

Cap II – Estrutura / Conteúdo



Tecnologias Web
Simão Paredes sparedes@isec.pt

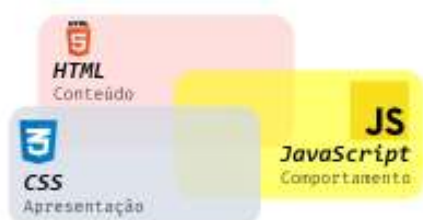
HTML

HyperText Markup Language

- As diferentes tecnologias utilizadas para a implementação de aplicação web podem ser agrupadas em diferentes camadas (*layers*)



- Diferentes Camadas
 - HTML: Estrutura / Conteúdo





```
<div class="menu">
  <ul class="firstLevel">
    <li class="firstLevel">
      <a href="/">INÍCIO</a>
    </li>
    <li class="firstLevel childrens">
      <a href="/web/Jogar/">JOGAR</a>
      <ul class="secondLevel">
        <li class="secondLevel"><a href="/web/Jogar/">Jogos e Jackpots</a></li>
        <li class="secondLevel"><a href="/web/JogarEuromilhoes/">Euromilhões</a></li>
        <li class="secondLevel"><a href="/web/JogarTotoloto/">Totoloto</a></li>
        <li class="secondLevel"><a href="/web/JogarLotClass/">Lotaria Clássica</a></li>
        <li class="secondLevel"><a href="/web/JogarLotPop/">Lotaria Popular</a></li>
        <li class="secondLevel"><a href="/web/JogarRaspadinha/">Raspadinha</a></li>
        <li class="secondLevel"><a href="/web/JogarTotobola/">Totobola</a></li>
      </ul>
    </li>
  </ul>
</div>
```



Editor HTML



HTML

- Linguagem definida pelo W3C <http://www.w3.org/> destinada a criar a estrutura de documentos (páginas) web.
 - Originalmente (web 1.0) o HTML era também responsável por toda a formatação do conteúdo. Atualmente deve ser usado **apenas** para definir a **estrutura/conteúdo**!
- Não é uma linguagem de programação mas sim uma linguagem baseada em marcas (*markup language*) diretamente interpretada pelo browser:
 - As marcas são designadas por **tags**
 - O conjunto de *tags* HTML permite definir toda a estrutura de um documento:
 - Cabeçalho
 - Corpo do documento
 - Parágrafo
 - Lista
 - Tabela, ...

■ HTML Evolução

- HTML é a tecnologia base da www

Version	Year
HTML	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2014

http://www.w3schools.com/html/html_intro.asp

HTML5

■ Evolução

“Evolution not Revolution”



- Compatibilidade com versões anteriores
- Simplificação da sintaxe
- Melhorar o tratamento de erros (markup inconsistente)
- Reduzir a necessidade de plug-in's (reprodução de áudio/vídeo)
- Reduzir a necessidade de scripting
- Aumentar a eficácia (tornar mais rápida a interpretação pelo browser)
- Melhorar a portabilidade entre browsers

- Para além das *major versions* o HTML é um *living standard*:
 - Atualizações/alterações Periódicas

HTML

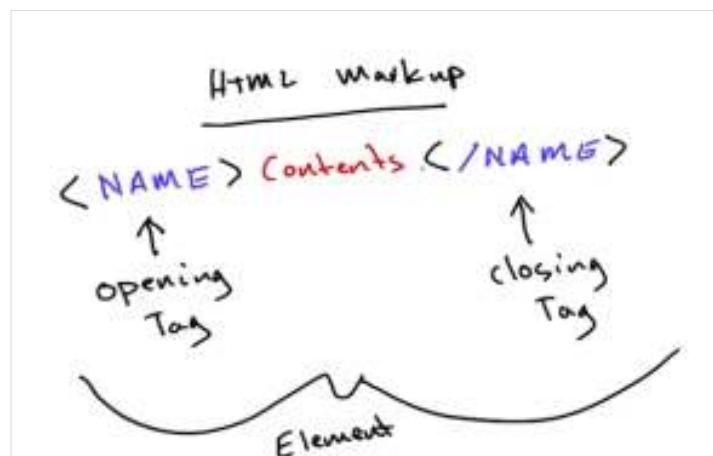
Living Standard — Last Updated 24 September 2023



<https://html.spec.whatwg.org>

HTML

- **tags** definem os elementos estruturais
- Regra geral as tags são definidas aos pares
 - <Opening tag> ... </Closing tag>
 - <h1> Exemplo de heading 1 </h1>



<https://www.udacity.com>

■ Elemento HTML

- Documento HTML é definido pelos elementos HTML
- *Opening tag* + conteúdo + *Closing tag*
 - O conteúdo de um elemento é tudo aquilo que é definido entre a *opening tag* e respetiva *closing tag*.
- Alguns elementos não têm conteúdo (*empty elements*), possuem apenas atributos
 - `
`; ``;...
 - Terminam na *opening tag*.
- Maioria dos elementos pode ter atributos

` Home `

■ Atributo HTML

- Atributos complementam os elementos HTML
- Podem ser **obrigatórios**, i.e. sem a sua informação a *tag* é ignorada

` Home `

- Atributos são **sempre** especificados **na opening tag**
- Sintaxe dos atributos é sempre efetuada do seguinte modo:
 - `name="value"`
- Um elemento pode ter vários atributos separados por espaço
 - `< openingTag attribute1="value1" attribute2="value2"...> .. </ closingTag>`

■ Atributos HTML

- Dois atributos chave para permitir o acesso a conteúdos específicos (formatação, definição de layout, ...)
 - **id** = “ value “ ; **class** = “value”
- O atributo **id** é único (atribuído uma única vez a apenas um elemento) [identificação]
- O atributo **class** pode ser atribuído a múltiplos elementos [classificação]
- Exemplos:
 - Referenciar elementos HTML para aplicar a tecnologia CSS destinados à formatação do conteúdo
 - Referenciar elementos para manipulação através de *JavaScript*
 - ...

■ Entidade HTML

- Códigos necessários para representar caracteres reservados no HTML

Result	Description	Entity Name	Entity Number
	non-breaking space	 	
<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
¢	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
€	euro	€	€
§	section	§	§
©	copyright	©	©
®	registered trademark	®	®
™	trademark	™	™

- O que é ignorado pelo *browser*?
 - Conteúdo dos comentários
 - `<!-- texto a ser ignorado, entre símbolos de comentário -->`
 - Espaços em branco seguidos;
 - Tabulações (*tabs*);
 - Mudança de linha (*carriage return*);
 - *Markup* inválido (*tags* não reconhecidas/válidas; atributos obrigatórios não especificados);

- Estrutura de um *html document*:

```
<!DOCTYPE html>

<html>

    <head>

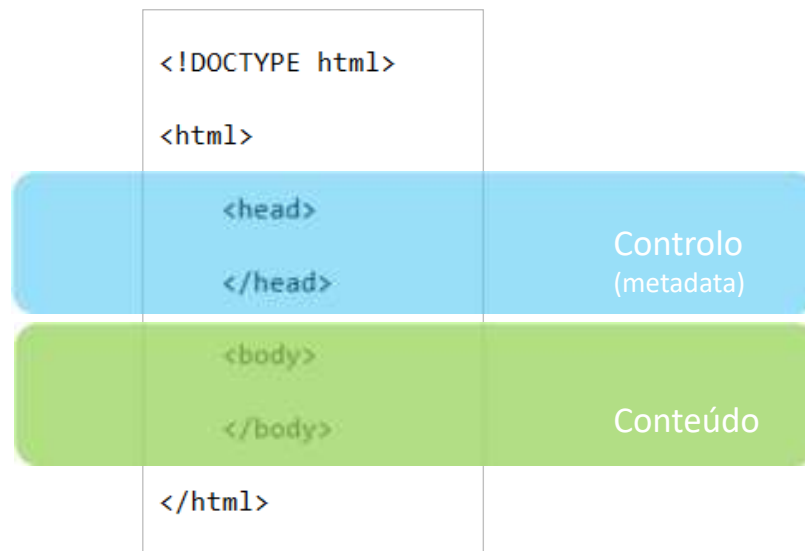
    </head>

    <body>

    </body>

</html>
```

- Declaração do documento **`<!DOCTYPE html >`**
 - Indica a versão do HTML que está a ser utilizada (elementos e atributos a utilizar no documento)
- O início e o fim do documento é delimitado pelas tags **`<html> ... </html>`**
- Cabeçalho (informação de controlo) **`<head> ... </head>`**
- Corpo (conteúdo do documento) **`<body>....</body>`**



According to the HTML5 standard; the `<html>`, the `<body>`, and the `<head>` tag **can be omitted**.

