



iscte

INSTITUTO
UNIVERSITÁRIO
DE LISBOA



emprego
digital

Módulo 4: Introdução à programação em javascript

Aula 09

HTML / CSS / jQuery



Exercício 1 - Revisão

- Adicionar ao evento “**onclick**”, uma função que **aumente o tamanho do texto**, cada vez que se clica no elemento.

```
<h1 id="titulo" style="font-size: 16px">  
  Sou o maior!  
</h1>
```

Bónus: E se quisermos colocar uma cor aleatória no background ao clicar?

Estrutura do HTML

- Elementos podem conter outros elementos

```
<p>Frase sem ênfase no meio</p>
```

Frase sem ênfase no meio

```
<p>Frase com <strong>ênfase</strong> no meio</p>
```

Frase com **ênfase** no meio

Template base HTML

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="UTF-8">  
    <title>Título</title>  
  </head>  
  <body>  
    <!--Conteúdo da página-->  
  </body>  
</html>
```

Tag **opcional** que identifica o tipo de documento

Tag **principal** que encapsula todo o documento

Tag de **cabeçalhos** que encapsula a informação “invisível” do documento

Tag de **conteúdo** que encapsula tudo o resto que queremos apresentar

Atributos dos elementos

- Elementos **podem conter atributos** que definem algumas regras, comportamentos ou estilos (em desuso);

```
<font color="red">Este texto é vermelho!</font>
```

Este texto é vermelho!

Atributos dos elementos

- Outros atributos definem comportamentos, formatação, etc.
- Importantes atributos funcionais:
 - **href** - Para definir o URL de destino de uma hiperligação (ou link);
 - **target** - Permite alterar o comportamento de um link (abrir num novo separador ou na janela atual, por exemplo);
 - **src** - Para definir o URL de um recurso de imagem, vídeo ou áudio;
 - **title** - Permite adicionar texto como “*tooltip*”;
 - **style** - Permite aplicar estilos CSS ao elemento;

Atributos dos elementos

```
<a href="https://google.com" target="_blank">Pesquisar no Google!</a>
```

[Pesquisar no Google!](#)

```

```



Atributo *style*

O atributo `style` poderá ser utilizado para estilizar um elemento. Existe uma enorme lista de opções possíveis, mas por enquanto vamos utilizar os seguintes atributos de estilos:

- `padding` (e respetivas variações);
- `margin` (e respetivas variações);
- `height`;
- `width`;
- `background-color`;
- `color`;
- `font-size`;
- `text-align`;

```
<p style="margin: 5px; color: green">Sou verde</p>
```


Elementos mais usados

 ``

Texto do link

``

 `<div style="background-color: lightblue">`

`<h1>Título</h1>`

`<h2>Título</h2>`

`<h3>Título</h3>`

`<h4>Título</h4>`

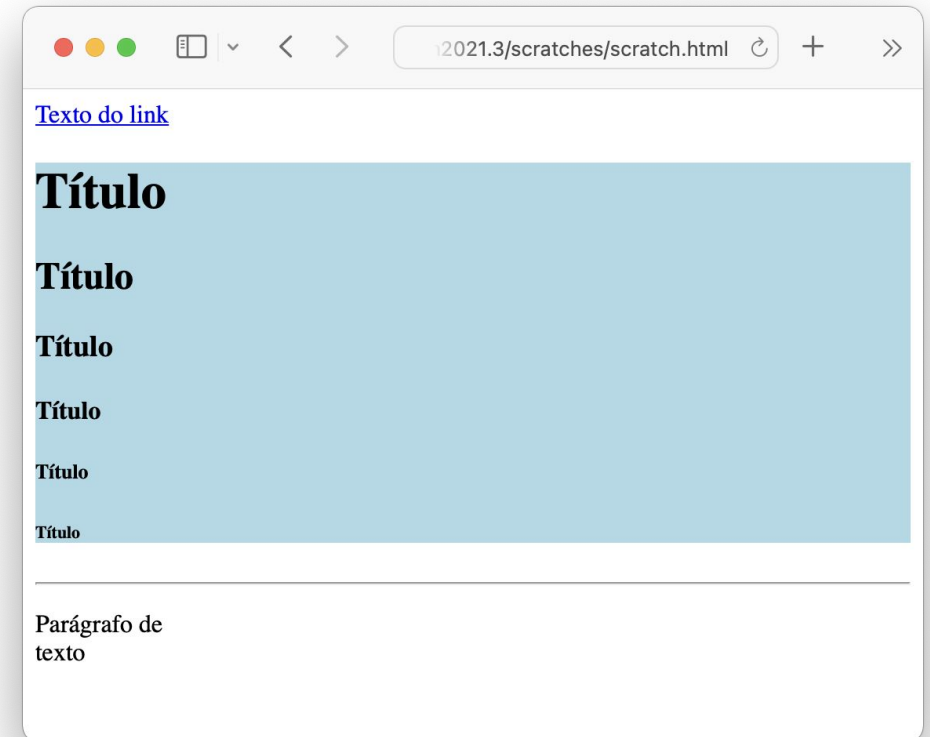
`<h5>Título</h5>`

`<h6>Título</h6>`

`</div>`

`<hr>`

 `<p>Parágrafo de
texto</p>`



Exercício 1

Pretende-se mostrar uma imagem e um link, **cada um em sua linha**, com um título azul a dizer “Viagem”

Bónus: Conseguem fazer uma imagem clicável com um link para outro sítio?

HTML com estilos inline

