



CSS

(Cascading Style Sheets)



Permitem manipular a aparência (estilo) dos elementos HTML

Vantagens da utilização de CSS:

- Grande liberdade de formatação
- Maior produtividade
- Maior facilidade de actualização

Os estilos dados pelo autor do site, terão prioridade sobre os estilos do browser

Os estilos 'inline' têm prioridade sobre os estilos dados numa folha de estilos externa





Folhas de estilo vs formatação em HTML

- Inline

```
<h1 style="color: blue">O meu portfolio</h1>
```

- Folha de estilos

```
h1 {  
  color: blue;  
}
```





Folhas de estilo vs formatação em HTML

A formatação com CSS evita que:

- Tenhamos de repetir a informação de formatação em cada uma das linhas: é apenas escrita uma vez!
- Tenhamos de alterar o documento todo se pretendermos alterar a formatação: basta alterar a linha em que a formatação é definida!





Comentários em CSS

Em CSS, os comentários inserem-se entre `/*` e `*/`

- **Comentário de uma linha**

```
a {  
  /* color: yellow; */  
  text-decoration: none;  
}
```

- **Comentário de múltiplas linhas**

```
/* a {  
  color: yellow;  
  text-decoration: none;  
} */
```





Heranças de formatação CSS

- Os documentos HTML têm uma estrutura hierárquica, em que uns elementos são filhos de outros
- Os elementos filhos herdam dos pais as características e propriedades, como por exemplo a cor ou o tamanho

Exemplo:

```
body {  
  font-family: helvetica, arial, sans-serif;  
}
```

(Faz inspect e observa que o teu site herdou a mesma font)





Heranças de formatação CSS

Exercício:

[Heranças de Formatação CSS](#)





CSS

Tipos de folhas de estilo





Tipos de folhas de estilos

Existem 3 tipos de folhas de estilos:

- **Externas** - a formatação é definida num ficheiro à parte
- **Internas** - a formatação é definida no próprio documento HTML
- **Em linha (inline)** - a formatação é definida na tag dos elementos HTML





Folhas de estilos: External

- É definida num ficheiro à parte, pode ser responsável pela formatação de todas as páginas de um website
- Tem de ser referenciada dentro da tag <link>, na secção <head> do documento HTML

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta content="width=device-width, initial-scale=1.0" name="viewport" />

    <title>Portfolio</title>
    <meta name="title" content="Portfolio" />
    <link rel="stylesheet" href="style.css" />
  </head>

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





Folhas de estilos: **Internal**

- A formatação é definida na secção head do próprio ficheiro, delimitada entre `<style>` e `</style>`
- As definições CSS valem apenas para a página em questão





Folhas de estilos: **Inline**

- As definições CSS são incluídas no próprio elemento que se pretende formatar com `style="propriedade: valor;"`
- A definição só formata o elemento HTML em questão, não permite a formatação de vários elementos ao mesmo tempo





“Cascading order” das folhas de estilo

Prioridades na leitura da formatação CSS:

- CSS em linha (inline styles)
- CSS externo e interno (a ordem é importante!)
- Valores default do browser

Nota: Se as mesmas propriedades forem definidas com valores diferentes para o mesmo elemento, **é o valor da última a ser lida que prevalece!**





CSS

Selectores básicos





Seletores de CSS

Existem vários tipos de seletores CSS, os que iremos abordar serão:

- Universal
- Tags
- Ids
- Classes





Seletor Universal

É feita uma seleção de todos os elementos do documento HTML

```
* {  
  box-sizing: border-box;  
}
```





Seletores tag

É feita uma seleção com o nome da tag do elemento HTML que queremos alterar.

```
a {  
  text-decoration: none;  
}  
  
p {  
  font-size: 16px;  
}  
  
h1 {  
  color: tomato;  
}
```





Seletores de classes (1\3)

É feita uma seleção de acordo com a classe do elemento:

```
✓ .heading {  
  position: relative;  
  font-size: 48px;  
  font-weight: 700;  
  text-align: center;  
}
```

Nota: Com estes seletores temos uma maior precisão na selecção de elementos. Além disso, esta forma permite-nos reutilizar os mesmos estilos para vários elementos diferentes.





