

# Introdução às tecnologias Web - ITW

Aula 2 – Listas, Estilos (CSS), Fontes & Formulários HTML

# Sumário

Marcadores HTML

Listas

Formulários

*Entrada de dados*

*Usabilidade e acessibilidade*

Estilos

Introdução às Cascading Stylesheet's (CSS's)

Tipos de letra - Fontes



# Estrutura base de um documento html

[relembrando...]

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="Author" content="Joaquim Sousa Pinto">
    <meta name="Keywords" content="exemplos">
    <meta name="Description" content="Exemplos das aulas">
    <title>Titulo do Documento</title>
  </head>
  <body>

  </body>
</html>
```

Cabeçalho do documento <head></head>  
[Este conteúdo não é representado]

Conteúdo do documento <body></body>  
[Este conteúdo é representado e é aqui que  
deve ser inserido o código HTML criado]

# Características de um elemento html

[relembrando...]

Um marcador / etiqueta (tag em inglês) num documento html fica sempre colocada entre os símbolos "<" e ">". As etiquetas são responsáveis pela formatação da linguagem

Nas últimas versões da linguagem, um elemento é constituído por um par de início e de fim de marcador.

**<marcador>texto</marcador>**

marcador de  
início do  
elemento

Marcador de fim do  
elemento

# Características de um elemento

[relembrando...]

Um elemento é formado por um nome de **etiqueta** (tag), **atributos**, **valores** e **filhos** (que podem ser outros elementos ou texto).

Os atributos modificam os valores padrão dos elementos e os valores caracterizam essa mudança.

Exemplo de um elemento simples (não possui filhos):

`<hr/>`

Exemplo de um elemento com atributos:

`<a href="http://www.ua.pt/">Universidade de Aveiro</a>`

Os atributos são sempre colocados dentro do marcador inicial

Exemplo de um elemento com filhos e atributos:

`<p>A <a href="http://www.ua.pt/">Universidade de Aveiro</a> é a minha Universidade.</p>`

Os elementos filho deverão estar completamente inseridos dentro do elemento pai

# Marcadores HTML

## Listas

# Listas

Listas ordenadas: `<ol>...</ol>`

Criar listas ordenadas – Marcador `<ol>`

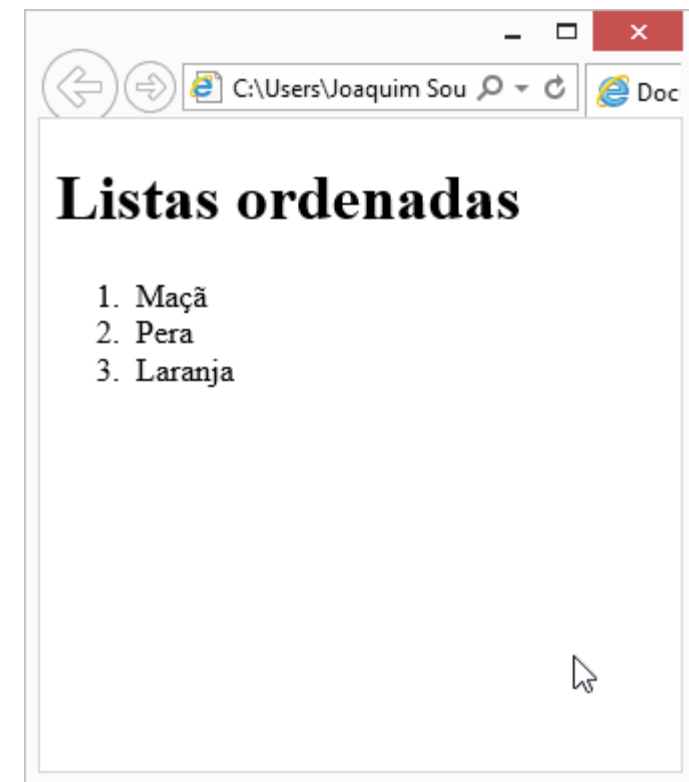
Elementos da lista – Marcador `<li>`

```
<h1>Listas Ordenadas</h1>
<ol type="1">
  <li>Maçã</li>
  <li>Pera</li>
  <li>Laranja</li>
</ol>
```

Tipo de marcador da lista – Atributo `type`

Valores possíveis: 1, A, a, I, i

<https://jsfiddle.net/JoaquimSousaPinto/3z0oqyd1/>



# Listas

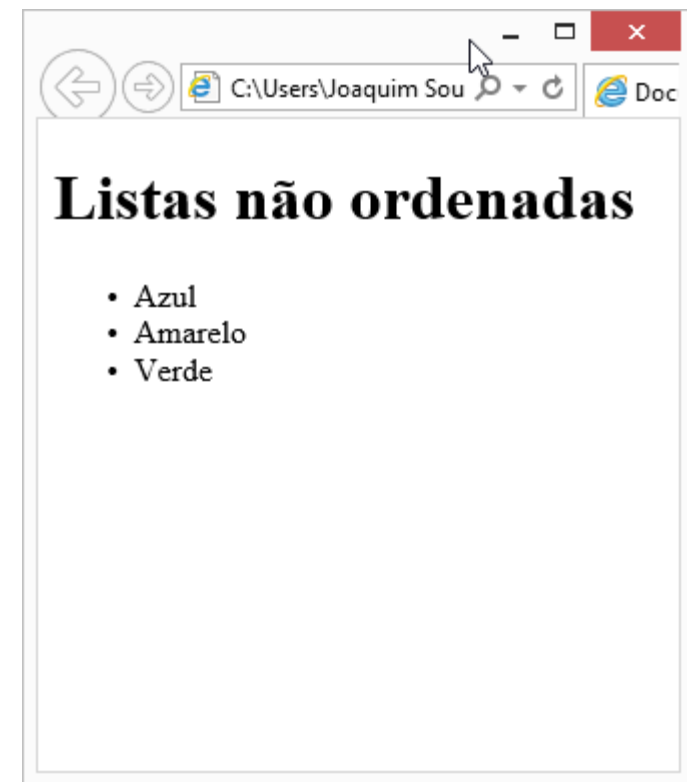
Listas não ordenadas: `<ul>...</ul>`

Criar listas não ordenadas – Marcador `<ul>`  
Elementos da lista – Marcador `<li>`

```
<h1>Listas Não Ordenadas</h1>
<ul type="disc">
  <li>Azul</li>
  <li>Amarelo</li>
  <li>Verde</li>
</ul>
```

Tipo de marcador da lista – Atributo `type`  
Valores possíveis: `disc`, `circle`, `square`

<https://jsfiddle.net/JoaquimSousaPinto/1d5qf05h/>





# Listas

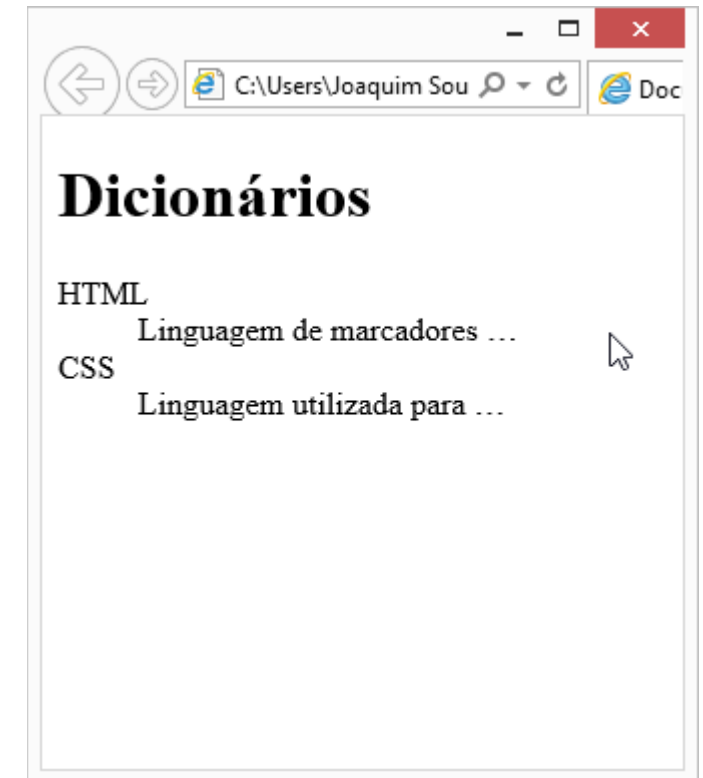
Dicionários / Listas de definições: `<dl>...</dl>`

Criar lista de definições – marcador: `<dl>`

Marcador para identificação do termo: `<dt>`;

Marcador para a definição do termo: `<dd>`