Note: W3Schools does not recommend omitting the `<html>`, `<body>`, and `<head>` tags. Omitting these tags can crash DOM/XML software and produce errors in older browsers.

http://www.w3schools.com/html/html_head.asp

<head>...</head>

Informação de **Controlo** da página

<head>...</head>

■ <meta ... >

■ Importante!

- Deve ser sempre especificada!

■ empty tag

■ metadados (metainformação)

- Dados sobre outros dados (conteúdo do documento HTML)

■ Informação não visível mas determinante para:

- Facilitar/garantir a correta interpretação do HTML pelo browser;
 - Codificação de caracteres, ...
- Possibilitar o *responsive web design* (ajuste automático do conteúdo às dimensões do dispositivo)
- Facilitar/permitir a indexação pelos motores de busca (SEO)
 - *description, keywords,*

http://www.w3schools.com/tags/tag_meta.asp

<head>...</head>

■ <meta ... >

■ Codificação de caracteres

- A codificação por defeito no HTML5 é UTF-8 (Unicode) que cobre a esmagadora maioria dos caracteres e símbolos disponíveis.
 - As versões anteriores do HTML suportavam por defeito diferentes codificações, o que frequentemente implicava a necessidade de forçar a codificação em Unicode
- No entanto e de forma a garantir a correta codificação dos caracteres, a codificação deve ser sempre especificada explicitamente
 - <https://www.w3.org/International/questions/qa-html-encoding-declarations>

<meta charset="UTF-8">

UTF-8

UTF-8 is an 8-bit variable-length encoding scheme designed to be compatible with ASCII encoding. In **this encoding, from 1 up to 4 bytes can be used to encode a character** (unlike ASCII which uses fixed 1 byte).

In a **variable-length encoding scheme**, a character is encoded in multiple **of N bits**. In UTF-8 encoding, a character can take **M x 8 bits of memory**, where N is 8 (fixed) and M can be 1 up to 4 (variable).

<https://medium.com/jspoint/introduction-to-character-encoding-3b9735f265a6>

<head>...</head>

■ <meta ... >

■ Responsive

A <meta> viewport element gives the browser instructions on how to control the page's dimensions and scaling

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Atributo **content** usado para definir o valor concreto, dos atributos **http-equiv** e **name**

<head>...</head>

■ <meta ... >

■ SEO

Atributo **content** usado para definir o valor concreto, dos atributos **http-equiv** e **name**

```
<meta name="description" content="Web tutorials">
```

```
<meta name="keywords" content="HTML,CSS,JavaScript">
```

Attributes

Attribute	Value	Description
charset	character_set	Specifies the character encoding for the HTML document
content	text	Specifies the value associated with the http-equiv or name attribute
http-equiv	content-security-policy content-type default-style refresh	Provides an HTTP header for the information/value of the content attribute
name	application-name author description generator keywords viewport	Specifies a name for the metadata

<head>...</head>

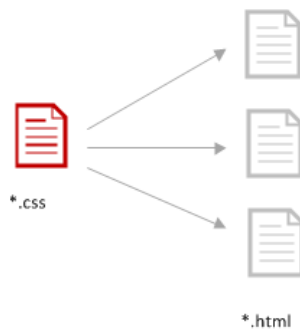
■ <link >

■ Importante!

- Permite associar ao documento HTML recursos/elementos externos (exemplo: ligação a folhas de estilo CSS)

```
<link rel="stylesheet" href="css/bootstrap.min.css" >
```

```
<link rel="stylesheet" href="css/styles.css" >
```



<head>...</head>

■ <script> </script>

■ Importante!

- Permite a definição/inserção direta de um *client-side script* (JavaScript).
 - Atualmente o JavaScript assume uma importância crucial nas *web applications*

```
<script>
  function showDetails(animal) {
    var animalType = animal.getAttribute("data-animal-type");
    var animalSize=animal.getAttribute("data-animal-size");
    alert("The " + animal.innerHTML + " is a " + animalSize + " " + animalType + ".");
  }
</script>
```

- Permite a ligação a um script externo

```
<script src="js/bootstrap.min.js"></script>
```

<head>...</head>

■ <title>...</title>

- Define o título de um documento HTML
- Importante para o posicionamento como resultado de uma pesquisa por um motor de busca (SEO)
- Permite a visualização do título da página quando adicionada aos favoritos

■ <style> ... </style>

- Permite a introdução direta de código CSS (formatação) num documento HTML
- Cada documento HTML pode conter múltiplas tags <style>

```
<style>
tfoot{background-color: lightblue}
thead{background-color: lightgreen}
tbody{background-color: lightgray}
</style>
```

<head>...</head>

■ Menos relevantes

■ <base ... >

- Permite definir um destino (alvo) para todas as ligações num documento HTML ou um comportamento dos links

■ exemplo:

```
<base target="_blank"/>
```

- Todos os links vão ser abertos numa nova janela
- Só pode existir um elemento <base> por documento HTML.
- Não prevalece sobre definições locais nos links dos atributos *href* ou *target*

```
<head>
  <base target="_blank"/>
</head>

<body>
  <a href="http://www.isec.pt" target="_self">tag base</a>
```

Prevalece a última definição do atributo

<head>...</head>

■ <noscript> </noscript>

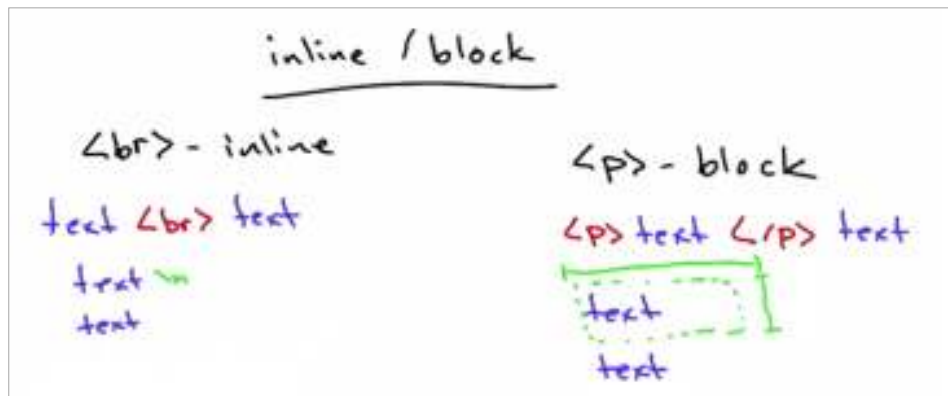
- Permite definir um conteúdo alternativo caso o script não possa ser executado
 - O *browser* não suporta JavaScript e/ou os scripts foram desativados no browser
- Também pode ser utilizada no body

```
<script>
document.write("Hello World!")
</script>
<noscript>Your browser does not support JavaScript!</noscript>
```

<body>...</body>

Conteúdo **Visível**

- Todos os elementos podem ser classificados de acordo com o seu *display value*:
 - block
 - inline
 - São interpretados de forma diferente pelo browser



Block-level Elements

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

The <div> element is a block-level element.

Examples of block-level elements:

- <div>
- <h1> - <h6>
- <p>
- <form>

http://www.w3schools.com/html/html_blocks.asp

Inline Elements

An inline element does not start on a new line and only takes up as much width as necessary.

This is an inline element inside a paragraph.

Examples of inline elements:

-
- <a>
-

http://www.w3schools.com/html/html_blocks.asp

Estruturação do Conteúdo

tags mais utilizadas

`<body>...</body>`

Estruturação do Conteúdo

■ *Headings* (Títulos)

- Os títulos são importantes para a definição de uma estrutura coerente do documento
 - A formatação da informação **não deve** ser efetuada com base na tag `<h*>..</h*>`

```
<h1>heading de nível 1</h1>  
<h2>heading de nível 2</h2>  
<h3>heading de nível 3</h3>  
<h4>heading de nível 4</h4>  
<h5>heading de nível 5</h5>  
<h6>heading de nível 6</h6>
```

heading de nível 1

heading de nível 2

heading de nível 3

heading de nível 4

heading de nível 5

heading de nível 6

■ *hgroup*

- Não tão frequente/importante como os headings
- Por vezes utilizado para agrupar elementos h1 – h6 quando existem títulos / subtítulos (vários níveis)

```
<hgroup id="document-title">
  <h1>HTML</h1>
  <h2>Living Standard — Last Updated</h2>
</hgroup>
```

HTML

Living Standard — Last Updated

■ Parágrafos

- **<p> ...</p>**
 - um parágrafo é um elemento estrutural, não deve ser confundido com a mudança de linha **
**
 - *block element* que contém texto

<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.... **</p>**

Estruturação do Conteúdo

■ Elementos **Containers**

- Absolutamente essenciais para estruturar conteúdo

- **<div> ... </div>**

- é um **block - level element**
- fundamental para, em conjunto com as CSS, permitir a formatação de blocos de conteúdo, construção de layouts, isolar comportamentos definidos em JavaScript, ...

