

Desenvolvimento para a Internet e Aplicações Móveis

JavaScript

Sumário










- JavaScript
 - Introdução
 - Exemplos simples de JavaScript
 - Variáveis e operadores
 - Estruturas de controlo (if, while, for)
 - Caixas de diálogo
 - Acesso e manipulação de elementos
- jQuery
 - Seletores
 - Eventos
 - Efeitos

JavaScript

JavaScript – o que é?

- O **JavaScript**, como o próprio nome sugere, é uma linguagem de *scripting*.
 - O código é interpretado e executado conforme vai sendo lido pelo browser (assim como o HTML)
- Uma das linguagem de programação mais populares no desenvolvimento Web.
- Usada para implementar elementos dinâmicos nas páginas
- Suportada por todos os navegadores
- Os script JavaScript em regra são executados do lado do cliente, ou seja, no *browser* do utilizador

JavaScript – o que é?

+ PROS	— CONS
 Supported by the most browsers	 Different view from different browsers
 Many libraries and frameworks	 The language is old
 No specific tools needed	 Risk of disablement
 Easy to learn and work with	 Access to the code
 Responsive web design	

<https://mobilunity.com/blog/hire-javascript-developer/>

Inserir JavaScript numa página

- Numa página HTML, o código em **JavaScript** pode ser inserido em qualquer lugar, desde que esteja dentro do elemento `<html>...</html>`
- Utiliza-se `<script>...</script>`

```
<script type="text/javascript">  
    // código js  
    // ...  
</script>
```

- Em alternativa pode-se utilizar um ficheiro externo, com extensão **.js**
 - Idêntico ao que já vimos para o CSS

```
<head>  
    <script src="scripts/myscripts.js"></script>  
    ...  
</head>
```

Inserir JavaScript numa página – exemplos

■ Exemplo – Olá Mundo!

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <title>JavaScript: básico</title>
</head>

<body>

  <h2>JavaScript: Exemplo Olá Mundo</h2>

  <script>
    alert("Olá Mundo!");
  </script>

</body>
</html>
```

JavaScript: Exemplo Olá Mundo

🌐 localhost:63342

Olá Mundo!

OK

Notas:

- Cada linha de código deve terminar com ';'
- Os scripts js são *case-sensitive*
- Comentários: `//` – linha; `/*...*/` – em bloco
- Em geral a sintaxe faz lembrar Java/C/C++

Inserir JavaScript numa página – exemplos

■ Exemplo – Botões e Lâmpada

adaptado de https://www.w3schools.com/js/js_intro.asp

...

<body>

<h2>JavaScript: Exemplo: botões e lâmpada</h2>



<button onclick="document.getElementById('lampada').src='on.gif'">

Acende

</button>



<button onclick="document.getElementById('lampada').src='off.gif'">

Apaga

</button>

</body>

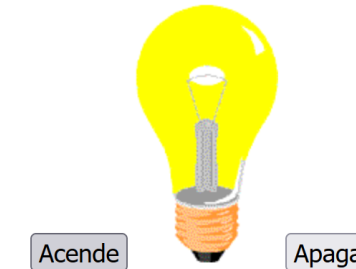
...

Alguns atributos do HTML também podem aceitar código JavaScript!

JavaScript: Exemplo botões e lâmpada



JavaScript: Exemplo botões e lâmpada



Variáveis

- O tipo das variáveis é determinado automaticamente (como no Python)

```
<h2>Variáveis Number</h2>
<script type="text/javascript">
  let a = 2, b = 3; // a, b e mult são do tipo number
  let mult = a * b;
  document.write('O valor da multiplicação é ' + mult);
</script>
```

```
<h2>Variáveis String</h2>
<script type="text/javascript">
  let str1 = 'Programação para Internet';
  let str2 = 'é uma cena fixe';
  document.write(str1 + ' ' + str2);
</script>
```