```
<dl>
  <dt>HTML</dt>
  <dd>Linguagem de marcadores ...</dd>
  <dt>CSS</dt>
  <dd>Linguagem utilizada para ...</dd>
</dl>
```



# Listas

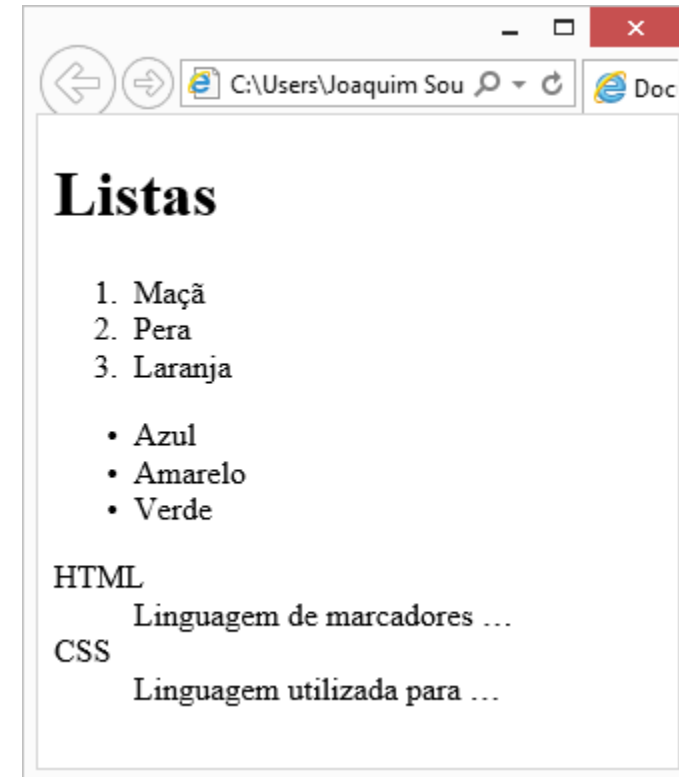
## Exemplos

```
<h1>Listas</h1>
<ol type="1">
  <li>Maçã</li>
  <li>Pera</li>
  <li>Laranja</li>
</ol>

<ul type="disc">
  <li>Azul</li>
  <li>Amarelo</li>
  <li>Verde</li>
</ul>

<dl>
  <dt>HTML</dt>
  <dd>Linguagem de marcadores ...</dd>
  <dt>CSS</dt>
  <dd>Linguagem utilizada para ...</dd>
</dl>
```

<https://jsfiddle.net/JoaquimSousaPinto/juavx9fz/>



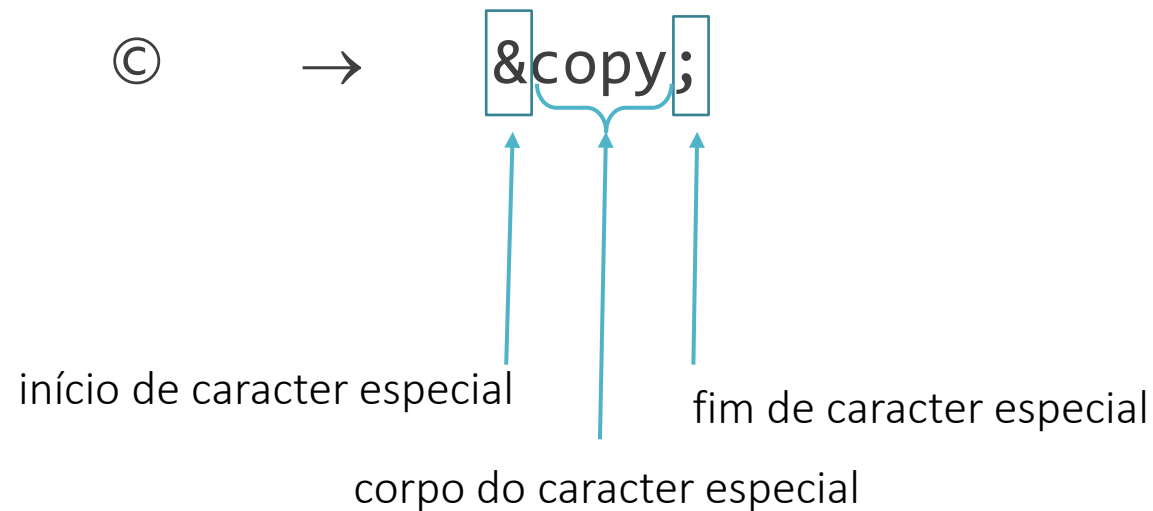
# Marcadores HTML

## Caracteres Especiais

# Representação de caracteres especiais em HTML

Há um conjunto de caracteres que, ou não possuem representação direta, ou não pertencem a todos os alfabetos, por isso precisam de uma forma especial de representação.

Exemplo:



# Representação de caracteres especiais em HTML

Caracteres & símbolos matemáticos: →

Outros símbolos:

Char	Number	Entity	Description
©	&#169;	&copy;	COPYRIGHT SIGN
®	&#174;	&reg;	REGISTERED SIGN
€	&#8364;	&euro;	EURO SIGN
™	&#8482;	&trade;	TRADEMARK
←	&#8592;	&larr;	LEFTWARDS ARROW
↑	&#8593;	&uarr;	UPWARDS ARROW
→	&#8594;	&rarr;	RIGHTWARDS ARROW
↓	&#8595;	&darr;	DOWNWARDS ARROW
♠	&#9824;	&spades;	BLACK SPADE SUIT
♣	&#9827;	&clubs;	BLACK CLUB SUIT
♥	&#9829;	&hearts;	BLACK HEART SUIT

Char	Number	Entity	Description
∀	&#8704;	&forall;	FOR ALL
∂	&#8706;	&part;	PARTIAL DIFFERENTIAL
∃	&#8707;	&exist;	THERE EXISTS
∅	&#8709;	&empty;	EMPTY SETS
∇	&#8711;	&nabla;	NABLA
∈	&#8712;	&isin;	ELEMENT OF
∉	&#8713;	&notin;	NOT AN ELEMENT OF
⊃	&#8715;	&ni;	CONTAINS AS MEMBER
∏	&#8719;	&prod;	N-ARY PRODUCT
Σ	&#8721;	&sum;	N-ARY SUMMATION

Fonte: [http://www.w3schools.com/html/html\\_symbols.asp](http://www.w3schools.com/html/html_symbols.asp) (ver mais)

Outros símbolos: <http://www.sabinanore.com/design/html-special-symbols/>

# CSS – Cascading Style Sheets

## Introdução

# CSS – Cascading Style Sheets

O que é?

CSS é o acrónimo de Cascading Style Sheets, ou em português livre, folhas de estilos encadeados.

**Os estilos CSS permitem fazer a separação entre a estrutura do documento HTML e a sua representação.**

*A linguagem HTML define que um conjunto de elementos estruturais de um documento*

*Exemplos: um cabeçalho de nível 1 é representado por (<h1> </h1>) ou um parágrafo é representado por (<p> </p>);*

*A linguagem CSS controla as fontes, cores, margens, linhas, alturas, larguras, imagens de fundo, posicionamento, entre muitos outros, de todos os elementos html.*

# CSS – Cascading Style Sheets

## Origem / necessidade

Tal como foi referido na primeira aula, a linguagem HTML foi criada para publicação e disseminação de informação científica.

Para isso, foram desenvolvidos um conjunto de marcadores que se preocupavam muito mais com a semântica e estrutura do documento que com a sua forma de representação.

Com a massificação da sua utilização (fora do contexto original) a WWW ganhava popularidade.

Mas o resultado não satisfazia e, ...

... os designers começavam a sentir a necessidade de encontrar meios de representar a informação de forma mais atrativa

*novos tipos de letra, cores, imagens, ...*



# CSS – Cascading Style Sheets

## Origem / necessidade

Foram criados novos marcadores HTML tais como, por exemplo, o marcador `<font>`, `<div>` e `<span>`.

Exemplo análogo ocorreu com o marcador `<table>`, que era destinado a representar informação tabular e que passou a ser utilizado para a definição do layout da página e não para a representação de informação na forma tabular – tal como fizemos na aula prática 1.

As CSS's vieram trazer ordem à confusão entretanto criada colocando à disposição dos web designers meios sofisticados para projetar layouts.

Assim foi possível manter uma separação entre os elementos de representação da estrutura dos documentos (p, div, span, li, etc...) e a sua representação (amarelo, grande, à esquerda, ...), facilitando a manutenção dos web sites. É assim possível que um mesmo documento, quando submetido a diferentes estilos, seja representado de forma completamente distinta.

# CSS – Cascading Style Sheets

## Formas de definição e hierarquia

As instruções CSS podem ser definidas de três formas distintas:

**Global** – colocadas num ficheiro externo que pode depois ser associado a um ou mais documentos html.

**Document** – colocadas dentro de um marcador `<style></style>` localizado no `<head>` do documento;

**In-line** – colocadas na linha do marcador html;

A precedência é Global / Document / Inline, ou seja, a instrução que prevalece é a que estiver mais próxima do elemento.

# CSS – Cascading Style Sheets

## Notação

### In-line