```
<div>
  <p style="padding: 10px;">
    <a href="fotografias.html" style="color: black; text-decoration: none;">
       Fotografias
    </a>
  </p>
  <p style="padding: 10px;">
    <a href="videos.html" style="color: black; text-decoration: none;">
       Vídeos
    </a>
  </p>
  <p style="padding: 10px;">
    <a href="musicas.html" style="color: black; text-decoration: none;">
       Músicas
    </a>
  </p>
</div>
```

HTML + CSS

```
<div class="menu">
  <p>
    <a href="fotografias.html">
      Fotografias
    </a>
  </p>
  <p>
    <a href="videos.html">
      Vídeos
    </a>
  </p>
  <p>
    <a href="musicas.html">
      Músicas
    </a>
  </p>
</div>
```

```
.menu > p {
  padding: 10px;
}

.menu > p > a {
  color: black;
  text-decoration: none;
}

.menu > p > a > img {
  margin-right: 10px;
  width: 25px;
}
```

Como adicionar CSS a um documento

Existem diversas formas de adicionar estilos a um documento:

- CSS Inline;
- CSS Interno (no próprio documento);
- CSS Externo (num ficheiro separado);



```
<h1 style="color: blue">Viagem</h1>
```

```
<style>
  h1 {
    color: blue;
  }
</style>
```

```
<h1>Viagem</h1>
```

```
<link rel="stylesheet" href="estilos.css">
```

Atributos e Seletores

```
<div class="menu">
  <!-- Menu -->
</div>

<div id="menu">
  <!-- Menu -->
</div>

<h1>
  <!-- Título -->
</h1>
```

```
.menu {
  /* Estilos */
}

#menu {
  /* Estilos */
}

h1 {
  /* Estilos */
}
```

- Para selecionar os elementos pelo atributo **class**, utiliza-se um ponto no início: **.**
- Para selecionar os elementos pelo atributo **id**, utiliza-se um cardinal no início: **#**
- Para selecionar os elementos pela sua **tag**, coloca-se só o nome

O que é o jQuery?

- É uma biblioteca escrita e montada em JavaScript, que contém ferramentas que facilitam a **obtenção de elementos** e **manipulação dos documentos**;
- Permite-nos construir **páginas dinâmicas** sem muito trabalho.



jQuery - Instalação

- A instalação do jQuery requer apenas a referência do script externo no documento.
- Aconselha-se a colocação desta tag no início do documento (dentro do head), para estar disponível em qualquer local do documento.

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
```


O seletor

- Para encontrar um elemento utilizando o jQuery, podem ser utilizados os seletores do CSS.

```
<div id="menu">
  <a class="item_menu" href="fotografias.html">
    Fotografias
  </a>
  <a class="item_menu" href="videos.html">
    Vídeos
  </a>
  <a class="item_menu" href="musicas.html">
    Músicas
  </a>
</div>
```

```
// Para encontrar o menu
let menu = $("#menu");

// Para encontrar os itens
let item_menu = $(".item_menu");

// Para encontrar as imagens
let imagens = $(".imagem");
```

Seletores - Find

- Permite encontrar um elemento descendente (que está dentro) do elemento atual.

```
<div id="menu">
  <a href="fotografias.html">
    Fotografias
  </a>
  <a href="videos.html">
    Vídeos
  </a>
  <a href="musicas.html">
    Músicas
  </a>
</div>
```

```
// Para encontrar o menu
let menu = $("#menu");

// Para procurar os links dentro do menu
let item_menu = menu.find("a");
```