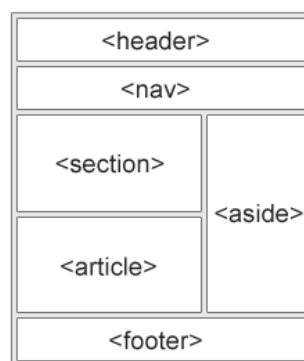
- ** ... **

- é um **inline element**
- utilizado como container para texto
- fundamental para, em conjunto com as CSS, isolar e formatar partes de texto

Estruturação do Conteúdo

■ **tags** com significado semântico

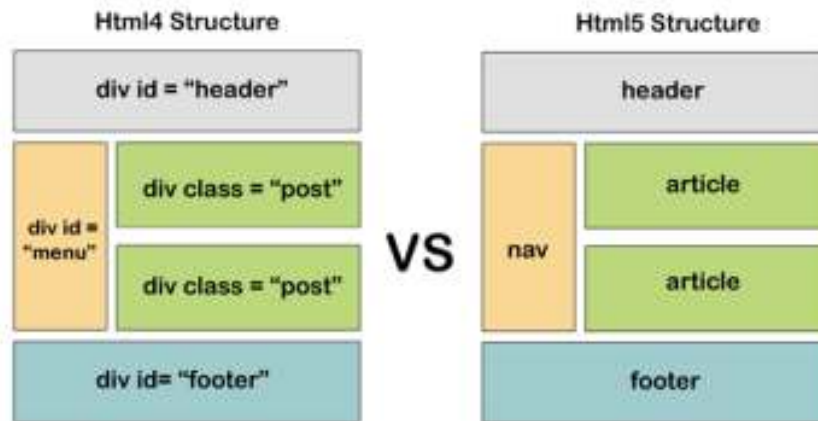
- **<header>...</header>**
- **<footer>...</footer>**
- **<nav>...</nav>**
- **<section>...</section>**
- **<article>...</article>, ...**



- Melhorar a estrutura de um documento tornando universal a sua interpretação por diversos *user agents* (ex: *screen reader*)
 - Melhorar a performance do browser (mais rápido), melhora a acessibilidade
 - Não possuem formatações/posicionamentos pré-definidos.
 - Antes do surgimento destas tags semânticas, toda a estruturação do layout era efetuada tendo por base o elemento <div>.

Estruturação do Conteúdo

- Duas abordagens à organização do conteúdo
 - Estrutura clássica baseada exclusivamente no container <div>
 - Estrutura baseada em *tags* semânticas



<https://www.javatpoint.com/html-vs-html5>

Estruturação do Conteúdo

<i>tag semântica</i>	<i>observações</i>
<code><header> ... </header></code>	enquadra a página ou um conteúdo específico
<code><nav> ... </nav></code>	permite estabelecer a navegação principal
<code><section>...</section></code>	designa secções genéricas de um documento
<code><article>...</article></code>	representa um conteúdo que forma um elemento independente
<code><aside>...</aside></code>	representa conteúdo que se encontra relacionado com o conteúdo principal do documento
<code><footer>...</footer></code>	elemento de rodapé
...	...

http://www.w3schools.com/html/html5_new_elements.asp

Estruturação do Conteúdo

■ Listas

- Elementos estruturais muito utilizados, em particular para implementar menus (horizontais/verticais) de navegação.

```
<ul class="firstLevel">
  <li class="firstLevel">
    <a href="/">INÍCIO</a>
  </li>
  <li class="firstLevel childrens">
    <a href="/web/Jogar/">JOGAR</a>
    <ul class="secondLevel">
      <li class="secondLevel"><a href="/web/Jogar/">Jogos e Jackpots</a></li>
      <li class="secondLevel"><a href="/web/JogarEuromilhoes/">Euromilhões</a></li>
      <li class="secondLevel"><a href="/web/JogarTotoloto/">Totoloto</a></li>
      <li class="secondLevel"><a href="/web/JogarLotClass/">Lotaria Clássica</a></li>
      <li class="secondLevel"><a href="/web/JogarLotPop/">Lotaria Popular</a></li>
      <li class="secondLevel"><a href="/web/JogarRaspadinha/">Raspadinha</a></li>
      <li class="secondLevel"><a href="/web/JogarTotobola/">Totobola</a></li>
    </ul>
  </li>
</ul>
```



- *unordered list* (lista não ordenada)

- ` ... ` (*unordered list*)

- ` ... ` (*list item*)

```
<h3>Lista Não Ordenada</h3>
<ul>
  <li>Objetivos </li>
  <li>Programa</li>
  <li>Avaliação</li>
</ul>
```

Lista Não Ordenada

- Objetivos
- Programa
- Avaliação

Estruturação do Conteúdo

■ Listas

- *ordered list* (lista ordenada)

- Implementação em tudo semelhante a uma lista não ordenada

- ` ... ` (*ordered list*);

- ` ... ` (*list item*)

```
<h3>Lista Ordenada</h3>
<ol>
  <li>Objetivos </li>
  <li>Programa</li>
  <li>Avaliação</li>
</ol>
```

Lista Ordenada

1. Objetivos
2. Programa
3. Avaliação

Estruturação do Conteúdo

Listas/Sublistas

- Nos dois tipos de lista é possível definir sublistas
 - Úteis para a construção de menus.
 - A ordem do fecho dos diversos *list items* não é arbitrária. A *tag* de fecho é posicionada marcando o final de todo o conteúdo desse elemento ``
 - O **encadeamento correto** das listas/sublistas é fundamental para garantir o controlo do seu funcionamento/formatação.

Lista e Sublistas

- Objectivos
- Programa
 - 1º Ano
 - 2º Ano
 - 1º Semestre
 - 2º Semestre
- Avaliação

```
<h2>Lista e Sublistas</h2>
<ol>
  <li>Objectivos</li>
  <li>Programa
    <ul>
      <li>1º Ano</li>
      <li>2º Ano
        <ul>
          <li>1º Semestre</li>
          <li>2º Semestre</li>
        </ul>
      </li>
    </ul>
  </li>
  <li>Avaliação</li>
</ol>
```

Estruturação do Conteúdo

Tabelas

- table*: `<table>...</table>`
- table row*: `<tr>...</tr>` (a definição da tabela é efetuada **por linha**)
- table header*: `<th> ...</th>`
- table data*: `<td>...</td>`

```
<table>
  <tr>
    <th>...</th>
    <th>...</th>
    <th>...</th>
  </tr>
  <tr>
    <td>...</td>
    <td>...</td>
    <td>...</td>
  </tr>
  <tr>
    <td>...</td>
    <td>...</td>
    <td>...</td>
  </tr>
  <tr>
    <td>...</td>
    <td>...</td>
    <td>...</td>
  </tr>
</table>
```

Estruturação do Conteúdo

■ Tabelas

- `<caption>...</caption>`
 - Título / Descrição do conteúdo da tabela (acessibilidade)
 - Deve ser o primeiro elemento a ser definido no interior da tabela

```
<table border="1">
  <caption>Vendas Mercado Automóvel</caption>
  <tr>
    <td></td><td>BMW</td><td>Volvo</td><td>Audi</td>
  </tr>
  <tr>
    <td>Alemanha</td><td></td><td></td><td></td>
  </tr>
  <tr>
    <td>França</td><td></td><td></td><td></td>
  </tr>
  <tr>
    <td>Itália</td><td></td><td></td><td></td>
  </tr>
</table>
```

	BMW	Volvo	Audi
Alemanha			
França			
Itália			

Estruturação do Conteúdo

■ Tabelas

- *Spanning* : dimensionar uma célula para ocupar várias colunas ou linhas
 - Agrupamento de colunas: *colspan*
 - Exemplo: Total de vendas por país

```
<table border="1">
  <caption>Vendas Mercado Automóvel</caption>
  <tr>
    <td></td><td>BMW</td><td>Volvo</td><td>Audi</td>
  </tr>
  <tr>
    <td>Alemanha</td><td colspan="3"></td>
  </tr>
  <tr>
    <td>França</td><td colspan="3"></td>
  </tr>
  <tr>
    <td>Itália</td><td colspan="3"></td>
  </tr>
</table>
```