```
<marcador style="propriedade : valor; propriedade : valor;">
```

### Document / Global

```
seletor {propriedade : valor; propriedade : valor;}
```

# CSS – Cascading Style Sheets

## Notação - Exemplos

### Inline

```
<div style="background-color: #00FF00;">
```

### Document

```
<html>
  <head>
    <style type="text/css">
      div { background-color: #FF0000; }
    </style>
  </head>
  ...
```

### Global

Documento html

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="style.css" />
  </head>
  ...
```

Ficheiro style.css

```
div { background-color: #FF0000; }
```

# Exercício 4

## CSS – Cascading Style Sheets - Hierarquia / Precedência

style.css

```
body {background-color: #FF0000;}
```

teste.html

```
<html>
  <head>
    <title>Exemplo</title>
    <link href="style.css" rel="stylesheet" />
    <style type="text/css">
      body {background-color: #00FF00;}
    </style>
  </head>
  <body style="background-color: #0000FF;">
    <p>De que cor é o fundo deste documento?</p>
  </body>
</html>
```

Respondam a esta pergunta ...

# CSS – Cascading Style Sheets

## Cores

A propriedade `color` define a cor de um elemento.

Exemplo 1:

Documento / Global: `p { color : #F0FFFF; }`

In-line: `<p style="color : #F0FFFF">...</p>`

- Os primeiros dois símbolos no código de cor HTML representam a intensidade da cor vermelho – 00 é o mínimo e FF é o mais intenso.
  - O terceiro e o quarto símbolos representam a intensidade de verde;
  - O quinto e o sexto símbolos representam a intensidade de azul.
- 
- Nem todas as cores são representadas na Web. O conjunto de cores representáveis é denominado por “safe colors” / “cores seguras”.
  - Uma tabela com os nomes e códigos destas colors está disponível em [http://www.flextool.com.br/tabela\\_cores.html](http://www.flextool.com.br/tabela_cores.html), visitado em 07Out2020

# CSS – Cascading Style Sheets

## Cores

A propriedade `color` define a cor de um elemento.

...

Exemplo2:

Documento / Global: `p { color : rgb(240,255,255); }`

In-line: `<p style="color : rgb(240,255,255)">...</p>`

- Neste caso a cor é representada na forma decimal através da função `rgb(rr,gg,bb)`.
- Há ainda uma forma similar de representação `rgba(rr,gg,bb, tt)`, em que `tt` é a transparência e pode variar entre 0.0 (transparente) e 1.0 (opaco)
- As cores são separadas por uma vírgula
- Testar cores em <http://www.css3maker.com/css-3-rgba.html>, visitado em 07out2020

# CSS – Cascading Style Sheets

## Cores

A propriedade `color` define a cor de um elemento.

...

### Exemplo 3:

Documento / Global: `p { color: Azure1; }`

In-line: `<p style="color: Azure1">...</p>`

- Neste caso a cor é definida pelo seu nome.
  - Lista de nomes de cores: <http://www.tedmontgomery.com/tutorial/clrnmsWH.html>, visitado em 07out2020
- Nem todos os browsers interpretam as cores pelo seu nome.



# CSS – Cascading Style Sheets

## Fundos

Propriedades:

**background-color**

**background-image** – `url("url da imagem")`

Exemplo:

```
div { background-image: url("http://grungetextures.com/sample/91/gray-brick-background.jpg"); }
```

**background-repeat**

`background-repeat: repeat-x` – repete-se na horizontal

`background-repeat: repeat-y` – repete-se na vertical

`background-repeat: repeat` – repete-se tanto na horizontal como na vertical

`background-repeat: no-repeat` – não se repete em qualquer direção

# CSS – Cascading Style Sheets

## Fundos

### **background-attachment**

Background-attachment: scroll - imagem move-se quando é feito o arrastamento

Background-attachment: fixed - imagem fica fixa quando é feito o arrastamento

### **background-position**

background-position: 2cm 2cm – imagem a 2 cm da esquerda e 2 cm para baixo na página

background-position: 50% 25% a – imagem centrada na horizontal e a um quarto (25%) para baixo na página

background-position: top right – imagem é posicionada no canto superior direito

# CSS – Cascading Style Sheets

## Fundos

É ainda possível representar um background combinando as diversas partes do mesmo.

Exemplo:

```
div { background: #FFCC66 url("http://grungetextures.com/sample/91/gray-brick-background.jpg") no-repeat; }
```



# CSS – Cascading Style Sheets

## Fontes de texto

### Propriedade font-family

A propriedade font-family é usada para definir a lista das fontes a utilizar num marcador e qual a sua prioridade para apresentação.

*Se a primeira fonte da lista não estiver instalada, deverá ser usada a segunda e assim por diante até ser encontrada uma fonte instalada.*

### Exemplo:

```
body {font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif}  
h1 {font-family: arial, verdana, sans-serif;}  
h2 {font-family: 'Times New Roman', serif;}
```

#### Perguntas:

- por que razão se pode/deve utilizar uma lista de fontes e não apenas uma?
- porque há nomes entre aspas e outros sem nada?

# CSS – Cascading Style Sheets

## Fontes de texto

### Propriedade

`font-style` - normal | italic | oblique | initial | inherit;

`font-variant` - normal | small-caps | initial | inherit;

`font-weight` - normal | bold | bolder | lighter | (100-900) | initial | inherit;  
300 = light; 400 = normal; 700 = bold;

`font-size`

1. medium | xx-small | x-small | small | large | x-large | xx-large | smaller | larger | initial | inherit
2. valor numérico (10px, 8pt, 1.2cm, ...)
3. % - percentagem relativamente ao element anterior (element pai) (80%, 75%, ...)

