar

- Mesmos parâmetros opcionais dos efeitos fade

```
$('#txt').slideUp('slow', function(){
    console.log('Terminou!');
});
```

Acrescentando eventos

- **on(evento, função)**
  - » Evento é uma string sem a palavra on
  - » Função pode ter um argumento para o evento
- Se o seletor se aplicar a mais de um elemento (tags, classes CSS), o evento será válido para todos

```
$('span').on('mouseover', function(e) {
    console.log(e.type);
    console.log(this); // igual a e.target
    console.log( $(this).text() );
});
```

Transforma **this** em um objeto jQuery

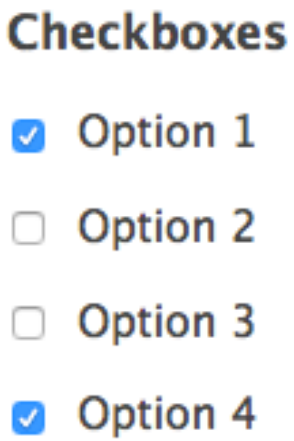
Lendo os valores de formulários

- **val()** → Retorna o valor inserido pelo usuário
  - » Caixa de texto → Texto digitado

```
<input type="text" id="nome">
```

```
console.log( $('#nome').val() );
```

Elementos **checkbox**

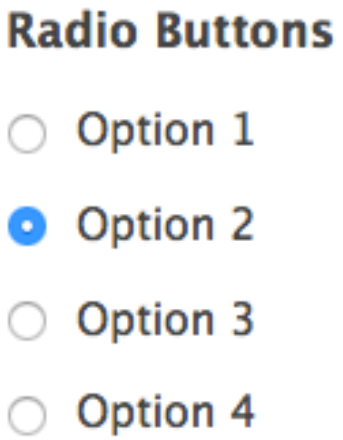


- Para verificar se uma checkbox foi marcada pelo usuário, deve-se verificar **`is(':checked')`**

```
<input type="checkbox" id="aceito">
```

```
console.log( $('#aceito').is(':checked') );
```

Elementos `radio`



- Para elementos radio, é necessário agrupar e selecionar por nome:

```
<input type="radio" name="queijo" value="parmesao">
<input type="radio" name="queijo" value="provolone">
<input type="radio" name="queijo" value="cheddar">
```

```
console.log( $("input[name='queijo']:checked").val() );
```