	BMW	Volvo	Audi
Alemanha			
França			
Itália			

Estruturação do Conteúdo

- Agrupamento de linhas: *rowspan*
 - Exemplo: Total de vendas por marca

```
<table border="1">
  <caption>Vendas Mercado Automóvel</caption>
  <tr>
    <td></td><td>BMW</td><td>Volvo</td><td>Audi</td>
  </tr>
  <tr>
    <td>Alemanha</td>
    <td rowspan="3"></td>
    <td rowspan="3"></td>
    <td rowspan="3"></td>
  </tr>
  <tr>
    <td>França</td>
  </tr>
  <tr>
    <td>Itália</td>
  </tr>
</table>
```

Vendas Mercado Automóvel

	BMW	Volvo	Audi
Alemanha			
França			
Itália			

Estruturação do Conteúdo

- Tabelas
 - Informação estrutural (útil para formatação e posicionamento):
 - **<thead>...</thead>** : Cabeçalho da tabela
 - **<tbody>...</tbody>** : Corpo da tabela
 - **<tfoot>...</tfoot>** : Rodapé da tabela

```
<table>
  <tfoot>
    <tr>
      <td>foot 1</td><td>foot 2</td></tr>
    </tfoot>

    <thead>
      <tr>
        <td>head 1</td><td>head 2</td></tr>
      </thead>

      <tbody>
        <tr>
          <td>body 1.1</td><td>body 1.2</td></tr>
          <tr>
            <td>body 2.1</td><td>body 2.2</td></tr>
          </tbody>
        </table>
```

```
<style>
  tfoot{background-color: lightblue}
  thead{background-color: lightgreen}
  tbody{background-color: lightgray}
</style>
```

head 1	head 2
body 1.1	body 1.2
body 2.1	body 2.2
foot 1	foot 2

Navegação

tags mais utilizadas

`<body>...</body>`

Navegação

- Hiperligação (*link*) `<a> ... `
 - A possibilidade de estabelecer hiperligações (acesso não linear à informação) é a base de um sistema hipermédia;



- Atributo **href** obrigatório na *tag* `<a>`
 - A única exceção ocorre quando se pretende referenciar uma ponto para navegação interna tendo por base também a *tag* `<a>`

Navegação

■ Caminho absoluto:

■ `href="http://www. ..."`

- Referência para um recurso na web num servidor remoto (servidor diferente daquele armazena o ficheiro)
- Especificação completa do URL

```
<a href="http://www.isec.pt" target="_blank">Instituto Superior de Engenharia de Coimbra</a>
```

[Instituto Superior de Engenharia de Coimbra](http://www.isec.pt)



Navegação

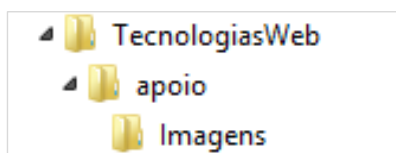
■ Caminho Relativo:

■ `href="/path ..."`

- Referência para um recurso/asset (imagem, html, ...) no mesmo servidor em que se encontra alojada a aplicação web
- Ao contrário do caminho absoluto não requer a especificação do protocolo nem do *host*, apenas do caminho (*path*)
- `../` :ligação para uma directoria hierarquicamente superior

```
<a href="Imagens/logo-isec-transparente.png">Símbolo ISEC - Link Interno</a>
```

Considerando o *.html na pasta apoio e o *.png armazenado na subdiretoria Imagens

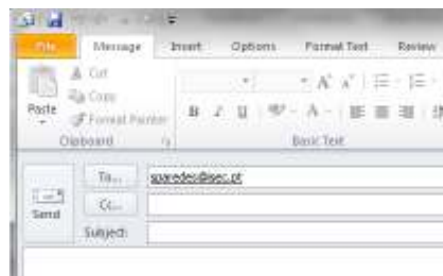


Navegação

■ Ligação automática a um cliente de mail

- `href="mailto: ..."`
 - O destinatário principal deve ser especificado

```
<a href="mailto:sparedes@isec.pt">Contactar Professor</a>
```



■ É possível definir título/corpo da mensagem e restantes destinatários

- %20 (UTF-8) para definir um espaço num URL

Parameter	Description
<code>mailto:name@email.com</code>	e-mail recipient address
<code>cc=name@email.com</code>	carbon copy e-mail address
<code>bcc=name@email.com</code>	blind carbon copy e-mail address
<code>subject=subject text</code>	subject of e-mail
<code>body=body text</code>	body of e-mail
<code>?</code>	first parameter delimiter
<code>&</code>	other parameters delimiter

<http://www.rapidtables.com/web/html/mailto.htm>

Navegação

■ Navegação Interna (*bookmarks*)

- `href="#value"`
 - Muito útil quando se tem uma página muito longa (evita o *scroll* repetido da página)
 - O símbolo **#** no atributo href indica ao browser que se trata de uma âncora de navegação
 - Requer que o destino da ligação seja referenciado através do respetivo atributo **id**
 - `<h2 id="value"> ... </h2>` ;
 - `<div id="value"> ... </div>`,

```
<a href="#destino">Âncora Visível</a>

<br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/>

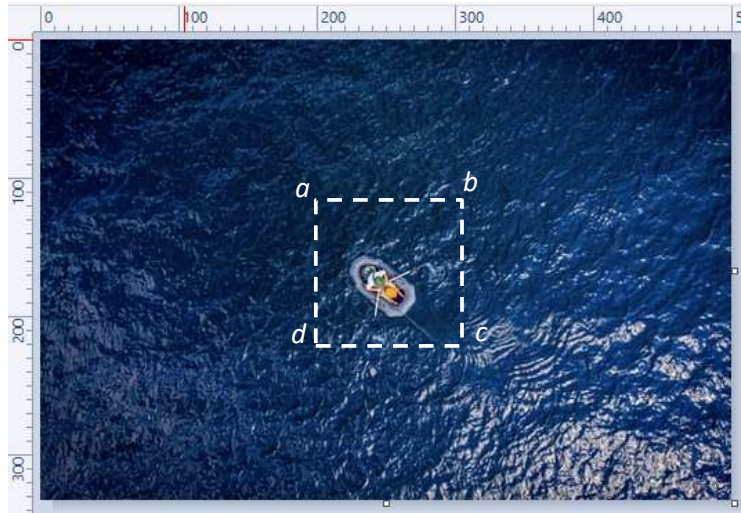
<div id="destino">Âncora</div>

<br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/>
```

■ Navegação com base numa imagem

■ <map> + <area>

- definir um map em que é possível definir áreas como origem para ligações cujo destino é definido na tag <area>
- Para a mesma imagem é possível definir várias áreas com ligações diferentes



Exemplo:

Criar uma ligação na zona identificada a tracejado:

a (200,100)

b (320,100)

c (320,230)

d (200,230)

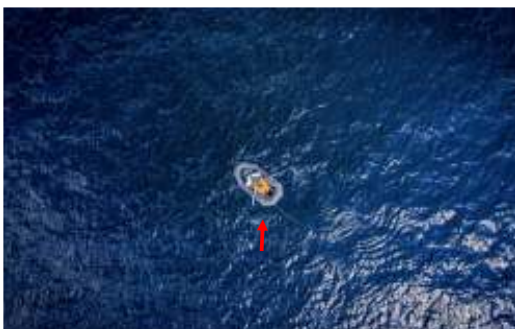
■ Navegação com base numa imagem

■ <map> + <area>

- definir um map em que é possível definir áreas como origem para ligações cujo destino é definido na tag <area>

usemap: define um *image map*, i.e.
uma imagem com áreas "clicáveis"

```
<body>  
    
  <map name="map_example">  
    <area href="https://www.oceanario.pt/" shape="poly" coords="200,100, 320,100, 320,230, 200,230">  
  </map>  
</body>
```



Embedded Content

tags mais utilizadas

<body>...</body>

Embedded Content

■ Imagem

-
- *empty element*
 - **src** atributo **obrigatório**, localização da imagem a inserir
 - Permite um conjunto adicional de atributos
 - **alt** descrição do conteúdo textual da imagem (leitores de ecrã)

```
<body>  
    
  ...  
</body>
```



■ Vídeo

- `<video>...</video>`
- Reprodução de vídeos
 - Minimiza eventuais problemas com a importação de vídeos
 - Dispensa a utilização de *plug-in(s)*
 - Suporta os *containers*:
 - MP4 (*H.264 video codec*) <https://docs.fileformat.com/video/mp4/>
 - WebM (*VP8 video codec*) <https://www.webmproject.org/>
 - Ogg (*Theora video codec*) <https://docs.fileformat.com/audio/ogg/>