# CSS – Cascading Style Sheets

## Fontes de texto

Exemplo de um estilo CSS na forma expandida

```
p {  
  font-style: 1em;  
  font-weight: normal;  
  font-size: 12px;  
  font-family: 'Segoe UI', sans-serif;  
}
```

Exemplo de um estilo CSS na forma reduzida

```
p { font: 1em normal 12px 'Segoe UI', sans-serif; }
```



# CSS – Cascading Style Sheets

alinhamento – horizontal e vertical

## Propriedade

**align** - left | right | center | justify;

**valign** - top | base | middle | bottom;

## Exemplo:

```
td {  
    align: center;  
    valign: top;  
}
```

# CSS – Cascading Style Sheets

alinhamento – largura e altura

## Propriedade

**width** – 100% | 800px | 600pt | 1.2em;

**height** – 100% | 600px | 200pt | 1.2em;

## Exemplo de um estilo CSS

```
td {  
    width: 100px;  
    height: 100px;  
}
```



# Elementos, classes e Id's

Qual a diferença?

Para descrever o conteúdo em um documento utilizamos marcadores básicos como `<h1>`, `<p>`, `<td>`, `<ul>`, etc.

Contudo, esse conjunto básico de marcadores não cobre todos os tipos possíveis de elementos da página ou escolha de layout.

Para isso, precisamos de **IDs** e **classes**.

Por exemplo para definir um rodapé, podemos fazer algo como isto: `<div id="footer">`.

Ou se quisermos caixas para manter o conteúdo separado de alguma forma: `<div class="sidebar-box">`.

# Elementos, classes e Id's

## Qual a diferença?

Esses **IDs** e **classes** são os “elos de ligação” que precisamos de utilizar na marcação para os podemos modelar da forma pretendida.

O CSS precisa desses elos de ligação para construir seletores e fazer os nossos estilos, mas outras linguagens como o JavaScript, também dependem deles.

Mas qual é a diferença entre **IDs** e **classe**?

Os IDs são únicos

*Cada elemento pode ter apenas um ID*

*Cada página pode ter apenas um elemento com esse ID*

As classes não são únicas

*Pode-se usar a mesma classe em vários elementos.*

*Pode-se usar várias classes no mesmo elemento.*

# Elementos, classes e Id's

Qual a diferença?

## Combinações de classes e IDs

O grande ponto aqui é que é possível direcionar os elementos que têm combinações de classes e IDs ao encadear esses seletores sem espaços.

## ID e seletor de classe:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    #one { color: blue; }
    #one.two { color: red; }
  </style>
</head>
<body>
  <h1 id="one">This Should Be Blue</h1>
  <h1 id="one" class="two">This Should Be Red</h1>
</body>
</html>
```



# Elementos, classes e Id's

Qual a diferença?

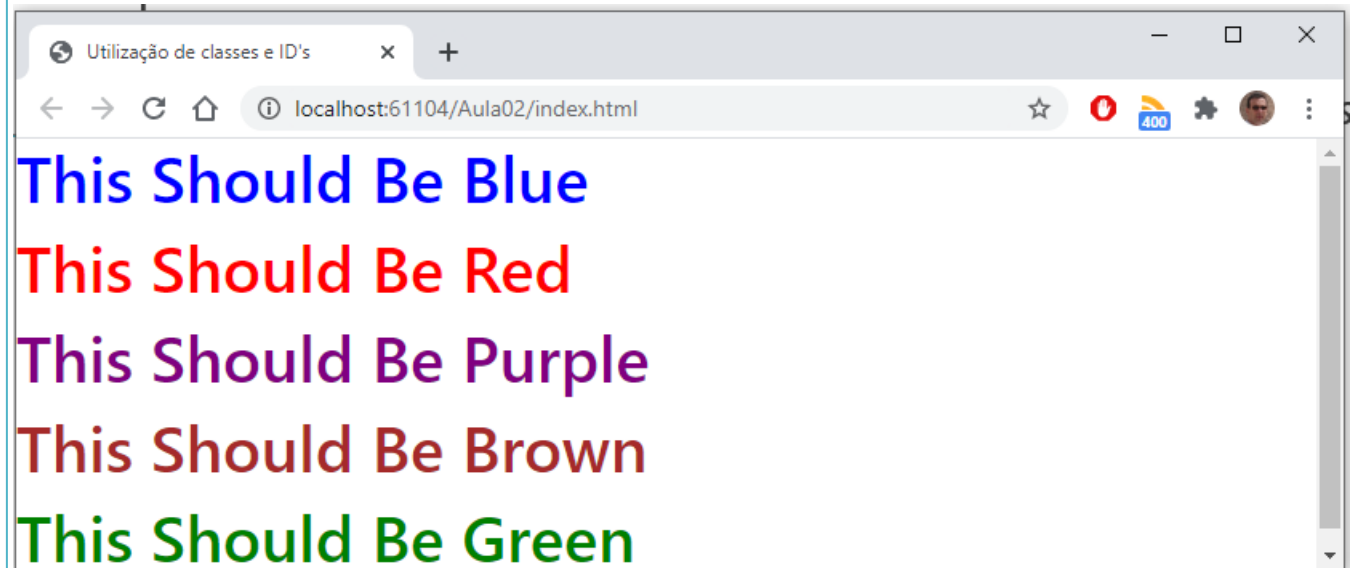
## Múltiplo seletor de classe

É possível marcar um elemento com várias classes.

*O exemplo abaixo possui duas classes, mas não está limitado a duas. Podem ser "N"*

```
<!DOCTYPE html>
<html>
<head>
  <title>Utilização de classes e ID's</title>
  <style>
    #one { color: blue; }
    #one.two { color: red; }
    .three { color: purple; }
    .four { color: brown; }
    .three.four { color: green; }
  </style>
</head>
<body>
  <h1 id="one">This Should Be Blue</h1>
  <h1 id="one" class="two">This Should Be Red</h1>
  <h1 class="three">This Should Be Purple</h1>
  <h1 class="four">This Should Be Brown</h1>
  <h1 class="three four">This Should Be Green</h1>
</body>
</html>
```

13/10/2020



# Elementos, classes e Id's

Qual a diferença?

Resumo importante:

Os estilos CSS dos marcadores html utilizam apenas o nome do marcador como identificados;

Os estilos CSS das classes começam com um ponto (".");

Os estilos CSS dos IDs começam com um cardinal ("#")

# Utilização de fontes

# CSS – Cascading Style Sheets

Fontes de texto - utilização de fontes próprias

Uma das principais características do desenvolvimento de websites institucionais é poder aplicar a imagem institucional na sua plenitude.

Isso implica, muitas vezes, na utilização de fontes/tipos de letra próprios, feitas especificamente para essa marca.

Essas fontes não estão disponíveis nos computadores dos clientes que lhes acedem...



# CSS – Cascading Style Sheets

Fontes de texto - utilização de fontes próprias

## Solução?

A solução inicial foi criar imagens e disponibilizar esses texto como imagens. Isso não é fácil e implica numa necessidade de largura de banda para a transmissão dessas imagens...

A solução ideal passa por mandar as fontes de texto para os computadores dos utilizadores remotos ...

Para além das fontes pré-existentes nos computadores é possível a instalação de novas fontes numa página web.

Neste caso, é necessário possuir os ficheiros de definição da fonte no seu computador.



# CSS – Cascading Style Sheets

Fontes de texto - utilização de fontes próprias

Deve ter em atenção que nem todas fontes estão adaptadas a todos os tipos de dispositivos.




# CSS – Cascading Style Sheets

## Fontes de texto - utilização de fontes próprias

[HOME](#) [FIND FONTS](#) [HOT](#) [RECENT](#) [ALMOST FREE](#) [WEBFONT GENERATOR](#) [FAQ](#)

# OPEN SANS

 [Ascender Fonts](#) | [Sans Serif](#) | [10 Styles](#)

[Specimens](#) [Test Drive](#) [Glyphs](#) [License](#) [Webfont Kit](#)

## Webfont Kit

This font's license appears to allow you to use @font-face css embedding!

**Choose a Subset:**  

Western Latin (Default) ▼

**Choose Font Formats:**  

☒ TTF ☒ EOT ☒ WOFF ☒ SVG

**Subsetting:**

Subsetting reduces the number of glyphs in the font to make a smaller file. If the font supports a particular language, it will appear in the menu.

**Formats:**

**TTF** - Works in most browsers except IE and iPhone.  
**EOT** - IE only.  
**WOFF** - Compressed, emerging standard.  
**SVG** - iPhone/iPad.

[DOWNLOAD @FONT-FACE KIT](#)

ads ▶ open-sans-fontfacekit.zip

Name	Type
web fonts	File folder
Apache License.txt	Text Document
How_to_use_webfonts.html	Firefox HTML Document

js ▶ open-sans-fontfacekit.zip ▶ web fonts

Name	Type
opensans_bold_macroman	File folder
opensans_bolditalic_macroman	File folder
opensans_extrabold_macroman	File folder
opensans_extrabolditalic_macroman	File folder
opensans_italic_macroman	File folder
opensans_light_macroman	File folder
opensans_lightitalic_macroman	File folder
opensans_regular_macroman	File folder
opensans_semibold_macroman	File folder
opensans_semibolditalic_macroman	File folder

# CSS – Cascading Style Sheets

Fontes de texto - utilização de fontes próprias

## A instrução CSS para instalar uma fonte