Seletores de classes (2\3)

Por vezes podemos querer restringir o alcance do selector de classe, como por exemplo restringir o alcance do selector apenas a um elemento:

```
h2.heading {  
  position: relative;  
  font-size: 48px;  
  font-weight: 700;  
  text-align: center;  
}
```





Seletores de classes (3\3)

Existem casos em que os elementos HTML podem ter mais do que uma classe:

```
<h1 class="bold text-center">O meu portfolio</h1>
```

```
.bold {  
  font-weight: bold;  
}  
  
.text-center {  
  text-align: center;  
}
```





Seletores de id

Seleção de acordo com o id do elemento:

```
<p id="alert">Não autorizado</p>
```

```
#alert {  
  color: ■ red;  
}
```

Nota: Estes selectores só seleccionam um elemento na página web, visto que os id são únicos





CSS

Tipografia





Tipografia

Propriedades mais usadas na formatação de texto:

- **color** - cor do texto
- **text-align** - alinhamento do texto: `left` | `right` | `center` | `justify`
- **text-decoration** - decoração do texto: `none` | `underline` | `line-through`
- **text-transformation** - capitalização: `none` | `capitalize` | `uppercase` | `lowercase`
- **text-indent** - indentação da primeira linha do texto
- **line-height** - especifica a altura de uma linha de texto





CSS

Cores






Cores

Podem ser definidas usando:

- **O seu nome válido** - "red", "blue", "green", "black", "yellow"...
- **Um valor RGB** - rgb(red, green, blue)
- **Um valor Hex** - #ff0000
- **Cores RGBA** - rgb(red, green, blue, alpha)
- **Cores HSL** - hsl(hue, saturation, lightness)
- **Cores HSLA** - hsl(hue, saturation, lightness, alpha)





CSS

Posicionamento



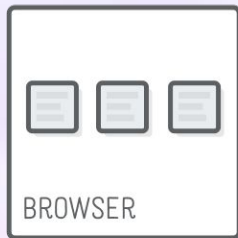


Posicionamento

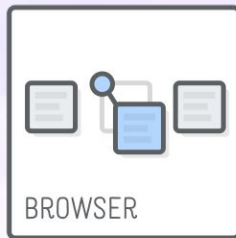
Especifica o tipo de posicionamento de um elemento: **static**, **relative**, **fixed**, **absolute**

O posicionamento é depois especificado usando as propriedades: **top**, **bottom**, **left** e **right**

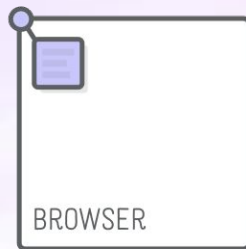
STATIC



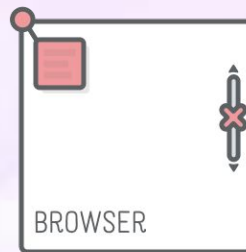
RELATIVE



ABSOLUTE



FIXED

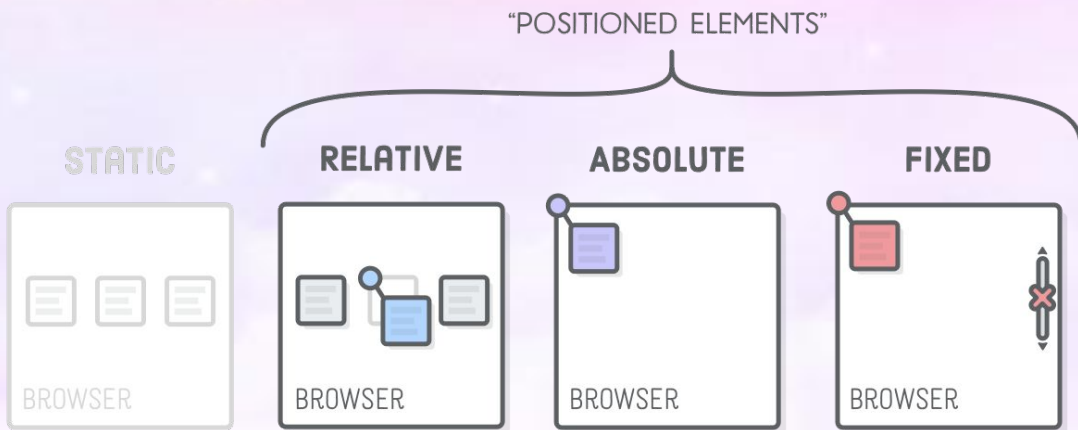




Posicionamento - static

É o default. Os elementos são posicionados de acordo com o fluxo normal da página

Se o “position” de um elemento não tiver um value de “static”, é chamado de elemento posicionado.