Browser	MP4	WebM	Ogg
Edge	YES	YES	YES
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES



■ Reprodução de vídeo

- `<video> ... </video> + <source ...>`
 - **src** : atributo obrigatório, localização do ficheiro a inserir
 - **type** : tipo de vídeo

```
<video width="320" height="240" controls>  
  <source src="Imagens/devstories.webm" type="video/webm"/>  
  <source src="Imagens/devstories.mp4" type="video/mp4"/>  
  
  Tag video não suportada.  
</video>
```

Embedded Content

■ Audio

■ `<audio> ... </audio> + <source ... >`

- **src** : atributo obrigatório, localização do ficheiro a inserir
- **type** : tipo de áudio

```
<audio controls>
  <source src="Imagens/BeautifulDay.mp3" type="audio/mp3"/>
  <source src="Imagens/BeautifulDay.ogg" type="audio/ogg"/>
  Tag audio não suportada.
</audio>
```



Browser	MP3	WAV	OGG
Edge / IE	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES

Embedded Content

■ *iframe*

■ `<iframe> ... </iframe>`

■ *nested browsing context*

- Utilizada para inserir outro documento no documento HTML atual.
- **src**: Localização do conteúdo a embeber (inserir)
- Devem ser formatadas através de CSS



■ <iframe> ... </iframe>

- Criação de uma janela com conteúdo de uma outra aplicação

Exemplo:

inserção direta a partir do
googlemaps de um mapa
com uma localização
específica

Localização ISEC



```
<h2>Localização ISEC</h2>
<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d1077.545330006343!2d-8.411735964765422!3d40.1921889830853!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0xd22f9916a32cfd3%3A0xca4589d604c71bc6!2sInstituto+Superior+de+Engenharia+de+Coimbra!5e0!3m2!1spt-PT!2spt!4v1504091656065" width="400" height="300" frameborder="0" style="border:0" allowfullscreen"></iframe>
```

■ <iframe> ... </iframe>

- **sandbox:** impõe um conjunto de restrições de forma a melhorar a segurança da *iframe*

```
<iframe src="exemplo.html" sandbox></iframe>
```

When the sandbox attribute is present, and it will:

- treat the content as being from a unique origin
- block form submission
- block script execution
- disable APIs
- prevent links from targeting other browsing contexts
- prevent content from using plugins (through <embed>, <object>, <applet>, or other)
- prevent the content to navigate its top-level browsing context
- block automatically triggered features (such as automatically playing a video or automatically focusing a form control)

Value	Description
(no value)	Applies all restrictions
allow-forms	Re-enables form submission
allow-pointer-lock	Re-enables APIs
allow-popups	Re-enables popups
allow-same-origin	Allows the iframe content to be treated as being from the same origin
allow-scripts	Re-enables scripts
allow-top-navigation	Allows the iframe content to navigate its top-level browsing context

Os vários valores
devem ser
separados por
espaço em branco

http://www.w3schools.com/tags/tag_iframe.asp

Formulários

tags mais utilizadas

`<body>...</body>`

Formulários

■ Formulários

- Interpretado pelo browser;
- Informação introduzida pelo utilizador
- Uma vez efetuado o *submit*, os dados são enviados para o servidor
 - Eventualmente é efetuada alguma validação/processamento local
- A informação é processada no lado do servidor;
- É enviada a respetiva resposta ao browser.

The image shows a web form titled "Create an account". It includes a link "or log in" in the top right. The form has four input fields: "First name", "Last name", "Email", and "Password". Below these fields is a checkbox labeled "I agree to Dropbox terms." and a blue button labeled "Create an account". At the bottom, there is a horizontal separator with the word "or" in the center, followed by a Google logo and a blue button labeled "Sign up with Google".

Formulários

- **<form>...</form>**: *container* para todo o formulário
 - Atributos necessários para o processamento dos dados:
 - **action**
 - especifica a localização da aplicação/script destinado ao processamento dos dados.

```
<form action="action.php" method="post">
```

- **method HTTP**

- **Post**: os dados são enviados ao servidor no corpo do *request*, o que torna os dados visíveis apenas pelo servidor. Não tem restrição quanto ao número de caracteres.
- **Get**: os dados são enviados no próprio URL tornando-se assim visíveis. Tem limite de caracteres.

Formulários

Compare GET vs. POST

The following table compares the two HTTP methods: GET and POST.

	GET	POST
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL

...

http://www.w3schools.com/tags/ref_httpmethods.asp

Formulários: Inserção de Dados

tags mais utilizadas

`<body>...</body>`

Formulários: Inserção de Dados

- Três tags para inserção de dados:
 - `<input...>`
 - `<textarea>...</textarea>`
 - `<select>...</select> + <option> ... </option>`
 - a tag `<option>` só pode ser utilizada juntamente com a tag `<select>`

Formulários: Inserção de Dados

■ `<input >`

- A mais versátil, dependendo do valor do atributo `type` pode assumir formas muito diversas para inserção de dados no formulário
- Atributo **`type`** admite um conjunto alargado de valores para definir diversos campos de formulário:

- | | |
|--------------------------------|------------------------------|
| ■ <code>text</code> | ■ <code>number</code> |
| ■ <code>submit</code> | ■ <code>range</code> |
| ■ <code>reset</code> | ■ <code>url</code> |
| ■ <code>password</code> | ■ <code>time</code> |
| ■ <code>radio</code> | ■ <code>date</code> |
| ■ <code>checkbox</code> | ■ <code>image</code> |
| ■ <code>file</code> | ■ <code>color</code> |
| ■ <code>email</code> | ■ <code>...</code> |

Formulários: Inserção de Dados

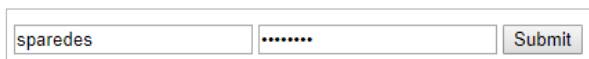
■ Atributo: **`name`**

- Identificador dos campos de formulário para permitir o tratamento dos dados
 - Identificação de cada um dos valores
 - Nesta situação o atributo **`name`** não pode ser substituído pelo atributo **`id`**

*The id of a form input element has nothing to do with the data contained within the element. ID's are for hooking the element with JavaScript and CSS. The name attribute, however, is used in the HTTP request sent by your browser to the server as a **variable name associated with the data contained in the value attribute**.*

<https://teamtreehouse.com/community/what-is-the-difference-between-id-and-name-attributes-in-form-elements>

```
<form>
  <input type="text" name="user" value="sparedes" >
  <input type="password" name="pass" value="abcd1234" >
  <input type="submit" >
</form>
```



Todos os campos para introdução de dados devem ter o atributo `name` definido

Formulários: Inserção de Dados

- valores do atributo *type* utilizados na generalidade dos formulários

- <input type="text" ... >**

- Campo simples de texto (linha única)

- <input type="submit" ... >**

- Uma vez selecionado envia os dados para processamento (*action*)

- <input type="reset" ... >**

- Repõe o estado inicial (estado logo após o download do formulário).

```
<form action="mailto:sparedes@isec.pt">
  Nome: <input type="text" name="user" value="sparedes" >
  <input type="password" name="pass" value="abcd1234" >
  <input type="submit" >
  <input type="reset" >
</form>
```

Nome:

Formulários: Inserção de Dados

- Outros valores do atributo *type*:

- <input type="password" ... >**

- Campo simples de texto (caracteres não visíveis)

```
<form action="mailto:sparedes@isec.pt">
  Nome: <input type="text" name="user" value="sparedes" >
  <input type="password" name="pass" value="abcd1234" >
  <input type="submit" >
  <input type="reset" >
</form>
```

- <input type="radio" ... >**

- Geralmente a seleção é exclusiva (apenas um botão selecionado), para tal os botões têm de ter o mesmo identificador (atributo *name*).

```
<form action="mailto:sparedes@isec.pt">
  <input type="radio" name="r1" >Sim
  <input type="radio" name="r1" >Não
  <input type="submit" >
  <input type="reset" >
</form>
```

☐ Sim ☐ Não

☒ Sim ☐ Não

☐ Sim ☒ Não

Formulários: Inserção de Dados

■ `<input type="checkbox" ... >`

- Deve-se associar o valor (atributo *value*) que resulta de cada uma das opções selecionadas.