```
<style>
  @font-face {
    font-family: 'MyWebFont';
    src: url('WebFont.eot');
    src: url('WebFont.eot?iefix') format('eot'), url('WebFont.woff') format('woff'), url('WebFont.ttf')
format('truetype'), url('WebFont.svg#webfont') format('svg');
  }
</style>
```

## Instrução para instalação da variante da fonte OpenSans - OpenSansBold

```
<style>
  @font-face {
    font-family: 'open_sansbold';
    src: url('OpenSans-Bold-webfont.eot');
    src: url('OpenSans-Bold-webfont.eot?#iefix') format('embedded-opentype'), url('OpenSans-Bold-webfont.woff')
format('woff'), url('OpenSans-Bold-webfont.ttf') format('truetype'), url('OpenSans-Bold-webfont.svg#open_sansbold')
format('svg');
    font-weight: normal;
    font-style: normal;
  }
</style>
```

# Fontes públicas – Google Fonts

Para além do exemplo anterior, em que os ficheiros de definição estão no computador do utilizador ou no servidor web, há outra forma de utilizar fontes – carregando-as diretamente do distribuidor.

Neste caso vamos utilizar as fontes públicas da Google,

Ver: <https://fonts.google.com/> (1.006 famílias em 13/10/2020)

Cada família possui um ou mais estilos

Google Fonts

fonts.google.com

Browse fontsFeaturedArticlesAbout

Search

SentenceType something

40px

Categories

Language

Font properties

☐ Show only variable fonts

1006 of 1006 families

Sort by: Trending

Roboto

Christian Robertson

12 styles

Almost before we knew it, we had left the ground.

Commissioner

Kostas Bartsokas

Variable

Almost before we knew it, we had left the ground.

Piazzolla

Juan Pablo del Peral, Huerta Tipográfica

Variable

Almost before we knew it, we had left the ground.

Open Sans

Steve Matteson

10 styles

Almost before we knew it, we had left the ground.

Sansita Swashed

Omnibus-Type

Variable

*Almost before we knew it, we had left the ground.*

Noto Sans JP

Google

6 styles

Almost before we knew it, we had left the ground.

Lato

Łukasz Dziedzic

10 styles

Almost before we knew it, we had left the ground.

Montserrat

Julieta Ulanovsky, Sol Matas, Juan Pablo del Peral, Jacques Le Bailly

18 styles

Almost before we knew it, we had left the ground.

Start

Open Sans - Google Fonts

fonts.google.com/specimen/Open+Sans?sidebar.open=true&selection.family=Open+Sans

Google Fonts

Browse fonts

Featured

Articles

About

Open Sans

Designed by Steve Matteson

Select styles

Glyphs

About

License

Pairings

Styles

Type here to preview text

Almost before we knew it, we had left the ground.

Size: 30px

Light 300

Almost before we knew it, we had left the ground.

Light 300 italic

Almost before we knew it, we had left the ground.

Regular 400

Almost before we knew it, we had left the ground.

Regular 400 italic

Almost before we knew it, we had left the ground.

Semi-bold 600

Almost before we knew it, we had left the ground.

Semi-bold 600 italic

Almost before we knew it, we had left the ground.

Download family

Selected family

Review

Embed

To embed a font, copy the code into the <head> of your html

<link>

@import

<link href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap" rel="stylesheet">

CSS rules to specify families

font-family: 'Open Sans', sans-serif;

46

API docs

# Exemplo de utilização

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <meta name="Author" content="Joaquim Sousa Pinto">
  <meta name="Keywords" content="exemplos">
  <meta name="Description" content="Exemplos das aulas">
  <title>Utilização de fonte externa</title>
  <link href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap" rel="stylesheet">
  <style>
    body {
      font-family: 'Open Sans', sans-serif;
    }
  </style>
</head>
<body>

</body>
</html>
```

# Marcadores HTML

## Formulários



# Formulários HTML

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

Os formulários são utilizados para a recolha de informação por parte dos utilizadores dos sítios na internet.

São inseridos num bloco `<form>...</form>`

Atributos:

Name – nome do formulário;

Action – endereço da entidade processadora a informação;

Method – forma de envio dos dados para a entidade processadora dos dados.

Suporta os valores: **GET**, **POST**, PUT, DELETE

Exemplo:

```
<form name="PersonData" action="http://192.168.160.36/FormEcho.aspx" method="POST">
...
</form>

<form name="PersonData" action="http://192.168.160.36/FormEcho.aspx" method="GET">
...
</form>
```

Para as aulas de ITW, de modo a que se possam fazer testes com dados reais, a entidade processadora está localizada em <http://192.168.160.36/FormEcho.aspx>.  
**Nota importante:** este endereço só é válido e visível dentro da UA – nas salas de aula ou, de casa, através de uma ligação por VPN.

# Campos de um formulário

Marcador input - `<input type="???" />`

O marcador `input` é um dos principais responsáveis pela recolha de informação em formulários.

Sintaxe:

```
<input type="???" />
```

Atributos:

Type - dependendo do valor assumido por este campo o comportamento do marcador altera-se.

Os valores possíveis para ao atributo type são: `button`, `checkbox`, `color`, `date`, `datetime`, `datetime-local`, `email`, `file`, `hidden`, `image`, `month`, `number`, `password`, `radio`, `range`, `reset`, `search`, `submit`, `tel`, `text`, `time`, `url`, `week` (23!)