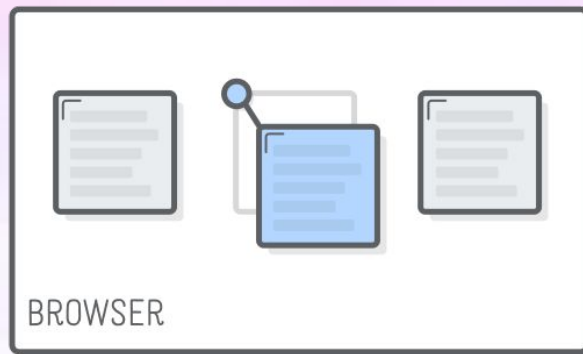




Posicionamento - relative

O posicionamento “relative” move o elemento relativamente onde este apareceria normalmente no flow estático da página.

```
.item-relative {  
  position: relative;  
  top: 30px;  
  left: 30px;  
}
```



RELATIVE POSITIONING





Posicionamento - absolute

O posicionamento “absolute” é como o relative, mas o offset é relativamente a ao seu elemento ancestral.

```
.item-absolute {  
  position: absolute;  
  top: 10px;  
  left: 10px;  
}
```



ABSOLUTE POSITIONING

Nota: Se nenhum elemento pai for definido, o elemento é posicionado em relação a toda a janela do browser

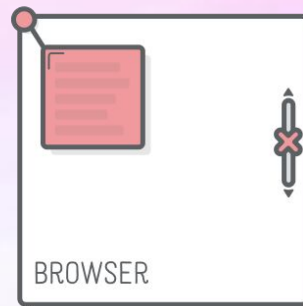




Posicionamento - fixed

O posicionamento “fixed” é relativo ao viewport, mantém-se sempre no mesmo sítio mesmo que o user faça scroll

```
.item-fixed {  
  position: fixed;  
  bottom: 0;  
  right: 0;  
}
```



FIXED POSITIONING





Posicionamento - exemplos

Exercicio: [CSS Position](#)





CSS

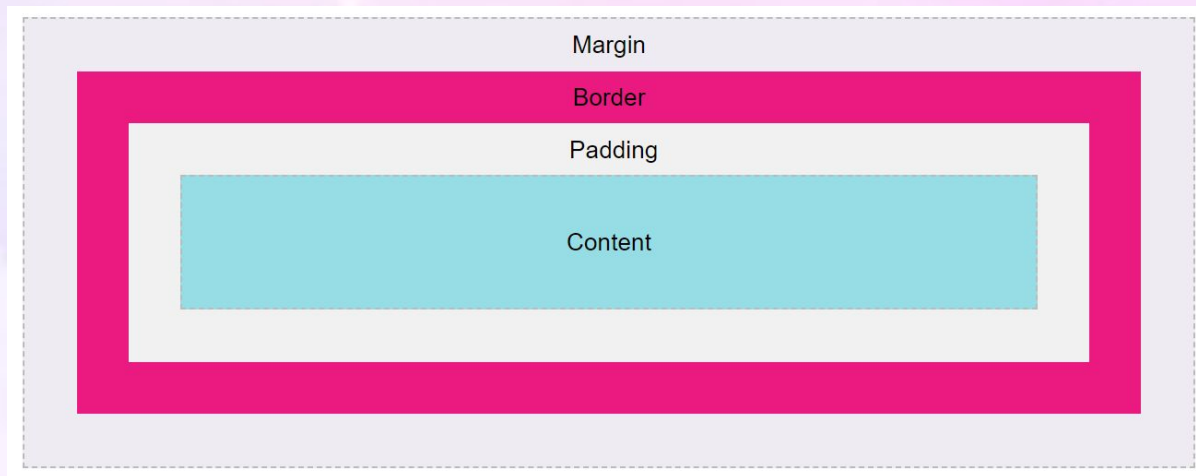
Box Model





Box Model

Todos os elementos HTML estão “dentro de uma caixa”, com margins, borders, paddings e o conteúdo do elemento.