```
<h2>E ainda há mais tipos...</h2>
```

Variáveis Number

O valor da multiplicação é: 6

Variáveis String

Programação para Internet é uma cena fixe

E ainda há mais tipos...

As variáveis podem ser declaradas com:

- **let** – contexto de bloco
- **var** – contexto global/da função
- **const** – constantes
- sem *keyword* – contexto global (em regra usa-se sobretudo **let** e **const**)

Os tipos principais de variáveis são:

- Number
- String
- Boolean
- Object
- null e undefined

Operadores

■ Aritméticos

Operador	Operação	Exemplo
+	Adição	$x + y$
-	Subtração	$x - y$
*	Multiplicação	$x * y$
/	Divisão	x / y
%	Resto da divisão inteira	$x \% y$
**	Potência	$x ** y$
++	Incrementação	$x++$
--	Decrementação	$x--$

■ Atribuição

Operador	Exemplo	O mesmo que
=	$x = 5$	
+=	$x += 1$	$x = x + 1$
-=	$x -= y$	$x = x - y$
*=	$x *= 5$	$x = x * 5$
/=	$x /= 3$	$x = x / 3$
...		

Operadores

■ Lógicos

Operador	Significado	Exemplo
&&	Conjunção	$x > 0 \ \&\& \ x < 5$
	Disjunção	$x < 0 \ \ x > 10$
!	Negação	$!(x < 0)$

■ Comparação

Operador	Significado	Exemplo
>	Maior que	$x > y$
<	Menor que	$x < y$
==	Igual a	$x == y$
===	Igual valor e do mesmo tipo que	$x === y$
!=	Diferente de	$x != y$
!==	Diferente valor ou de tipo de diferente que	$x !== y$
>=	Maior ou igual a	$x >= y$
<=	Menor ou igual a	$x <= y$

Tirando um ou outro caso os operadores do JavaScript são muito idênticos aos que já conhecia do Java

Estruturas de controlo – if

- if simples

```
if (expressao) {  
    // código a executar caso  
    // "expressao" seja verdadeira  
}
```

- if-else

```
if (expressao) {  
    // código a executar caso  
    // "expressao" seja verdadeira  
}  
else {  
    // código a executar caso  
    // "expressao" seja falsa  
}
```

- if - else if - else

```
if (expressao1) {  
    // ...  
}  
else if (expressao2) {  
    // ...  
}  
else if (expressao3) {  
    // ...  
}  
else {  
    // ...  
}
```

Estruturas de controlo – Ciclos

■ while

```
while (expressao) {  
    // código a executar  
    // enquanto 'expressao' for  
    // verdadeira  
}
```

```
<script type="text/javascript">  
    let i = 0, soma = 0;  
    while (i<=10) {  
        soma += i;  
        i++;  
    }  
    alert('A soma dos números de 0 a 10 é: ' + soma);  
</script>
```

■ for

```
for (inicializacao; condicao; iteracao) {  
    // operações a serem executadas caso  
    // 'condicao' seja verdadeira  
}
```

```
<script type="text/javascript">  
    soma = 0;  
    for (let i=0; i<=10; i++)  
        soma += i;  
    alert('A soma dos números de 0 a 10 é: ' + soma);  
</script>
```

Funções

- Tal como nas restantes linguagens, no JavaScript também se podem definir funções

```
function nome_da_funcao(argumentos) {  
    // código da função  
    // (ou procedimento, caso não devolva nada)  
}
```

```
<script type="text/javascript">  
    function max(a, b) {  
        if (a > b)  
            return a;  
        return b;  
    }  
    alert("O max entre 5 e 3 é: " + max(5,3));  
</script>
```

Caixas de diálogo (pop-up boxes)

- Alert – para mostrar mensagens – tem apenas um botão de OK

```
alert(mensagem);    ou    window.alert(mensagem);
```

- Confirm – para o utilizador confirmar (ou não) alguma coisa. Caso clique em OK, a caixa devolve true; caso clique em Cancel, devolve false

```
if (confirm("Prosseguir?"))  
    txt = "Carregaste em OK!";  
else  
    txt = "Carregaste em Cancel!";
```

- Prompt – para pedir input – se o utilizador carregar em OK devolve o input introduzido; se carregar em Cancel devolve null