# Campos de um formulário

Texto - Linha simples – `<input type="text" />`

Permite a inserção de uma linha de texto

Atributos:

Type – define o tipo de input. Para uma linha de texto, "texto";

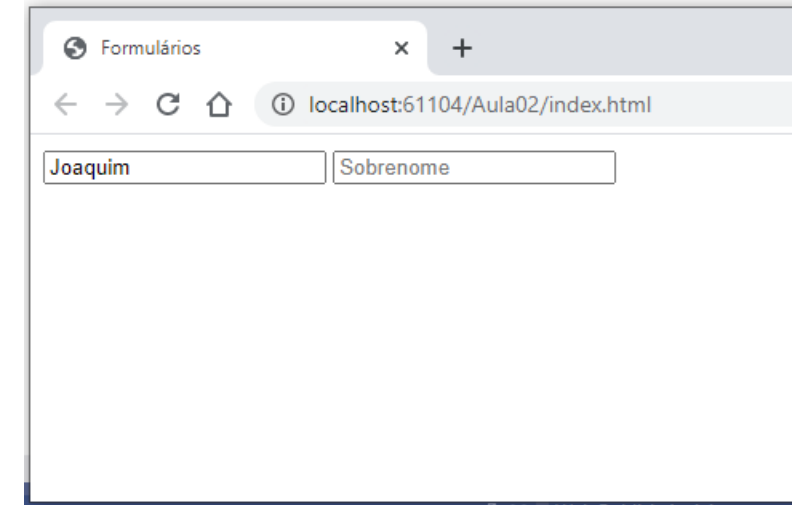
Name – define o nome do atributo. Obrigatório. De outra forma essa informação não é passada à entidade processadora da informação;

Value – (pode estar vazio ou omissa). Quando preenchido contém o valor a apresentar pela linha de texto;

Placeholder – texto que aparece na caixa quando esta está vazia. Normalmente serve de ajuda ao utilizador.

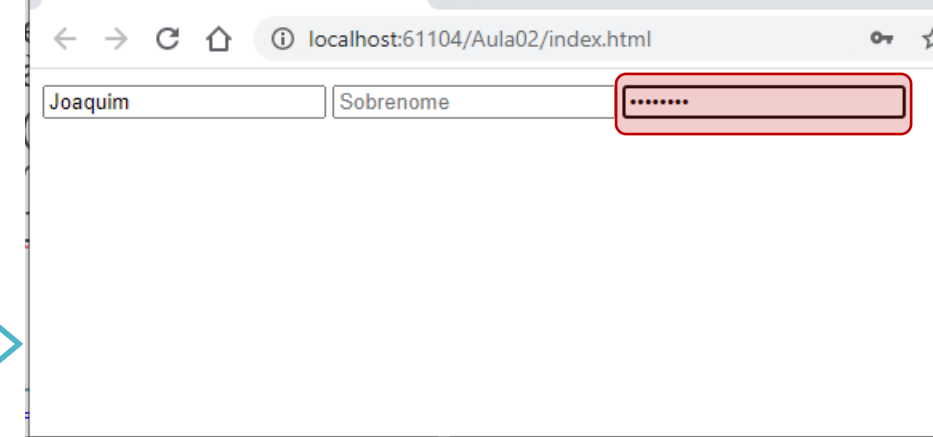
Exemplos:

```
<body>
  <input type="text" name="FirstName" value="Joaquim" placeholder="Nome" />
  <input type="text" name="LastName" placeholder="Sobrenome" />
</body>
```



# Campos de um formulário

Texto - Password – `<input type="password" />`



A screenshot of a web browser window at the URL 'localhost:61104/Aula02/index.html'. The browser shows a form with three input fields. The first field contains the text 'Joaquim'. The second field is labeled 'Sobrenome'. The third field is a password field, indicated by a red border and masked characters (dots). The browser's address bar and navigation buttons are visible at the top.

Permite a inserção de uma linha de texto sem que o seu conteúdo possa ser lido na interface

## Atributos:

Type – define o tipo de input. Para uma linha de texto, "password";

Name – define o nome do atributo. Obrigatório. De outra forma essa informação não é passada à entidade processadora da informação;

Value – (normalmente/recomendavelmente vazio). Não faz sentido introduzir um texto que se pretende secreto e depois ele estar escrito no documento html ...

Placeholder – texto que aparece na caixa quando esta está vazia. Normalmente serve de ajuda ao utilizador.

## Exemplo:

```
<input type="password" name="Password" placeholder="Digite a palavra passe" />
```

# Campos de um formulário

Texto – Hidden – `<input type="hidden" />`

Permite a inserção de texto sem que o seu conteúdo seja mostrado na interface

## Atributos:

Type – define o tipo de input. Para uma linha de texto, "hidden";

Name – define o nome do atributo. Obrigatório. De outra forma essa informação não é passada à entidade processadora da informação;

Value – contém o valor a enviar para a entidade processadora.

## Exemplo:

```
<input type="hidden" name="FormName" value="UserForm" />
```

# Campos de um formulário

Texto multilinha – `<textarea>...</textarea>`

Permite a inserção de um texto com várias linhas.

## Atributos:

Name – define o nome do atributo. Obrigatório. De outra forma essa informação não é passada à entidade processadora da informação;

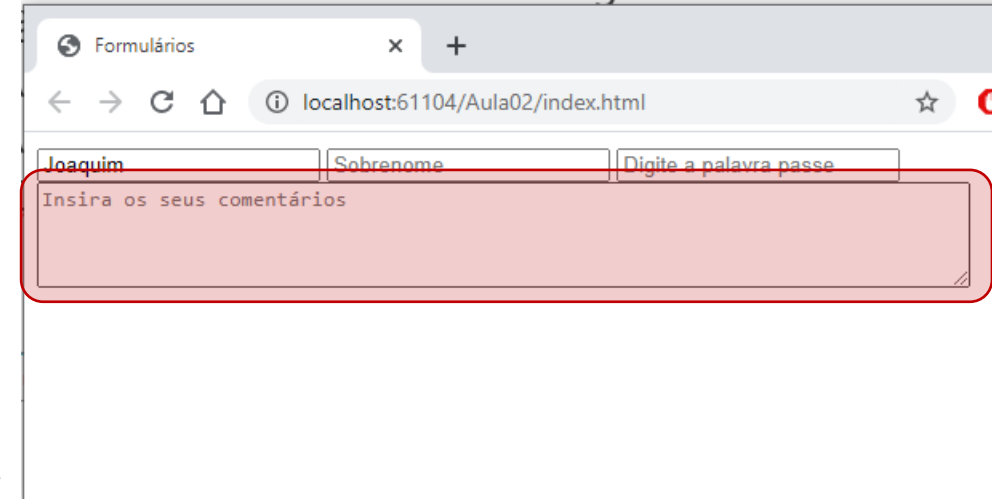
Rows – número de linhas

Cols – número de colunas

Placeholder – texto que aparece na caixa quando esta está vazia. Normalmente serve de ajuda ao utilizador

## Exemplo:

```
<textarea name="Comments" rows="4" cols="80" placeholder="Insira os seus comentários"></textarea>
```



The screenshot shows a web browser window with the title 'Formulários'. The address bar displays 'localhost:61104/Aula02/index.html'. The form contains three input fields: 'Joaquim' (with a red border), 'Sobrenome', and 'Digite a palavra passe'. Below these is a large text area with a red border and a placeholder text 'Insira os seus comentários'.

A screenshot of a web browser window. The address bar shows 'localhost:61104/Aula02/index.html'. The form contains four input fields: 'Joaquim', 'Sobrenome', 'Digite a palavra passe', and a larger text area labeled 'Insira os seus comentários'. Below the text area is a button labeled 'Enviar formulário' with a red border.

# Campos de um formulário

Botão de submit- `<input type="submit">`

O botão de submit é o que permite o envio dos dados do formulário para a entidade processadora

Atributos:

Type – define o tipo de input. Para as opções, o tipo é "submit";

Name – define o nome do atributo. Opcional.

Value – contém o texto do botão; também é enviado para a entidade processadora, caso o botão tenha um nome.

```
<input name="Button" type="Submit" value="Enviar formulário" />
```

Tal como referido, Para as aulas de ITW, de modo a que se possam fazer testes com dados reais, a entidade processadora está localizada em <http://192.168.160.36/FormEcho.aspx>.

**Nota importante:** este endereço só é válido e visível dentro da UA – nas salas de aula ou, de casa, através de uma ligação por VPN.

<https://jsfiddle.net/JoaquimSousaPinto/ke8zpz9e/>

# Campos de um formulário

Botão de reset - `<input type="reset">`

O botão de reset permite reverter o estado atual de um formulário ao seu estado inicial – tal como foi mostrado inicialmente - `<input type="reset">`

Atributos:

Type – define o tipo de input. Para as opções, o tipo é "reset";

Name – define o nome do atributo. Opcional.