Box Model

- **Content** - o conteúdo da box, onde o texto e as imagens aparecem
- **Padding** - é a área em torno do conteúdo, este é transparente
- **Border** - a border está em torno do padding e do conteúdo
- **Margin** - é a área fora da border, esta é transparente





Box Model - como medir?

Importante: Ao definir as propriedades de largura e altura de um elemento com CSS, só estamos a definir a largura e a altura da área de **conteúdo**.

Para calcular o tamanho total de um elemento, também devemos adicionar o padding, border e margins.

- A largura total de um elemento deve ser calculada assim:

Largura total do elemento = largura + padding esquerdo + padding direito + border esquerda + border direita + margin esquerda + margin direita

- A altura total de um elemento deve ser calculada assim:

Altura total do elemento = altura + padding superior + padding inferior + border superior + border inferior + margin superior + margin inferior



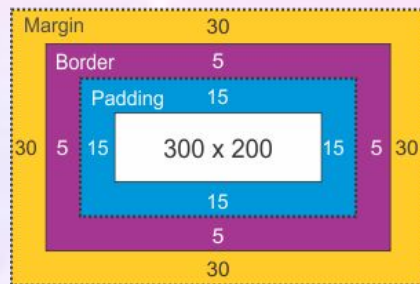


Box Model - como usualmente usamos?

Por norma, a **box-sizing** é sempre definida como **border-box** no início de cada ficheiro css.

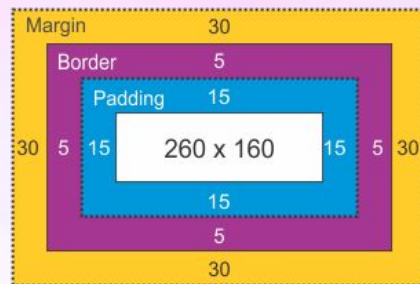
Desta forma o **padding** e a **border** fazem parte da largura/altura total do elemento em si, excepto a **margin** que é exterior.

Box Model is content-box



```
div{  
  width: 300px;  
  height: 200px;  
  padding: 15px;  
  border: 5px solid grey;  
  margin: 30px;  
  -moz-box-sizing: content-box;  
  -webkit-box-sizing: content-box;  
  box-sizing: content-box;  
}
```

Box Model is border-box



```
div{  
  width: 300px;  
  height: 200px;  
  padding: 15px;  
  border: 5px solid grey;  
  margin: 30px;  
  -moz-box-sizing: border-box;  
  -webkit-box-sizing: border-box;  
  box-sizing: border-box;  
}
```





CSS

Fundos (Background)





Fundos (background)

As propriedades de background são usadas para adicionar um efeito de fundo nos elementos:

- **Background-color** - define a cor de fundo, pode ter opacidade, ser um gradiente de cor, etc
- **Background-image** - imagem usada como fundo
- **Background-repeat** - define se a imagem é repetida, horizontal e verticalmente
- **Background-attachment** - define se a imagem deve fazer scroll ou ser fixa
- **Background-position** - define o ponto de partida da imagem
- **background** (shorthand property) - usado para combinar várias propriedades numa só declaração





Fundos - background-image

A propriedade background-image especifica a imagem que irá ser usada com background de um elemento.

Por default, a imagem é repetida caso seja pequena, de modo a preencher a totalidade do elemento.

```
.footer {  
  background-image: url("../img/footer-new.png");  
  background-size: cover;  
  background-repeat: no-repeat;  
  background-position: top left;  
  height: 400px;  
}
```

Exercício: Como é ficaria se usasse apenas a propriedade "background"?





CSS

Pseudo-classes





Pseudo-classes

Uma pseudo-class é usada para definir um estado especial de um elemento

Exemplos:

- Estilizar um elemento quando um user passa o rato por cima
- Estilizar links visitados/não visitados de cor diferente
- Estilizar um elemento quando está focado

Exercicio: [Pseudo-classes](#)





CSS

Pseudo-elements





Pseudo-elements

Um pseudo-element é usado para estilizar partes específicas de um elemento

Exemplos:

- Estilizar a primeira letra ou linha de um elemento
- Inserir conteúdo antes ou depois de um elemento

Exercicio: [Pseudo-elements](#)





CSS

Revisão





CSS: Revisão

Consegues dizer me 3 tipos diferentes de selectores CSS?





CSS: Revisão

Porque é que não devemos usar Ids quando estamos a dar estilos aos elementos?