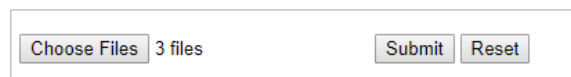
```
form action="mailto:sparedes@isec.pt">
  <input type="checkbox" name="tecweb1" value="1" >HTML <br >
  <input type="checkbox" name="tecweb2" value="2" >CSS <br >
  <input type="checkbox" name="tecweb3" value="3" >jQuery <br >
  <input type="submit"/>
  <input type="reset"/>
</form>
```



■ `<input type="file" ... >`

- Efetuar o *upload* de ficheiros (único ficheiro/múltiplos ficheiros)

```
<form action="mailto:sparedes@isec.pt" method="post">
  <input type="file" multiple >
  <input type="submit" >
  <input type="reset" >
</form>
```

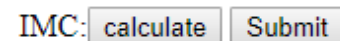


Formulários: Inserção de Dados

■ `<input type="button" ... >`

- Permite definir um botão
- Possui um evento associado (exemplo:click) caso contrário seria um elemento inútil

```
<form action="mailto:sparedes@isec.pt" method="post">
  IMC:<input type="button" value="calculate" >
  <input type="submit" >
</form>
```



Formulários: Inserção de Dados

- Suporte generalizado pelos diferentes browsers

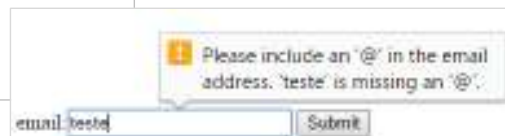


- Quando não suportados, são interpretados como campos de texto.

- **<input type="email"... >**

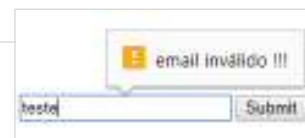
- Campo com validação automática

```
<form action="mailto:sparedes@isec.pt" method="post">
  <input type="email" >
  <input type="submit" >
</form>
```



- É possível alterar a mensagem de erro, associando a execução do método JS **setCustomValidity()** controlado através do evento **oninvalid**

```
<form action="mailto:sparedes@isec.pt" method="post">
  <input type="email" oninvalid="this.setCustomValidity('email inválido !!!')" >
  <input type="submit" >
</form>
```

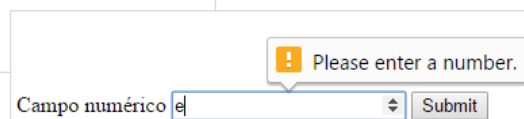


Formulários: Inserção de Dados

- **<input type="number"... >**

- Campo com validação automática

```
<form action="mailto:sparedes@isec.pt" method="post">
  <input type="number" >
  <input type="submit" >
</form>
```



- **<input type="range"... >**

- Utilizado para valores numéricos numa determinada gama
- Dependendo do browser pode ser mostrado como um *slider*

```
<form action="mailto:sparedes@isec.pt" method="post">
  <input type="range" min="0" max="10" >
  <input type="submit" >
</form>
```



Formulários: Inserção de Dados

■ `<input type="url"... >`

- Campo específico para a introdução de um URL válido

```
<form action="mailto:sparedes@isec.pt" method="post">  
  URL:<input type="url" >  
  <input type="submit" >  
</form>
```

URL: teste Submit

■ `<input type="time"... >`

- Inserção de uma hora específica (não permite seleccionar zonas)

```
<form action="mailto:sparedes@isec.pt" method="post">  
  <input type="time" name="t1" >  
  <input type="submit" >  
</form>
```

04:17 PM Submit

Formulários: Inserção de Dados

■ `<input type="date"... >`

- Permite a seleção direta (*datepicker*) de uma data assim como limitar o período admissível para seleção das datas.

■ <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/date>

```
<form action="mailto:sparedes@isec.pt" method="post">  
  <input type="date" name="d1" >  
  <input type="submit" >  
</form>
```

June 2024

Su	Mo	Tu	We	Th	Fr	Sa
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

Today

■ `<input type="image"... >`

- Permite substituir o botão *submit* por uma imagem

```
<form action="mailto:sparedes@isec.pt" method="post">  
  Name:<input type="text" name="n1" > <br><br>  
  <input type="image" src="Imagens/submit.jpg" width="80" height="30" >  
</form>
```

Name: Submit

Formulários: Inserção de Dados

■ Outras *tags* importantes para a definição de formulários:

■ **<textarea></textarea>**

- Área de texto (caixa de texto)
 - utilizada sempre que é necessário introduzir um comentário/sugestão (múltiplas linhas / múltiplas colunas)

```
<form action="mailto:sparedes@isec.pt" method="post">  
  Comentário:<br >  
  <textarea cols="30" rows="10"></textarea><br >  
  <input type="submit" >  
</form>
```

Comentário:

Submit

Formulários: Inserção de Dados

■ **<select> ... </select> + <option> ... </option> + [<optgroup> ... </optgroup>]**

- Lista de seleção (*drop-down*)
- <optgroup> útil quando a lista de opções é extensa, permite melhorar a organização das opções
- Em situação de escolha múltipla é preferível optar por *checkbox*

```
<form action="mailto:sparedes@isec.pt" method="post">  
  Tutorial:  
  <select name="s1">  
    <optgroup label="Client-Side">  
      <option value="1"> HTML</option>  
      <option value="2"> CSS</option>  
      <option value="3"> JavaScript</option>  
    </optgroup>  
    <optgroup label="Client-Side">  
      <option value="4"> PHP</option>  
      <option value="5"> ASP</option>  
    </optgroup>  
  </select>  
  <input type="submit" >  
</form>
```

Tutorial:

HTML ▼

Client-Side

HTML

CSS

JavaScript

Client-Side

PHP

ASP

Submit

Formulários: Organização

tags mais utilizadas

`<body>...</body>`

Formulários: Organização

■ `<label> ... </label>`

- Uma *label* pode ser associada a apenas um elemento (controlo) do formulário
- Associação Implícita
 - Através da inclusão do campo do formulário na própria `<label>`

```
<label>Nome:<input type="text" name="n1" ></label>
```

■ Associação Explícita

- Efetuada através do atributo **for** da *tag label* em conjunto com o atributo **id** do campo de formulário ao qual se pretende associar

```
<label for="idtext">Nome:</label>  
<input type="text" name="n1" id="idtext" >
```

Nome:

Formulários: Organização

■ `<fieldset>...</fieldset> + <legend> ... </legend>`

- Cria áreas distintas (permite uma melhor organização dos campos do formulário)

```
<form action="mailto:sparedes@isec.pt" method="post">
  <fieldset>
    <legend>Client-Side Technologies</legend>
    <input type="checkbox" name="tecweb1" value="1" >HTML <br >
    <input type="checkbox" name="tecweb2" value="2" >CSS <br >
    <input type="checkbox" name="tecweb3" value="3" >jQuery <br >
  </fieldset>
  <fieldset>
    <legend>Server-Side Technologies</legend>
    <input type="checkbox" name="tecweb4" value="4" >PHP <br >
    <input type="checkbox" name="tecweb5" value="5" >ASP <br >
  </fieldset>
  <input type="submit" >
</form>
```

Client-Side Technologies

☐ HTML

☐ CSS

☐ jQuery

Server-Side Technologies

☐ PHP

☐ ASP

Submit

Formulários (tags)

■ Início/Fim do formulário

```
<form>...</form>
```

■ Introdução de dados

```
<input .... >
<textarea></textarea>
<select></select>
```

■ Legenda/Organização

```
<label></label>
<fieldset></fieldset>
```

Formulários: Funcionalidade

tags mais utilizadas

`<body>...</body>`

Formulários: Funcionalidade

- Atributos tag **`<input>`**
 - A tag input possui um conjunto de atributos muito úteis, os quais permitem definir de forma direta alguns aspetos funcionais
 - `<input type="text" value="..." >`
 - Valor inicial do campo
 - Quando o é efetuado o download do formulário o valor é assumido por defeito.

```
<form action="">
  <input type="text" value="valorInicial" >
  <input type="submit" >
  <input type="reset" >
</form>
```

Value:

Formulários: Funcionalidade

■ Atributos tag `<input>`

■ `<input type="text" maxlength="..." >`

- Especifica o valor máximo de caracteres que o campo pode receber
- Diferente do atributo **size** usado (originalmente) para definir o tamanho do campo

```
<form action="">
  Value: <input type="text" size="15" maxlength="4" ><br ><br >
  <input type="submit" >
  <input type="reset" >
</form>
```

Value:

Formulários: Funcionalidade

■ `<input type="value"... min="..." max="...">`

- Permite especificar o valor máximo e mínimo para alguns `<input type="...">`
 - number, range, date, datetime, datetime-local, month, time and week
 - por exemplo, aplicados a um datepicker permite definir um intervalo de datas possíveis de seleccionar

```
<form action="mailto:sparedes@isec.pt">
  <input type="date" name="d1" min="2022-10-04" max="2022-10-08">
</form>
```