Value – contém o texto a mostrar no botão.

```
<input name="resetBtn" type="reset" value="Limpar Formulário" />
```



# Campos de um formulário

Secções do formulário – `<fieldset>...</fieldset>`

O marcador `<fieldset>...</fieldset>` permite criar secções dentro de um formulário.

Marcadores filhos:

Cabeçalho da secção: `<legend>...</legend>`;

Todos os outros campos de um formulário.

Atributos:

Name – define o nome do `fieldset`.

# Campos de um formulário

## Checkboxes - `<input type="checkbox">`

Este marcador permite a escolha de ZERO OU MAIS opções de uma lista

### Atributos:

Type – define o tipo de input. Para as opções, o tipo é "checkbox";

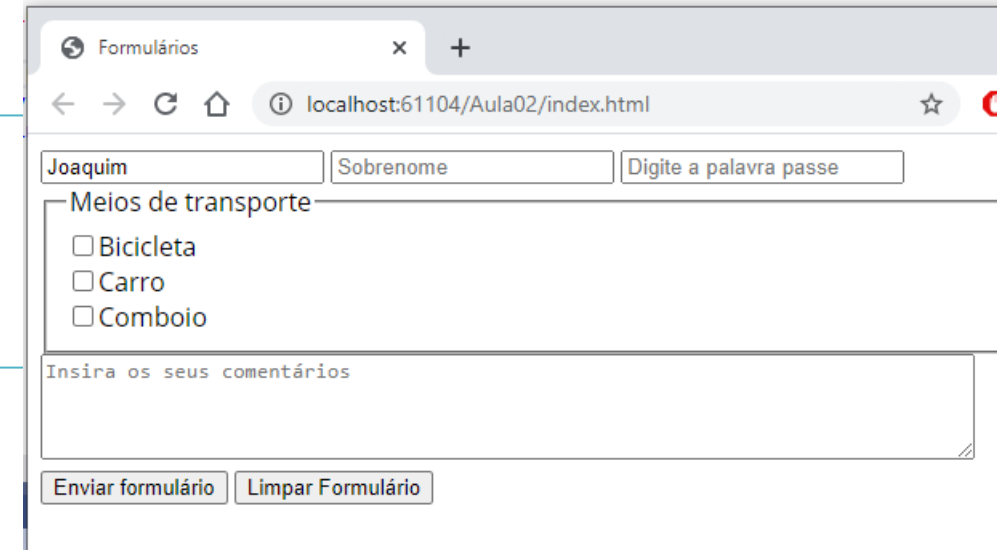
Name – define o nome do atributo. Obrigatório. De outra forma essa informação não é passada à entidade processadora da informação;

Value – contém o valor a enviar para a entidade processadora.

### Exemplos:

```
<fieldset>
  <legend>Meios de transporte</legend>
  <input type="checkbox" name="vehicle" value="Bicicleta">Bicicleta<br />
  <input type="checkbox" name="vehicle" value="Carro">Carro<br />
  <input type="checkbox" name="vehicle" value="Comboio">Comboio
</fieldset>
```

<https://jsfiddle.net/JoaquimSousaPinto/m5yhd5hq/>



# Campos de um formulário

## Radio boxes - `<input type="radio">`

Este marcador permite a escolha de ZERO OU UMA OPÇÃO de uma lista.

### Atributos:

**Type** – define o tipo de input. Para as opções, o tipo é "checkbox";

**Name** – define o nome do atributo. Obrigatório. De outra forma essa informação não é passada à entidade processadora da informação;

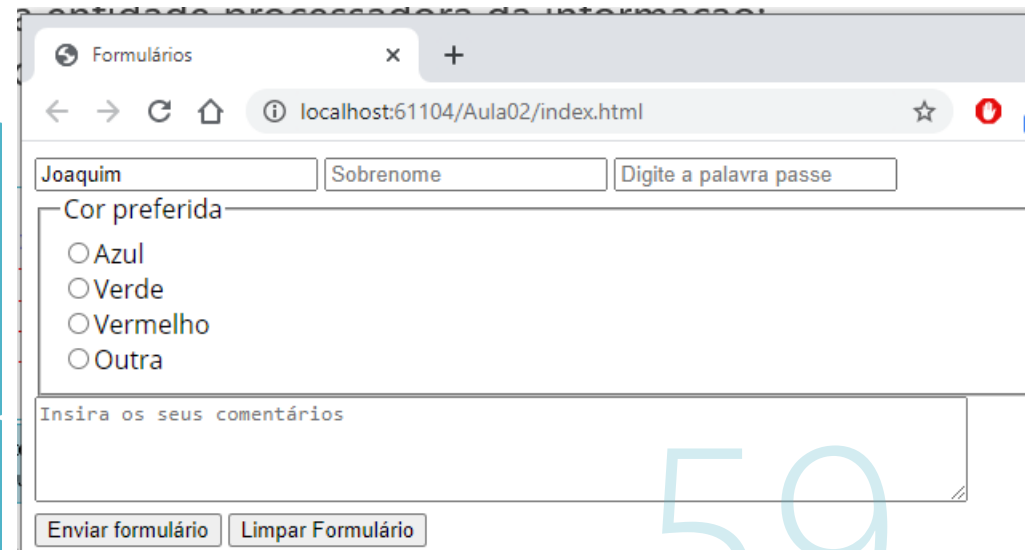
**Value** – contém o valor a enviar para a entidade processadora.

### Exemplos:

```
<fieldset>
  <legend>Cor preferida</legend>
  <input type="radio" name="color" value="Azul">Azul<br />
  <input type="radio" name="color" value="Verde">Verde<br />
  <input type="radio" name="color" value="Vermelho">Vermelho<br />
  <input type="radio" name="color" value="Outra">Outra
</fieldset>
```

**Nota importante:** se mais que um input do tipo radio possuir o mesmo nome, estes input's comportam-se como um grupo.

Num grupo NENHUMA ou APENAS UMA das opções pode ser seleccionada



# Campos de um formulário

## Botão genérico - `<input type="button">`

O botão genérico não possui um comportamento associado por omissão. Depende do que for configurado pelo utilizador

### Atributos:

Type – define o tipo de input. Para as opções, o tipo é "reset";

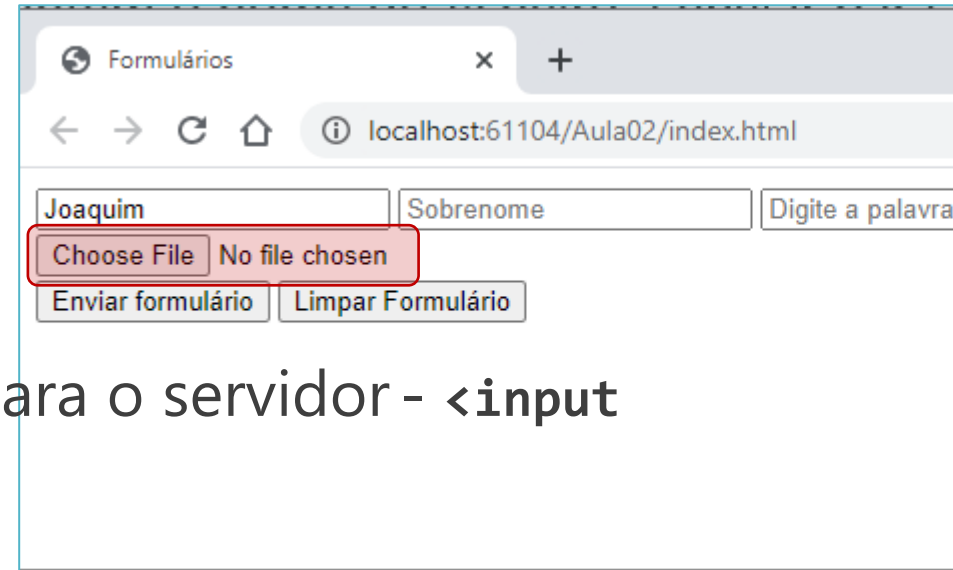
Value – contém o texto a mostrar no botão.

```
<input type="button" value="Click Me" />
```