CSS: Revisão

Como é que escolhemos nomes para as nossas classes de CSS?





CSS: Revisão

Qual é a propriedade que adiciona espaço fora de um elemento?





CSS

Animações





Animações de CSS

O CSS permite a animação de elementos HTML sem javascript!

Propriedades:

- @keyframes
- animation-duration
- animation-name
- animation-iteration-count
- ...





Animações de CSS: @keyframe

Para usarmos uma animação primeiro temos de especificar as keyframes para a animação

```
@keyframes colormix {  
  0% {background-color: pink; left:0px; top:0px;}  
  25% {background-color: turquoise; left:200px; top:0px;}  
  50% {background-color: tomato; left:200px; top:200px;}  
  75% {background-color: lawngreen; left:0px; top:200px;}  
  100% {background-color: rebeccapurple; left:0px; top:0px;}  
}
```

```
div {  
  position: relative;  
  width: 100px;  
  height: 100px;  
  border-radius: 50%;  
  background-color: red;  
  animation-name: colormix;  
  animation-duration: 4s;  
  animation-iteration-count: infinite;  
}
```

Depois, para a nossa animação funcionar, temos de a ligar a um elemento!

Exercício: [Animação de CSS](#)





CSS Display





Display

A propriedade display especifica o comportamento visual que um elemento vai ter.

Propriedades:

- **inline** - mostra um elemento como um elemento inline (como o span). As propriedades de altura e largura não tem efeito
- **block** - mostra um elemento como um elemento block (como o p ou h1). Inicia uma nova linha e toma conta do comprimento total
- **flex** - mostra um elemento como um elemento bloco de um container flex
- **grid** - mostra um elemento como um elemento bloco de um container grid
- **none** - o elemento é totalmente removido
- ...

Exercicio: [Display](#)





CSS

Display Flex





Display: Flex

Um flex container expande os items para preencher o espaço livre e permite alguma liberdade no nosso layout.

Propriedades mais usadas:

- **flex-direction** - row | row-reverse | column | column-reverse
- **flex-wrap** - no-wrap | wrap | wrap-reverse
- **align-items** - flex-start | flex-end | center | baseline | stretch
- **justify-content** - flex-start | flex-end | center | space-between | space-around
- ...

Exercicio: [Flex Kawaii](#) 🐼





CSS

Unidades





Unidades

CSS tem várias unidades para expressar um comprimento. Existem dois tipos de unidades de comprimento:

- **Absoluto** - são fixas e um comprimento expresso em qualquer uma delas aparecerá exatamente com esse tamanho:
 - **px** - relativos ao tamanho do dispositivo de visualização
- **Relativo** - especificam um comprimento relativo a outra propriedade de comprimento. Estas escalam melhor entre diferentes meios de renderização:
 - **em** - relativo ao tamanho da fonte do elemento (2em significa 2 vezes o tamanho da fonte atual)
 - **rem** - relativo ao tamanho font do elemento raiz





Unidades - css pixel (px)

“O pixel CSS - denotado em CSS com o sufixo px - é uma unidade de comprimento que corresponde aproximadamente à largura ou altura de um único ponto que pode ser visualizado confortavelmente pelo olho humano sem esforço, mas sendo o menor possível” - [MDN Web Docs](#)

Nota: Um hardware pixel é um ponto luminoso individual no ecrã. Um software pixel, também chamado de CSS pixel, é uma unidade de medida.





CSS Breakpoints





Breakpoints



0-480

Smaller
smartphones



481-768

Tablets & larger
smartphones



769-1279

Laptops, larger tablets
in landscape, and small
desktops



1280+

Larger desktops
and monitors





CSS

Media Queries





Media Query

Uma media query usa a @media para incluir um block de propriedades de css, apenas em caso de uma certa condição ser verdadeira

Por exemplo, se quisermos mudar a cor de fundo do nosso website quando o nosso ecrã tem 600px ou for menor

```
@media only screen and (max-width: 600px) {  
  body {  
    background-color: lightblue;  
  }  
}
```





CSS

Bibliotecas





Bibliotecas: Bootstrap

O [Bootstrap](#) é uma biblioteca de CSS, gratuita e open-source, baseada no desenvolvimento “mobile-first”

Contem design templates baseadas em css e javascript, para tipografia, formulários, botões, navegação e outros componentes de interface.