Operações

```
<div id="menu">
  <a href="fotografias.html">
    Fotografias
  </a>
  <a href="videos.html">
    Vídeos
  </a>
  <a href="musicas.html">
    Músicas
  </a>
</div>
```

```
let menu = $("#menu");
let primeiro = menu.find("a").eq(0);
let segundo = menu.find("a").eq(1);

// Adiciona classe
primeiro.addClass("selecionado");

// Para ver se o elemento tem uma class
console.log(primeiro.hasClass("selecionado"));
>> true

// Remove classe
primeiro.removeClass("selecionado");

// Esconde elemento
segundo.hide();

// Mostra elemento
segundo.show();
```

Eventos

- O jQuery permite-nos registar *callbacks* em diversos eventos, nomeadamente:
 - `click()`
 - `mouseenter()`
 - `mouseleave()`

```
let menu = $("#menu");
let primeiro = menu.find("a").eq(0);
let segundo = menu.find("a").eq(1);

primeiro.click(function(){
    console.log('Clicou no primeiro elemento');
});

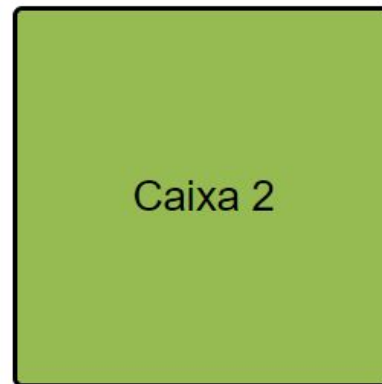
segundo.mouseenter(function(){
    console.log('Cursor está em cima do segundo elemento');
})

segundo.mouseleave(function(){
    console.log('Cursor saiu de cima do segundo elemento');
})
```

Eventos - Exemplo

- Neste exemplo queremos que ao passar por cima de um dos quadrados, o outro mude de cor:

```
<div class="caixas">
  <div class="caixa1">Caixa 1</div>
  <div class="caixa2">Caixa 2</div>
</div>
```



```
let c1 = $(".caixa1");
let c2 = $(".caixa2");

c1.click(function(){
  c2.toggleClass("azul");
})

c2.mouseenter(function(){
  c1.addClass("rosa");
})

c2.mouseleave(function(){
  c1.removeClass("rosa");
})
```

Modificação do conteúdo

```
<div id="menu">
  <a href="fotografias.html">
    Fotografias
  </a>
  <a href="videos.html">
    Vídeos
  </a>
  <a href="musicas.html">
    Músicas
  </a>
</div>
```

```
let menu = $("#menu");
let primeiro = menu.find("a").eq(0);
let segundo = menu.find("a").eq(1);

// Para alterar o conteúdo HTML
primeiro.html("Fotografias <b>minhas</b>");
console.log(primeiro.text());
>> "Fotografias minhas"

// Para alterar o conteúdo em texto
segundo.text("Vídeos fantásticos");

// Para alterar um atributo
segundo.attr("href", "videos_1.html");
```

Criação dinâmica de conteúdo

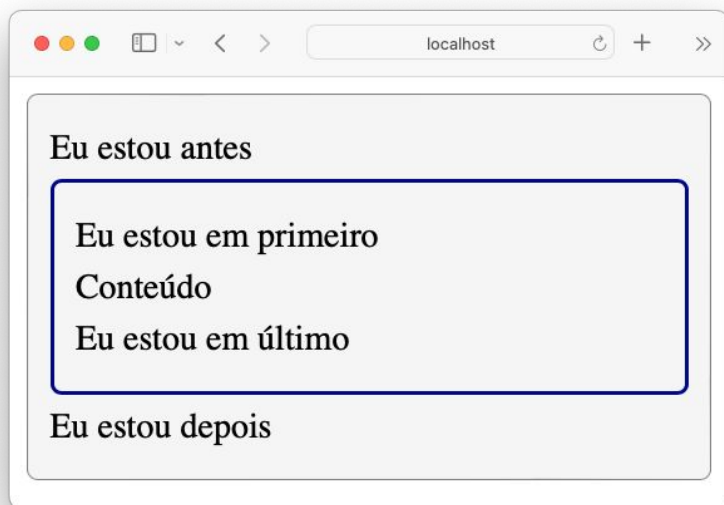
- Podemos criar os elementos dinamicamente antes de os adicionarmos ao documento ou adicioná-los diretamente indicando o código HTML.
- Esta funcionalidade poderá ser útil quando não sabemos à partida o conteúdo que a página vai ter.