Voltaremos a este botão quando for lecionada programação de comportamento de botões

# Campos de um formulário

Botão genérico - `<input type="file">`



O botão do tipo file permite o envio de ficheiros para o servidor - `<input type="file">`

Atributos:

**Type** – define o tipo de input. Para as opções, o tipo é "reset";

**Name** – contém o nome do ficheiro a enviar para a entidade processadora.

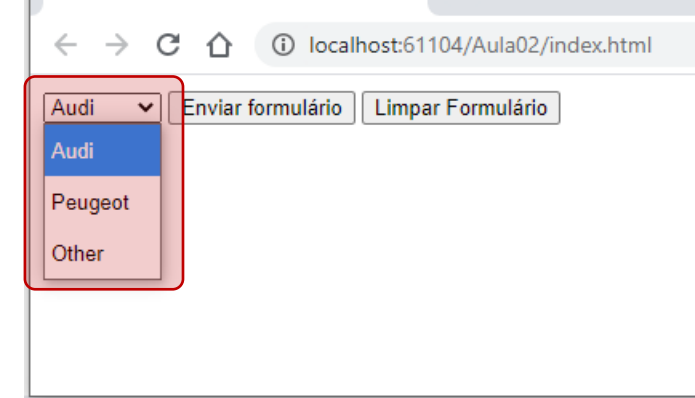
```
<input type="file" name="photo" />
```

**Nota importante:** para que os ficheiros possam ser recolhidos no servidor é imprescindível incluir no marcador do `<form>` o atributo `enctype` com o valor `multipart/form-data`.

```
<form name="PersonData" action="http://192.168.160.36/FormEcho.aspx" method="POST" enctype="multipart/form-data">
...
  <input type="file" name="photo" />
...
</form>
```

# Campos de um formulário

Listas de valores – seleção simples - `<select>...</select>`



As listas de valores são importantes quando se pretende que o utilizador selecione valores dentro de uma gama pré-definida. Para definir a lista é utilizado o marcador `<select>...</select>`.

As opções da lista são delimitadas por marcadores `<option>...</option>`

Atributos:

**Value** – o valor a enviar para a entidade processadora

**Selected** – (Opcional). Indica que esta opção é a pré-selecionada. Toma sempre o valor "selected".

```
<select name="Car">
  <option value="1" selected="selected">Audi</option>
  <option value="2">Peugeot</option>
  <option value="3">Other</option>
</select>
```

Caso o utilizador escolha a marca Audi, o valor enviado à entidade processadora será "1"!

# Campos de um formulário

Listas de valores – seleção múltipla - `<select multiple="multiple">...</select>`

Quando se pretende utilizar uma lista em que o utilizador pode escolher mais do que um elemento, isso deve ser assinalado no marcador `<select>...</select>` com o atributo `multiple`.

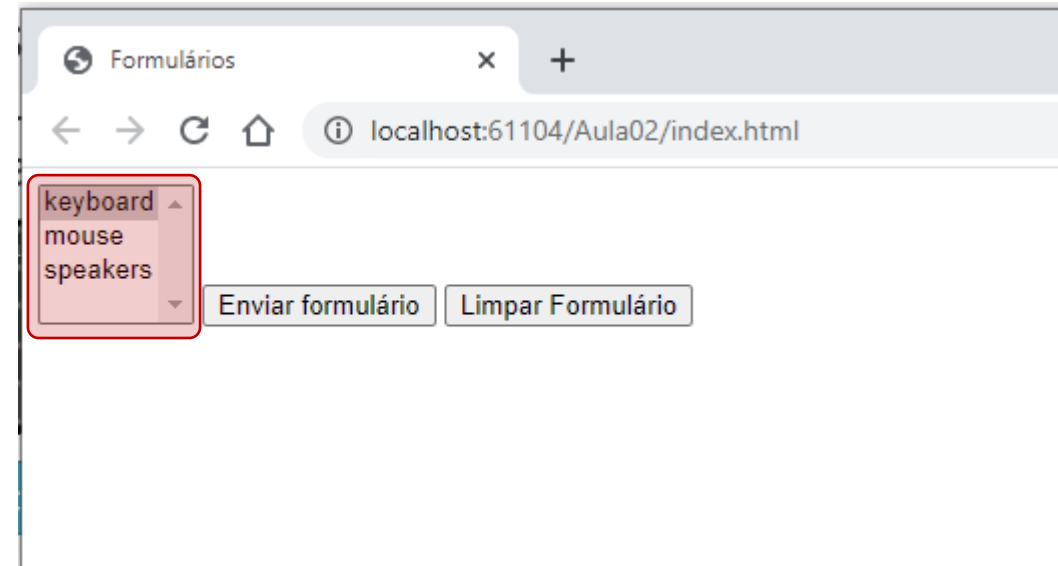
Atributos:

*Multiple – atributo que indica que é possível escolher mais que um elemento na lista. Caso esteja presente, toma sempre o valor "multiple".*

```
<select name="ComputerAccessories" multiple="multiple">
  <option value="Value 1" selected="selected">keyboard</option>
  <option value="Value 2">mouse</option>
  <option value="Value 3">speakers</option>
</select>
```

Pergunta: como se escolhe mais que um elemento na lista (keyboard e mouse, por exemplo ...)?

<https://jsfiddle.net/JoaquimSousaPinto/xhLfg5t8/2/>



The screenshot shows a web browser window with the title 'Formulários' and the address bar displaying 'localhost:61104/Aula02/index.html'. The main content area contains a form with a multiple-select dropdown menu. The dropdown menu has three options: 'keyboard', 'mouse', and 'speakers'. The 'keyboard' option is currently selected. Below the dropdown menu are two buttons: 'Enviar formulário' and 'Limpar Formulário'.

# Usabilidade e acessibilidade

## Labels

Os `labels` são utilizados para associar um texto explicativo a um marcador de um formulário. A associação entre ambos faz-se através do atributo ID

Isso significa que para além de um `Name`, os marcadores passam também a necessitar de um atributo `ID` que pode, ou não, ser igual ao `Name`.

```
<label for="fn">First Name</label>  
<input type="text" id="fn" name="FirstName" />
```

Sempre que se selecciona um `label` (*click com o rato*), se o campo associado for um `input` do tipo `"text"` ou uma `<textarea>` o campo respetivo fica selecionado; se for um `input` do tipo `"radio"` a opção fica imediatamente selecionada; se for um `input` do tipo `"checkbox"` a opção troca de estado (selecionado/desselecionado)



# Usabilidade e acessibilidade

## Labels

Os labels são muito importantes do ponto de vista da usabilidade e da acessibilidade de uma página web.

São obrigatórios para que um formulário seja considerado “acessível” nos testes respetivos.

Para mais informação sobre acessibilidade na web , ver em:

<https://www.w3.org/WAI/test-evaluate/>, consultado em 13out2020

Para testar páginas pessoais sobre a sua conformidade de acordo com as normas de acessibilidade, testar em:

<http://achecker.ca/checker/index.php>, consultado em 07out2020

# Usabilidade e acessibilidade

## TabIndex

O atributo tabindex controla a ordem por que os campos e hiperligações são apresentadas sempre que carregamos na tecla [Tab].

São um elemento de auxílio e de usabilidade pois a utilização do rato como dispositivo apontador nem sempre é uma opção para todos os utilizadores.

```
<input type="text" id="fn" name="FirstName" tabindex="10" />
```