10/04/2022

October 2022

Su	Mo	Tu	We	Th	Fr	Sa
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

Today

Formulários: Funcionalidade

■ `<input ...step="...">`

- Permite definir os valores válidos com base em intervalos fixos

```
<form action="mailto:sparedes@isec.pt" method="post" id="form1">  
  <input type="number" step="3" min="1" max="20" >  
  <input type="submit" ><br >  
</form>
```

■ `<input ... multiple >`

- Para alguns `<input type=" " />` (email, file) permite a introdução de vários valores/ficheiros

```
<form action="mailto:sparedes@isec.pt" method="post" id="form1">  
  <input type="file" multiple >  
  <input type="submit" >  
</form>
```

Formulários: Funcionalidade

■ `<input ...pattern="[caracteres]{número}">`

- Permite estabelecer um padrão para os dados de entrada
 - Um primeira validação quando os dados verificam um padrão pré definido (exemplo: código postal)
 - O padrão mais simples, permite definir entre parêntesis retos qual a gama admissível de caracteres [...] e entre chavetas {...} o número de caracteres esperado
 - *Regular Expressions* http://www.w3schools.com/jsref/jsref_obj_regexp.asp

```
<form action="mailto:sparedes@isec.pt" method="post" id="form1">  
  <input type="text" pattern="[A-Za-z]{3}" >  
  <input type="submit" ><br >  
</form>
```

Formulários: Funcionalidade

- `<input ...pattern="[caracteres]{número}" >`
 - Regular Expressions http://www.w3schools.com/jsref/jsref_obj_regexp.asp
 - Exemplos:

```
[012345679]    //any single digit
[0-9]          //also any single digit
[a-z]          //any single lower case letter
[a-zA-Z]       //any single letter
```

The character (^) can be used to negate matches
For example:

```
[^0-9]         //any single non-digit
[^aeiouAEIOU] //any single non-vowel
```

<https://users.cs.cf.ac.uk/Dave.Marshall/PERL/node79.html>

Formulários: Funcionalidade

- `<input ...required>`
 - Campo obrigatório! Muito utilizado para garantir a inserção de valores.

```
<form action="mailto:sparedes@isec.pt" method="post" id="form1">
  username:<input type="text" name="user" required >
  <input type="submit" >
</form>
```

username:

! Please fill out this field.

- `<input ...placeholder="..." >`
 - Informação suplementar para a inserção de valores, é eliminado quando se inicia a inserção dos dados
 - Solução muito utilizada para reduzir a complexidade visual dos formulários

```
<form action="mailto:sparedes@isec.pt" method="post" id="form1">
  <input type="text" placeholder="First Name" >
  <input type="text" placeholder="Last Name" >
  <input type="submit" >
</form>
```

First name
Last name

Simão
Last name

■ Atributos **<input>**

■ **<input ... form="..." >**

- Mesmo fora dos limites do formulário um campo **input** pode ser associado ao formulário
- Com base no atributo **form** cujo valor deve coincidir com o valor do atributo **id** do formulário ao qual se pretende associar o campo de formulário

```
<form action="mailto:sparedes@isec.pt" method="post" id="form1">  
  <input type="text" name="first" >  
  <input type="submit" >  
</form>  
  <input type="text" name="last" form="form1" >
```

Simao	Submit
Paredes	

Send	Subject	
first=Simao&last=Paredes		

Botões

tags mais utilizadas

`<body>...</body>`

■ **<button>**

- Permite definir um botão ao qual é associado um dado evento
- Muito usado por event listeners para despoletar um determinado processamento
- O atributo type deve ser sempre definido
 - button
 - reset
 - submit

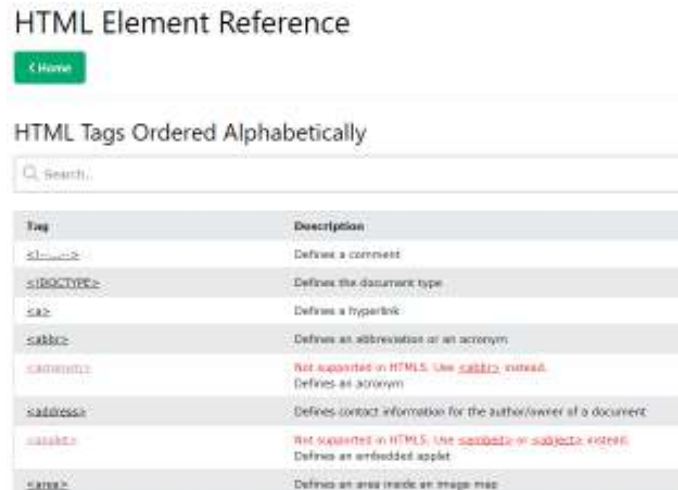
```
<button type="button" class="button button1" onclick="alert('Tag Button')">First Button</button>
```



Listagem completa de *tags* HTML

Listagem completa de *tags* HTML

■ <http://www.w3schools.com/tags>



Tag	Description
<!-- -->	Defines a comment
<DOCTYPE -->	Defines the document type
<a -->	Defines a hyperlink
<abbr -->	Defines an abbreviation or an acronym
<acronym -->	Not supported in HTML5. Use <abbr --> instead. Defines an acronym
<address -->	Defines contact information for the author/owner of a document
<applet -->	Not supported in HTML5. Use <object --> instead. Defines an embedded applet
<area -->	Defines an area inside an image map

- Atualmente, a generalidade das *tags* dedicadas a aspetos de formatação (originalmente assegurado pelo HTML) são consideradas *deprecated*.
 - Toda a formatação **deve ser assegurada via CSS**

WEB API

Application Programming Interface

Application Programming Interface

■ Integração de API

- Na realidade o HTML para além de ser uma markup language, encontra-se associado a um conjunto de funcionalidades encapsuladas em API's suportadas nativamente pelos *browsers*, as quais podem ser acedidas via *JavaScript*.

"An application programming interface (API) is a protocol intended to be used as an interface by software components to communicate with each other. An API is a library that may include specification for routines, data structures, object classes and variables."



<http://html5index.org/>

Application Programming Interface

■ Geolocation API

The `getCurrentPosition()` Method - Return Data

The `getCurrentPosition()` method returns an object on success. The latitude, longitude and accuracy properties are always returned.

Property	Returns
<code>coords.latitude</code>	The latitude as a decimal number (always returned)
<code>coords.longitude</code>	The longitude as a decimal number (always returned)
<code>coords.accuracy</code>	The accuracy of position (always returned)
<code>coords.altitude</code>	The altitude in meters above the mean sea level (returned if available)
<code>coords.altitudeAccuracy</code>	The altitude accuracy of position (returned if available)
<code>coords.heading</code>	The heading as degrees clockwise from North (returned if available)
<code>coords.speed</code>	The speed in meters per second (returned if available)
<code>timestamp</code>	The date/time of the response (returned if available)

https://www.w3schools.com/html/html5_geolocation.asp

Application Programming Interface

```
<button onclick="getLocation()">Try it!</button>  
<p id="coord">Coordinates!</p>
```

1. Quando o botão é clicado, é chamada a função *getLocation()*

```
<script>  
  var x=document.getElementById('coord');  
  
  function getLocation(){  
    if (navigator.geolocation)  
      { navigator.geolocation.getCurrentPosition(function(position){  
        x.innerHTML="latitude:" + position.coords.latitude + "<br>" +  
        "longitude:" + position.coords.longitude;  
      })  
    }  
    else { x.innerHTML="Geolocation is not supported!"; }  
  }  
</script>
```

2. Se a API geolocation é suportada é executado o método *getCurrentPosition()* caso contrário mostra mensagem

3. *getCurrentPosition()* retorna um objeto *position* o qual contém a latitude e longitude (browser)

Try it!

Coordinates!

Try it!

latitude:40.1923479
longitude:-8.4126341

Application Programming Interface

■ Exemplos de algumas API's muito utilizadas:

■ *Web Storage*

- Armazenamento local (browser). Permite guardar mais informação que os *cookies*

■ *Offline Web Application*

- Aplicação web é armazenada em cache o que a torna acessível mesmo sem uma ligação à internet (offline).

■ *Drag & Drop*

- Possibilidade de arrastar o um conteúdo específico para uma outra área na página ou para outra página.

