



Web
CSS

Catarina Oliveira

DCT DEPARTAMENTO DE CIÊNCIA
E TECNOLOGIA

CONTEÚDO

1. Contexto
2. Sintaxe
3. Integração de CSS em HTML
4. Seletores
 1. Por tipo de elemento
 2. Por identificador (#)
 3. Por classe (.)
 4. Por classe (.)
 5. Por atributos
 6. Baseados em combinadores
 7. Por pseudoclasses
 8. Por pseudoelementos
5. Cascata
6. Herança
7. Grid CSS
8. Boas Práticas

Contexto

CSS: *Cascading Style Sheets*

- Linguagem de estilização
- Criada e mantida pelo consórcio W3C
- Atualmente na versão 4 (vamos ver a versão 3 <https://www.w3schools.com/css/>)
- Usada como um mecanismo para adicionar estilos ao conteúdo de uma página web
- Agrega estilos num ficheiro à parte
 - Potencia legibilidade, manutenção e modularização do código da página web
- Vantagens
 - Economia de tempo: escreve-se o código uma vez e pode reutilizar-se noutras páginas
 - Carregamento rápido: não precisa de atributos HTML para cada *tag*
 - Facilidade de manutenção: pode mudar-se o estilo da página fazendo apenas alterações ao ficheiro CSS
 - Independência de plataforma: CSS é *standard* e suportada por todos os *browsers*

Sintaxe

Os estilos são definidos como um conjunto de regras



Componentes

- **Seletor**: seleciona os elementos HTML a estilizar
- **Bloco de declarações**: uma ou mais declarações de estilos cercadas por chavetas. Cada declaração tem um par `propriedade: valor` e termina sempre com `;`. Os comentários são feitos com `/*` e `*/`

Integração de CSS em HTML

- **Integração local**

```
<h1 style="color:blue;margin-left:30px;">Cabeçalho</h1>
```

- **Integração Interna**

```
<html>
  <head>
    <style>
      body {
        background-color: red;
      }
    ...
  </style>
</head>
<body>
  <h1>Cabeçalho</h1>
</body>
</html>
```

- **Integração externa**

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

Seletor por tipo de elemento

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo1.css">
7   </head>
8
9   <body>
10    <h1>Cabeçalho 1</h1>
11    <h2>Cabeçalho 2</h2>
12    <h3>Cabeçalho 3</h3>
13    <h4>Cabeçalho 4</h4>
14    <h5>Cabeçalho 5</h5>
15    <h6>Cabeçalho 6</h6>
16    <p>Parágrafo</p>
17  </body>
18 </html>

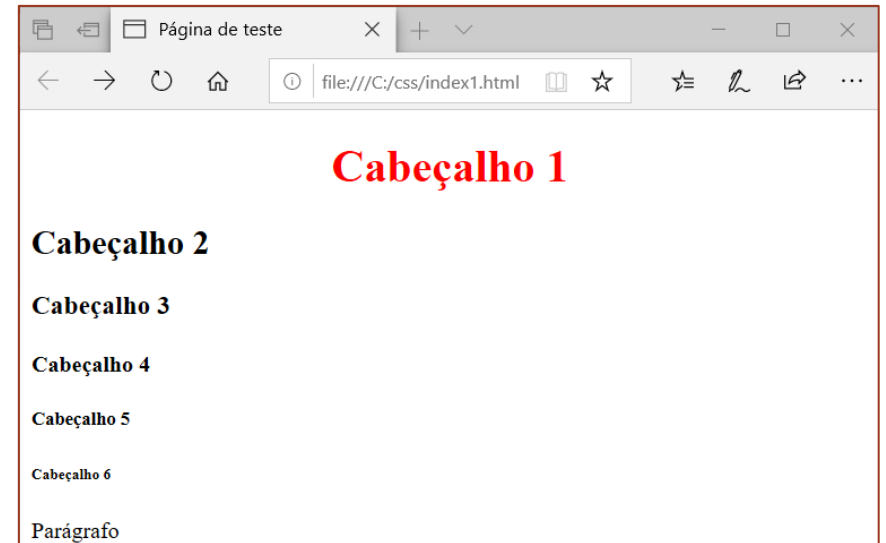
```

Ln : 20 Col : 1 Sel : 0 | 0 Windows (CR LF) UTF-8 INS

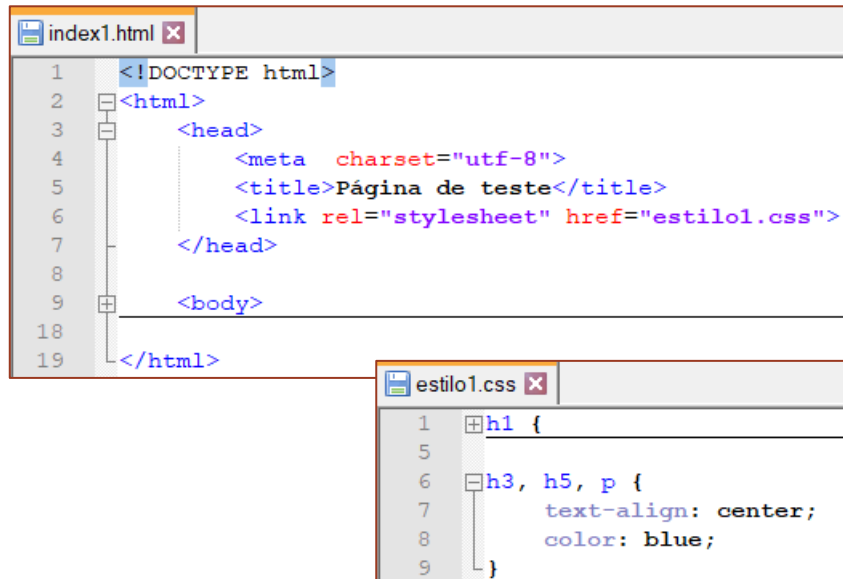
```