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

```
let menu = $("#menu");  
  
// Criar elemento com html  
let fotografias = $("<a>Fotografias</a>");  
menu.append(fotografias);  
  
// Criar elemento e definir o html  
let videos = $("<a></a>");  
videos.html("<strong>Vídeos</strong>");  
menu.append(videos);  
  
// Adicionar o elemento diretamente com HTML  
menu.append("<a>Músicas</a>");
```

Criação de conteúdo

- Podemos usar funções como o before, prepend, append e after para colocar o conteúdo em diferentes posições relativas ao elemento que conhecemos.

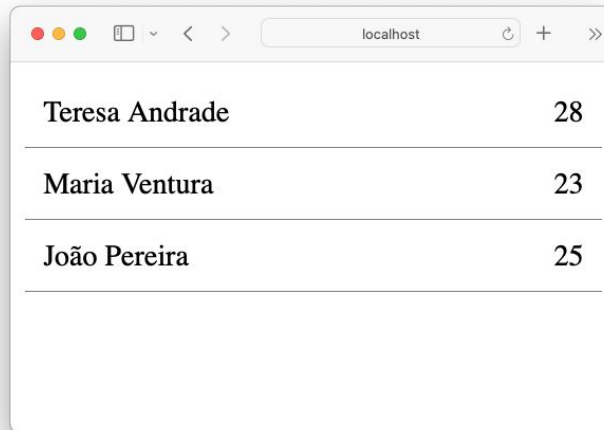


```
<div class="caixa">
  <div class="bloco">
    <p>Conteúdo</p>
  </div>
</div>
<script>
  let prefixo = 'Eu estou';

  let bloco = $(".caixa > .bloco");
  bloco.before(`<p>${prefixo} antes</p>`);
  bloco.prepend(`<p>${prefixo} em primeiro</p>`);
  bloco.append(`<p>${prefixo} em último</p>`);
  bloco.after(`<p>${prefixo} depois</p>`);
</script>
```


Exercício 2

- Utilizando a lista de alunos da aula passada, constrói uma página que a apresenta da seguinte forma:



A screenshot of a web browser window displaying a table with student information. The browser's address bar shows 'localhost'. The table has two columns: the first column contains the full names of the students, and the second column contains their ages. The students listed are Teresa Andrade (28), Maria Ventura (23), and João Pereira (25).

Teresa Andrade	28
Maria Ventura	23
João Pereira	25

Dica: também podes utilizar a interpolação de strings (templates) na construção dos blocos HTML que se repetem

```
let alunos = [  
  {  
    nome: "Teresa",  
    apelido: "Andrade",  
    idade: 28  
  }, {  
    nome: "Maria",  
    apelido: "Ventura",  
    idade: 23  
  }, {  
    nome: "João",  
    apelido: "Pereira",  
    idade: 25  
  },  
];
```

```
<div class="lista_alunos"></div>
```

Formulários

- O HTML fornece ferramentas para construir um formulário com **entrada de dados** nativamente:

```
<form method="POST">  
  <input type="text" name="name" placeholder="Nome">  
  <input type="text" name="age" placeholder="Idade">  
  <button type="submit">Enviar</button>  
</form>
```

A screenshot of a web browser window displaying a simple HTML form. The browser's address bar shows 'localhost'. The form consists of two text input fields stacked vertically. The first input field has the placeholder text 'Nome' and the second has 'Idade'. Below these inputs is a button labeled 'Enviar'.