```
nome = prompt("Introduz o teu nome:", "John Doe");  
if (nome == null || nome == "")  
    msg = "Sem dados...";  
else  
    msg = "Olá " + nome + "! Como estás hoje?";
```

Exemplos

html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>JavaScript demo</title>
  <script src="myscripts.js"></script>
</head>
<body>

  <h2>Exemplo - cálculo de fatorial</h2>
  <button onclick="calcularFatorial()">
    Calcular!
  </button>

</body>
</html>
```

myscripts.js

```
function fatorial(n) {
  let f = 1;
  for (let i = 1; i <= n; i++) {
    f = f * i;
  }
  return f;
}

function calcularFatorial() {
  let num = prompt("Introduza um numero inteiro positivo: ");

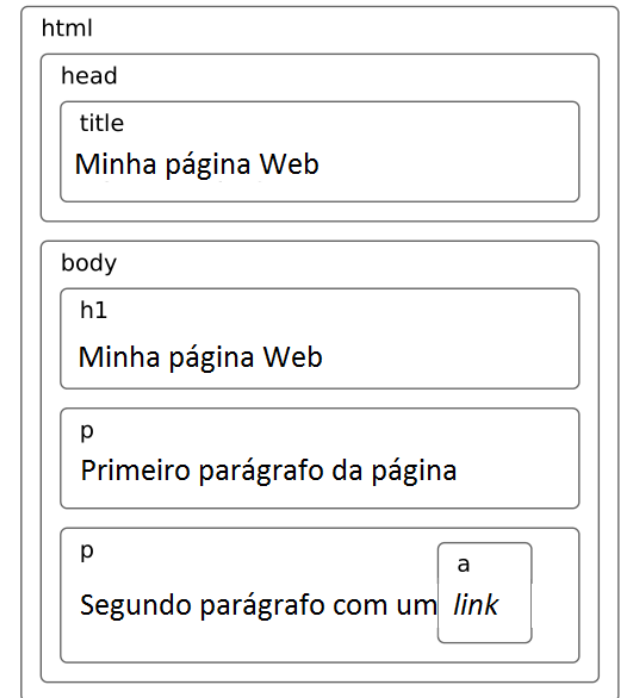
  // parseInt() converte de string para numero inteiro
  let f = fatorial(parseInt(num));

  document.write('O fatorial de ' + num + ' é: ' + f);
}
```