■ ...

data- Attributes*

data-*

- Atributo usado para armazenar/embeber dados num elemento HTML
 - Dados acessíveis via JS **sem necessidade de qualquer pedido** adicional ao servidor

```
<h1>Species</h1>
<p>Click on a species:</p>
<ul>
  <li onclick="showDetails(this)" data-animal-type="bird" data-animal-size="small">Sparrow</li>
  <li onclick="showDetails(this)" data-animal-type="fish" data-animal-size="medium">Salmon</li>
  <li onclick="showDetails(this)" data-animal-type="spider" data-animal-size="big">Tarantula</li>
</ul>
```

JS

```
<script>
  function showDetails(animal) {
    var animalType = animal.getAttribute("data-animal-type");
    var animalSize=animal.getAttribute("data-animal-size");
    alert("The " + animal.innerHTML + " is a " + animalSize + " " + animalType + ".");
  }
</script>
```

- O atributo data-* consiste em duas partes.
 - Não deve ter qualquer carater maiúsculo e devem ter pelo menos um caracter a seguir ao hífen
 - O valor do atributo pode ser uma string (sem qualquer restrição)

<li onclick="showDetails(this)" data-animal-type="bird" data-animal-size="small">Sparrow

<li onclick="showDetails(this)" data-animal-type="fish" data-animal-size="medium">Salmon

<li onclick="showDetails(this)" data-animal-type="spider" data-animal-size="big">Tarantula



Clean HTML

Clean HTML

■ Alguns Princípios

■ Indentar código

- Fundamental para uma percepção imediata da organização/estrutura aumentando dessa forma a legibilidade do código

X

```
<ol>
  <li>Item1</li>
  <li>Item2
    <li>Item2.1</li>
    <li>Item2.2</li>
  </li>
</ol>
```

Ok!

```
<ol>
  <li>Item1</li>
  <li>Item2
    <ul>
      <li>Item2.1</li>
      <li>Item2.2</li>
    </ul>
  </li>
</ol>
```

■ Consistente ao longo de todo o código

- aspas/plicas
- mesmo espaço na indentação
- para a designação dos atributos id/class usar hífen (***_***) ou underscore (***_***)
-

Clean HTML

■ Optimizar a utilização dos div's

- Frequentemente a tendência é sobreutilizar os containers div para segmentar o código, o que impossibilita a criação de código mais estruturado

X

```
<div id="content">
  <div class="headline">Headline</div>
  <div class="subtitle">Subtitle</div>
  <div class="post">Post Content</div>
  <div class="list">
    <ul>
      <li>Element 1</li>
      <li>Element 2</li>
    </ul>
  </div>
</div> <!-- end content div -->
```

Ok!

```
<div id="content">
  <h1>Headline</h1>
  <h2>Subtitle</h2>
  <p>Post Content</p>
  <ul>
    <li>Element 1</li>
    <li>Element 2</li>
  </ul>
</div>
```

■ Reduzir o número de comentários

- <!-- -->

"..., comments aren't necessary because HTML markup is very much self-explanatory. If you find yourself commenting your HTML a lot, you should review the HTML"

■ Atributos *id/class*

- Nomes com significado: curtos, descritivos e devem representar apenas 1 conceito

- `id="myHeader"`
- `id="sitenavigation"`
- `class="searchField"`
-

■ Garantir o espaçamento do código

- eliminar espaços conduz necessariamente a código mais confuso

X

```
<body>
<h1>Table</h1><table><tr><th>Data 1</th><th>Data 2</th></tr>
<tr><td>Calcutta</td><td>Orange</td></tr></table></body>
```

Ok!

```
<body>
<h1>Table</h1>
<table>
  <tr>
    <th>Data 1</th>
    <th>Data 2</th>
  </tr>
  <tr>
    <td>Calcutta</td>
    <td>Orange</td>
  </tr>
</table>
</body>
```

Optimização HTML

Optimização HTML

- Reduzir o *page weight do HTML*
 - Este é sem dúvida o parâmetro mais importante para otimizar o tempo de download
 - mover inline scripts para ficheiro externo
 - mover CSS para ficheiro externo
 - minimização do ficheiro (remover espaços em branco desnecessários, comentários,)
 - Utilização de ferramentas para limpeza de código
 - <https://designmodo.com/tools-clean-code/>



<https://www.html-tidy.org/>

Optimização HTML

- Minimizar o número de ficheiros
 - A redução do número de ficheiros referenciados por uma aplicação web reduz o número de ligações HTTP para efetuar o seu download.
 - Reduz o tempo para enviar os *HTTP requests* e receber as respetivas *HTTP responses*
 - Na incorporação de imagens de pequena dimensão (logos), deve-se recorrer a CSS image sprites
 - 1 HTTP Request (evita múltiplos requests) e eventual ajuste local das coordenadas



<https://www.formget.com/css-image-sprites/>

Optimização HTML

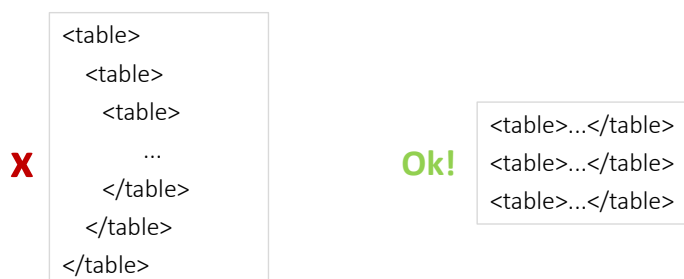
- Redução de *inline scripts*
 - Reduzir o número de scripts embebidos diretamente no HTML
- Usar *valid markup*
 - Evita a necessidade do browser ter de proceder a correções na interpretação do HTML
 - Possibilita a utilização de ferramentas para limpeza de código as quais não podem ser utilizadas se detetado markup inválido
 - Novas funcionalidades CSS permite reduzir algum markup utilizado para construção de layouts



<https://jsonformatter.org/html-validator>

Optimização HTML

- Minimizar encadeamentos de conteúdo
 - Evitar de todo construir um layout com base em tabelas
 - A utilização de tabelas deve ficar limitada à apresentação de dados tabulares, garantindo a independência entre tabelas diferentes



Optimização HTML

■ Minimificar e Comprimir os recursos (assets)

- exemplo: Comprimir ficheiros de imagem para reduzir o peso do ficheiro
- exemplo: “Minimificar” os SVG
 - Remoção de todos os metadados incluídos num ficheiro SVG e que não são absolutamente necessários para a sua visualização <https://www.svgminify.com/>



Optimização HTML

■ Especificação das dimensões de imagens e tabelas

- Permite ao browser a interpretação direta destes elementos sem necessidade de um reflow (reajuste) de todo o conteúdo.

- A especificação das dimensões deve ser efetuada via CSS

■ Ponderar a utilização de *lazy loading images*

- Por defeito, todas as imagens são obtidas e “renderizadas” de acordo com a interpretação do fluxo HTML.
- As *lazy loading images* são apresentadas apenas quando são efetivamente necessárias para ser formado o conteúdo a visualizar no viewport

- <https://imagekit.io/blog/lazy-loading-images-complete-guide/>

```
<img href="./images/footerlogo.jpg" loading="lazy">
```

■ cross-browser

- Considerar os browsers mais modernos e relevantes, ou seja as mais recentes versões de:

- Google Chrome



- Edge



- Firefox



- Opera



- Safari



Example page structure

- `<html>`
- `<head>`
 - `<link>` ...
CSS files required for page appearance. Minimize the number of files for performance while keeping unrelated CSS in separate files for maintenance.
 - `<script>` ...
JavaScript files for functions **required** during the loading of the page, but not any interaction related JavaScript that can only run after page loads. Minimize the number of files for performance while keeping unrelated JavaScript in separate files for maintenance.
- `<body>`
 - User visible page content in small chunks (`<header>` / `<main>` / `<table>`) that can be displayed without waiting for the full page to download.
 - `<script>` ...
Any scripts which will be used to perform interactivity. Interaction scripts typically can only run after the page has completely loaded and all necessary objects have been initialized. There is no need to load these scripts before the page content. That only slows down the initial appearance of the page load.
Minimize the number of files for performance while keeping unrelated JavaScript in separate files for maintenance.
If any images are used for rollover effects, you should preload them here after the page content has downloaded.

- Usar um Content Delivery Network (CDN)
 - Reduzir a distância entre cliente e servidor
 - Reduz a latência logo reduz o tempo de download
 - CDN providers: <https://wpforms.com/best-cdn-providers/>
 - Não se aplica/justifica em situações de acesso local



<https://www.imperva.com/learn/performance/what-is-cdn-how-it-works/?redirect=incapsula>

Limitações HTML

- O HTML não assegura dois aspetos essenciais, uma vez que a tecnologia não se destina a essa finalidade:
 - Consistência/Limitação na formatação dos conteúdos.
 - *Cascading Style Sheets (CSS)*
 - Criação de ambientes mais dinâmicos nas suas várias perspetivas: geração de conteúdo, interatividade com o utilizador, ...
 - *JavaScript, frameworks JavaScript, ...*