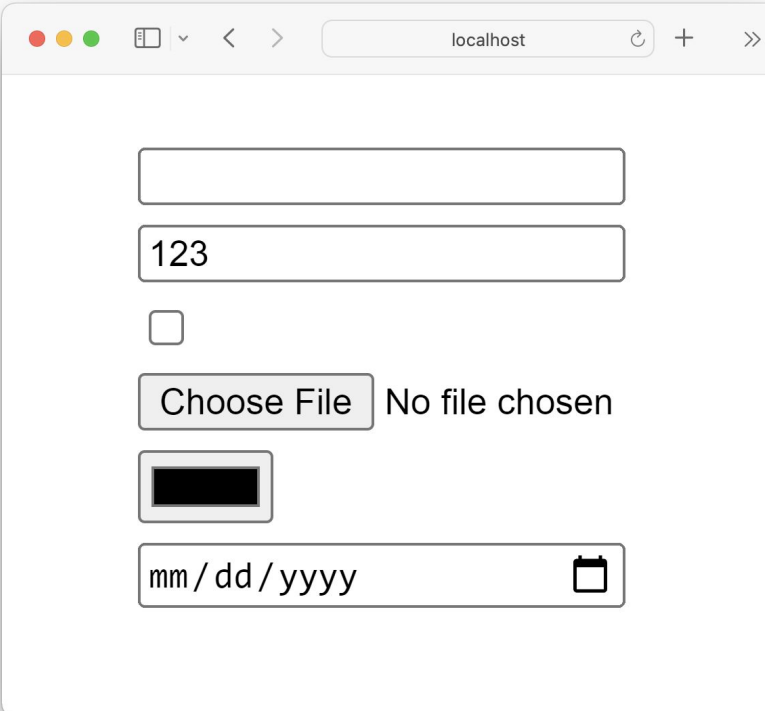
Elemento <form>

- É utilizado como parente hierárquico dos elementos de *input* que permite agrupá-los e **submeter o formulário** a uma determinada página;
- Contém um atributo “**action**” que recebe um URL para o qual é enviado o formulário;
- Utilizado em conjunto com um `<button type="submit">` para, ao clicar, agregar a informação dos inputs e enviar.

Elemento <input>

- Tem um comportamento especial porque permite a interação do utilizador e a inserção de informação, que pode ser utilizada nos formulários;

```
<input type="text">  
<input type="number">  
<input type="checkbox">  
<input type="file">  
<input type="color">  
<input type="date">
```



A screenshot of a web browser window displaying a series of HTML input elements. The browser's address bar shows 'localhost'. The elements shown are: an empty text input field, a number input field containing '123', an unchecked checkbox, a file input field with a 'Choose File' button and the text 'No file chosen', a color input field showing a black color swatch, and a date input field with the placeholder text 'mm/dd/yyyy' and a calendar icon.

Elemento <input>

- Atributos especiais do <input>:

```
//type: tipo de entrada
```

```
<input type="email">
```

```
//placeholder: valor apresentado quando o input está vazio
```

```
<input type="text" placeholder="Nome">
```

```
//name: representa o nome do atributo enviado no formulário
```

```
<input type="number" name="idade">
```

Elemento <input>

- Atributos de **controlo** do <input>:

//required: representa um campo obrigatório

```
<input type="email" required>
```

//minlength/maxlength: obriga que o texto tenha mais ou menos de X caracteres

```
<input type="text" minlength="10">
```

//min/max: para o type number, obriga um número mínimo ou máximo

```
<input type="number" max="99">
```

Elemento <input>

- Outros atributos do <input>:

```
//readonly: torna o campo não editável
```

```
<input type="email" readonly>
```

```
//disabled: parecido ao readonly, mas nem permite selecionar texto
```

```
<input type="text" disabled>
```

```
//step: para o type number, de quanto em quanto avança o número
```

```
<input type="number" step="5">
```

.val()

- A função val() permite-nos obter o valor atual de um input.

```
<input type="email" id="email">  
<input type="password" id="password">
```

```
let email = $("#email");  
console.log(email.val());  
>> "joao@upskill.pt"
```


Manipulação do Formulário

- Com o jQuery podemos **obter os valores** inseridos nos inputs, **detectar** a introdução de determinadas teclas e **validar** o que o utilizador colocou no formulário:

```
<input type="email" id="email">  
<input type="password" id="password">
```

```
.error {  
  background-color: #edd1d1;  
  color: #cd5c5c;  
  border: 1px solid #bf5b5b;  
}
```

bad@email

```
let email = $("#email");  
if(!email[0].checkValidity()) {  
  email.addClass("error");  
}  
  
let password = $("#password");  
if(password.val().length < 8) {  
  password.addClass("error");  
}
```

.on("input")

O evento "input" permite-nos detetar alterações num determinado campo. Com a definição correta, poderemos executar uma função sempre que um valor é alterado.


```
<input type="email" id="email">  
<input type="password" id="password">
```

```
let email = $("#email");  
email.on("input", function(){  
    console.log(email.val());  
});
```

Exercício 3

- Pretende-se criar um formulário que receba o **nome** e a **data de nascimento** da pessoa e coloque uma mensagem com o nome e idade por baixo (calculada pelo ano), num h2.

Dica: Podes usar o `.split("-")[0]` para ir buscar o **ano** a uma string com o formato "1997-06-22"



A screenshot of a web browser window displaying a form. The browser's address bar shows 'localhost'. The form consists of two input fields: the first contains the text 'Marisa Santos', and the second contains the date '06/22/1997' with a calendar icon to its right. Below these fields, a message is displayed: 'Nome: Marisa Santos, Idade: 25 anos'.

O futuro profissional começa aqui

iscte

INSTITUTO
UNIVERSITÁRIO
DE LISBOA



emprego
digital



UPskill