DOM – Document Object Model

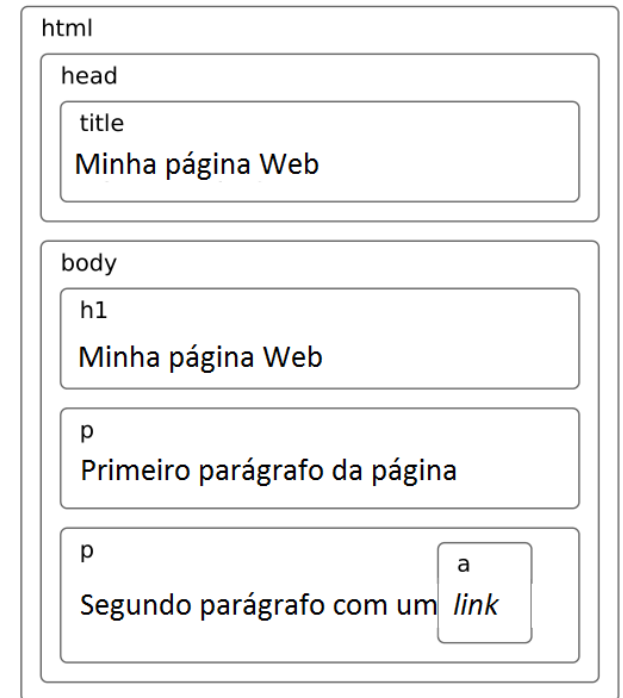
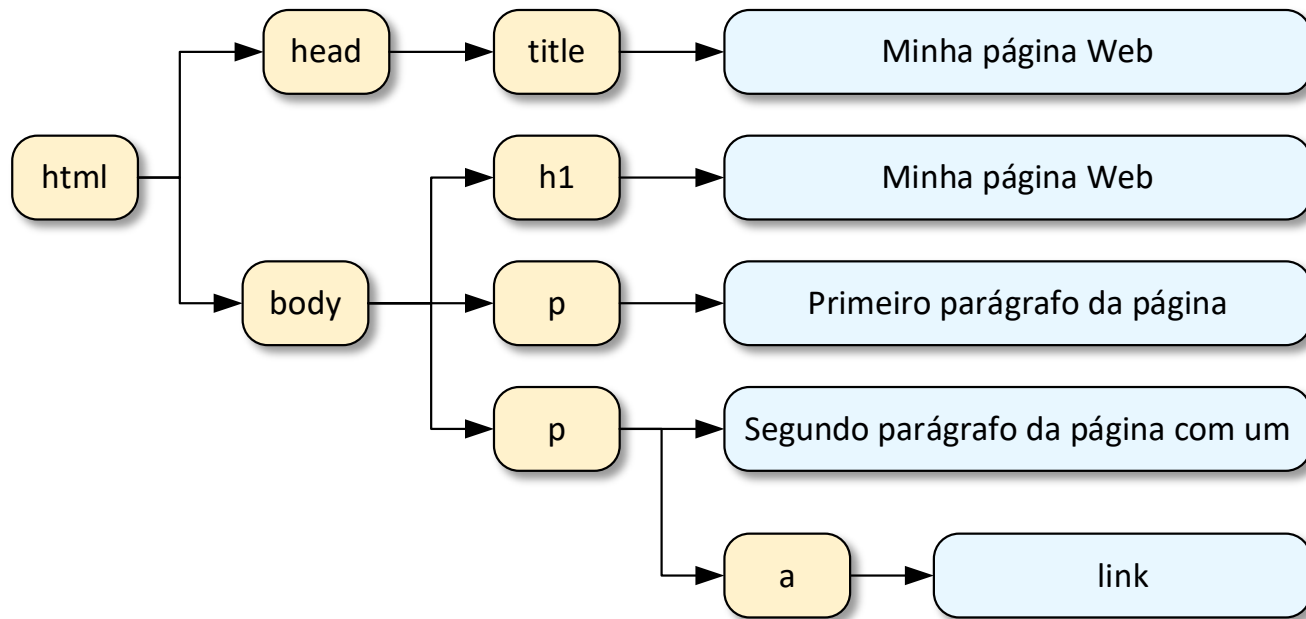
- Modelo de representação de elementos em documentos, entre os quais HTML
 - Os elementos são representados numa estrutura de árvore

```
<html>
  <head>
    <title>Minha página Web</title>
  </head>
  <body>
    <h1>Minha página Web</h1>
    <p>Primeiro parágrafo da página</p>
    <p>Segundo parágrafo com um<a href="https://www.iscte-iul.pt/">link</a>.</p>
  </body>
</html>
```



DOM – Document Object Model

- A hierarquia de objetos do documento pode ser visualizada como uma árvore hierárquica contendo vários nós
- No caso do exemplo anterior, ter-se-ia:



DOM – Document Object Model

- Assim, todos os conteúdos de um documento HTML correspondem a **nós**:
 - O documento propriamente dito é um nó do tipo **document** – a raiz da árvore
- Todos os elementos HTML, como <div>, <p>, etc., são nós do tipo **element**
- Todos os atributos HTML são nós do tipo **attribute**
- O texto dentro dos elementos HTML são nós do tipo **text**
- E até mesmo os comentários são nós do tipo **comment**

DOM e JavaScript

- No JavaScript, a estrutura de objetos do documento pode ser acedida por meio do objeto *document*, que é o nó raiz da hierarquia
- Um elemento específico no documento pode ser obtido por meio do método `getElementById(...)` do nó *document*

```
document.getElementById('id_do_elemento');
```

- O texto interno do elemento pode ser obtido ou alterado através das propriedades *innerHTML*, *innerText* ou *textContent*

```
document.getElementById('id_do_elemento').innerHTML = 'novo texto'
```

- Aliás qualquer atributo de um elemento pode ser obtido ou alterado

```
document.getElementById('id_do_elemento').nome_do_atributo = 'valor_do_atributo'
```

DOM e JavaScript – exemplo

html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>JavaScript demos</title>
  <script src="JS/myscripts.js"></script>
</head>
<body>

  <h2>Exemplo - cálculo de potência</h2>

  <p>Base: <input type="text" id="base" size="5"></p>
  <p>Expoente: <input type="text" id="expoente" size="5"></p>

  <button onclick="calcularPotencia()">Calcular!</button>

  <p id="resultado">O resultado aparecerá neste parágrafo!</p>

</body>
</html>
```

myscripts.js

```
function calcularPotencia() {

  let b = parseInt(document.getElementById("base").value);
  let e = parseInt(document.getElementById("expoente").value);

  let res = b ** e;

  document.getElementById("resultado").textContent = "O resultado é: " + res;
}
```



Exemplo - cálculo de potência

Base:

Expoente:

O resultado aparecerá neste parágrafo!



Exemplo - cálculo de potência

Base:

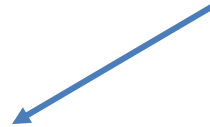
Expoente:

O resultado é: 25

DOM e JavaScript – outro exemplo

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

html



```
<p><a id="link1" href="https://tecnico.ulisboa.pt/pt/">IST</a></p>
<button id="btn1" onclick="trocaLink()">
  Muda o link para uma universidade mais fixe!
</button>
```

myscripts.js



```
</body>
</html>
```

```
function trocaLink() {
  document.getElementById("link1").innerText = "ISCTE";
  document.getElementById("link1").href = "https://www.iscte-iul.pt/";

  document.getElementById("btn1").remove();
}
```

Remove o elemento com ID
indicado, neste caso o botão



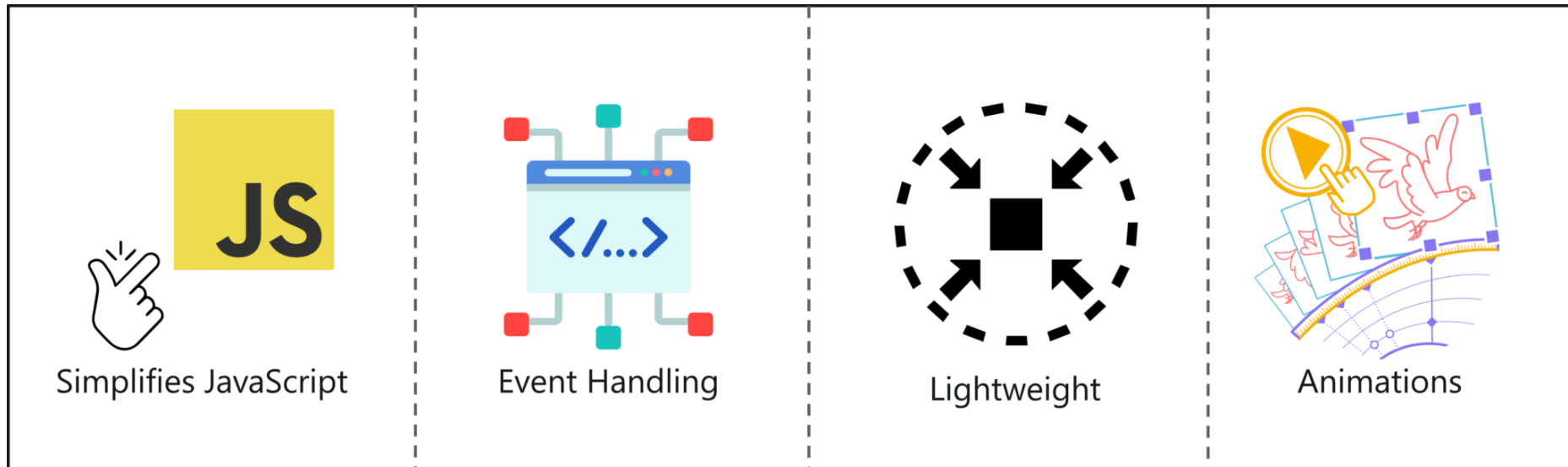
Recursos externos

- JavaScript na W3Schools
<https://www.w3schools.com/js/default.asp>
- Vídeos sobre JavaScript:
 - Introdução com o básico da linguagem
<https://www.youtube.com/watch?v=W6NZfCO5SIk>
 - *Crash course* com conceitos mais avançados (incluindo classes e exp. lambda)
<https://www.youtube.com/watch?v=hdl2bqOjy3c>
 - Curso mais longo, muito aplicado à web e com exercícios para ir fazendo
<https://www.youtube.com/watch?v=jS4aFq5-91M>
 - Aprender JavaScript implementando jogos "clássicos" em páginas web
<https://www.youtube.com/watch?v=ec8vSKJuZTk>

jQuery

Jquery – o que é?

- Uma biblioteca "leve" de JavaScript que facilita:
 - Gestão dos eventos (carregar em botões, passar com o rato por cima de elementos, etc.)
 - Animações (esconder/aparecer, fade in/out, slide up/down, mudar propriedades CSS, etc.)



<https://www.edureka.co/blog/jquery-tutorial/>

Exemplo simples

html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>jQuery Demo</title>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.4/jquery.min.js"></script>
  <script src="myscripts.js"/></script>
</head>

<body>
  <h2>Exemplo JQuery</h2>
  <p class="texto1">Qualquer coisa aqui.</p>
  <p class="texto1">Mais outra coisa aqui.</p>
  <button id="btn1">Esconder texto!</button>
</body>
</html>
```

Este exemplo evidencia que de facto o jQuery simplifica algumas coisas:

- No HTML, não foi necessário usar atributos relacionados com eventos (onclick)
- No JavaScript, a seleção de elementos está mais simplificada, sem ser necessário usar `getElementById()`, `getElementsByClassName()`
- A animação para esconder é feita com recurso ao método `hide()`, em vez de se manipularem diretamente os atributos dos elementos em causa

myscripts.js

```
$(document).ready(function() {
  $("#btn1").click(function() {
    $(".texto1").hide();
    $("#btn1").text("Agora já não funciona...");
  });
});
```

Seletores

- A utilização de seletores é um dos aspetos mais importantes do jQuery
- Permitem obter elementos e atributos HTML
- Os seletores são sempre usados com a sintaxe: `$("seletor")`
- A sua utilização é muito semelhante ao que já vimos no CSS:

<code>\$("p")</code>	<code>// seleciona todos os elementos <p></code>
<code>\$(".classe1")</code>	<code>// seleciona todos os elementos com class="classe1"</code>
<code>\$("#id1")</code>	<code>// seleciona o elemento com id="id1"</code>
<code>\$("a[target='_blank']")</code>	<code>// seleciona todos os links com atributo target="_blank"</code>
<code>...</code>	

Lista completa de seletores em: https://www.w3schools.com/jquery/jquery_ref_selectors.asp

Eventos

- O jQuery contempla eventos com diferentes origens: rato, teclado, formulários e documento/janela
- Alguns exemplos são:
 - `click()` – clique do rato sobre um elemento
 - `dblclick()` – duplo clique do rato sobre um elemento
 - `mouseenter()` / `mouseleave()` – entrada / saída do ponteiro do rato sobre um elemento
 - `keypress()` – carregar numa tecla (nem todas despoletam este evento)
 - `submit()` – quando é submetido um formulário
 - `ready()` – despoletado quando o documento é carregado
 - ...

Ver mais em: https://www.w3schools.com/jquery/jquery_events.asp

Manipulação de atributos e classes

■ Obter e alterar atributos

- `attr('nome_do_atributo')` – obter o valor de um atributo (getter)
- `attr('nome_do_atributo', 'valor_do_atributo')` – alterar o valor de um atributo (setter)
- `text()` – obter o texto de um elemento (ou texto combinado no caso de serem vários)
- `text('novo_texto')` – alterar o texto dos elementos selecionados
- ...

```
let img_link = $("#img1").attr('src');  
$("#p").attr('style', 'color:red; font-weight: bold');  
$("#h1:first").text("Agora é este o título");
```

■ Atribuir e desatribuir classes

- `addClass('nome_da_classe')` – atribui uma classe a um elemento
- `removeClass('nome_da_classe')` – retira uma atribuição de classe a um elemento

Manipulação de classes – exemplo

```
<html>
<head>
  ...
  <style>
    .green-arial-text{
      color: green;
      font-family: Arial, sans-serif;
      font-size: large;
    }
  </style>
</head>
<body>

  <p>Qualquer coisa aqui.</p>
  <p>Mais outra coisa aqui.</p>

  <button id="btnG">Texto verde e grande!</button>
  <button id="btnN">Normal, sem estilo...</button>

</body>
</html>
```

```
$(document).ready(function() {
  $("#btnG").click(function() {
    $("p").addClass("green-arial-text");
  });
});

$(document).ready(function() {
  $("#btnN").click(function() {
    $("p").removeClass("green-arial-text");
  });
});
```

Qualquer coisa aqui.

Mais outra coisa aqui.

Texto verde e grande!

Normal, sem estilo..



Qualquer coisa aqui.

Mais outra coisa aqui.

Texto verde e grande!

Normal, sem estilo..

Efeitos

- É possível adicionar com muita facilidade alguns efeitos aos elementos
 - `hide()` / `show()` – esconder ou mostrar elementos
 - `fadeIn()` / `fadeOut()` – aparecer/desaparecer de forma progressiva
 - `slideDown()` / `slideUp()` – aparecer/desaparecer aumentando/diminuindo a altura do elemento
- Estes métodos aceitam um argumento (opcional) que regula a duração do efeito

- Exemplos:

```
$("#bloco1").hide();           // esconde o elemento que tem id="bloco1"  
$("#texto1").fadeIn(1000);    // fade in do elemento que tem id="texto1" em 1000 ms  
$("p").slideDown('fast');     // aparecem todos os elementos <p> com um efeito rápido de slide  
$(".classe1").show('slow');   // aparecem todos os elementos da classe1, mais lentamente
```

Recursos externos

- Tutorial com bastantes (e boas) explicações
<https://www.tutorialrepublic.com/jquery-tutorial/>
- jQuery na W3Schools
<https://www.w3schools.com/jquery/default.asp>
- Séries de vídeos sobre jQuery
https://www.youtube.com/playlist?list=PLoYCgNOIyGABdI2V8I_SWo22tFpgh2s6_
<https://www.youtube.com/playlist?list=PLWKjhJtqVAbkyK9woUZUtunToLtNGoQHB>

Exercício JS/jQuery

- Considere o projeto tutorial desenvolvido nas últimas semanas. Pretende-se:
 1. Quando o utilizador está logado, ao fazer um duplo clique na sua foto (a que aparece sobre o banner), esta desaparece.
 2. Quando se dá um click no nome do utilizador mostra de novo a foto.
 3. Em `index.html`, colocar um botão que faz aparecer/desaparecer a lista de questões. Inicialmente a lista está escondida e o botão tem o texto “Mostrar”. Quando o utilizador prime o botão, a lista aparece e o texto do botão passa a ser “Esconder”.
 4. Na página de registo, juntar um campo de `<input>` onde o utilizador pode inserir um comentário. O texto do comentário não pode incluir as palavras insultuosas. Se o comentário for aceite deve aparecer o texto “Comentário aceite”. Se o comentário não for aceite o campo de `<input>` deve ser limpo.

Sugestões:

- para a operação de validação do texto crie um novo botão “Validar comentário”.
- as palavras insultuosas podem ser dadas diretamente num array JavaScript, e.g. `const insultos=["totó", "carago", "fonix"];`
- no JavaScript também existe o método `split(separador)` que pode ser aplicado ao texto do comentário a validar, e.g. `palavras = frase.split(" ");`