1 h1 {
2   text-align: center;
3   color: red;
4 }

```



Seletor por tipos de elementos (vários)



The screenshot shows two files in a code editor. The first file, `index1.html`, contains the following HTML code:

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo1.css">
7   </head>
8
9   <body>
10
18
19 </html>

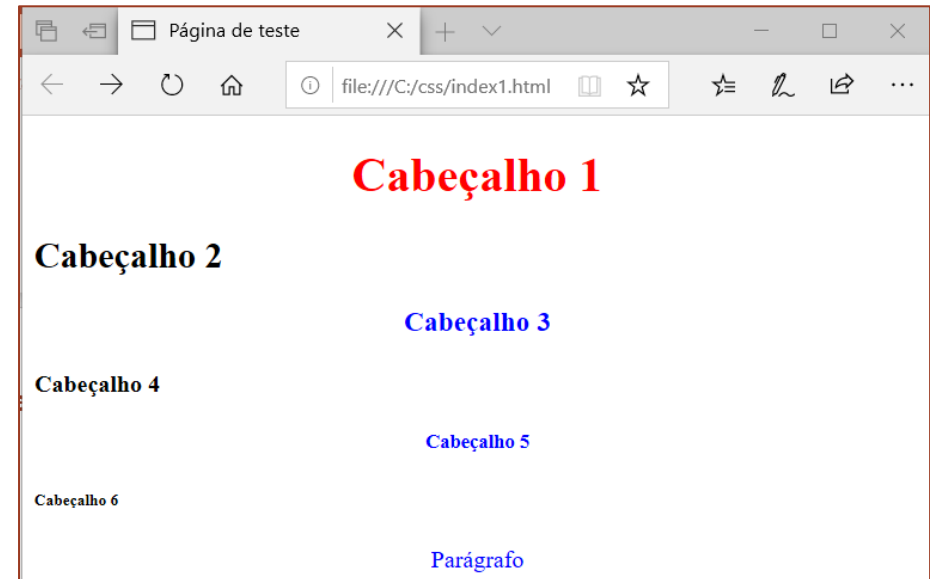
```

The second file, `estilo1.css`, contains the following CSS code:

```

1 h1 {
5
6 h3, h5, p {
7   text-align: center;
8   color: blue;
9 }

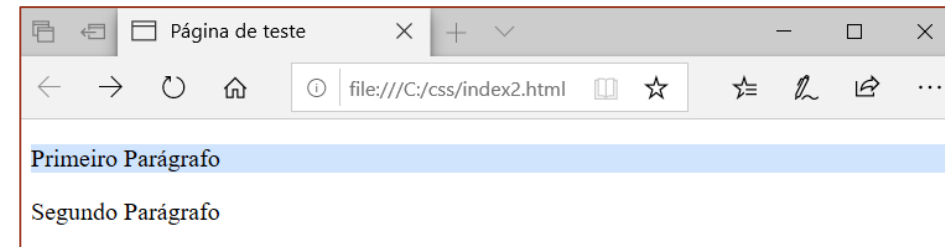
```



Seletores por identificador (#)

Seleciona elementos HTML com base no seu id

```
index2.html x
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo2.css">
7   </head>
8
9   <body>
10    <p id="p1">Primeiro Parágrafo</p>
11    <p id="p2">Segundo Parágrafo</p>
12  </body>
13
14 </html>
```



```
estilo2.css x
1 #p1 {
2   background-color: #d0e4fe;
3 }
```

Escolher cores em código hexadecimal:

https://www.w3schools.com/colors/colors_picker.asp

Identificador # antes do nome do id

Seletores por classe (.)

Seleciona elementos HTML com base na sua `class`

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Página de teste</title>
6      <link rel="stylesheet" href="estilo3.css">
7  </head>
8
9  <body>
10     <h1 class="centro">Cabeçalho 1</h1>
11     <h2 class="esquerda">Cabeçalho 2</h2>
12     <h3 class="centro">Cabeçalho 3</h3>
13     <h6 class="esquerda">Cabeçalho 4</h6>
14     <p class="centro">Primeiro Parágrafo</p>
15     <p class="esquerda">Segundo Parágrafo</p>
16 </body>
17 </html>

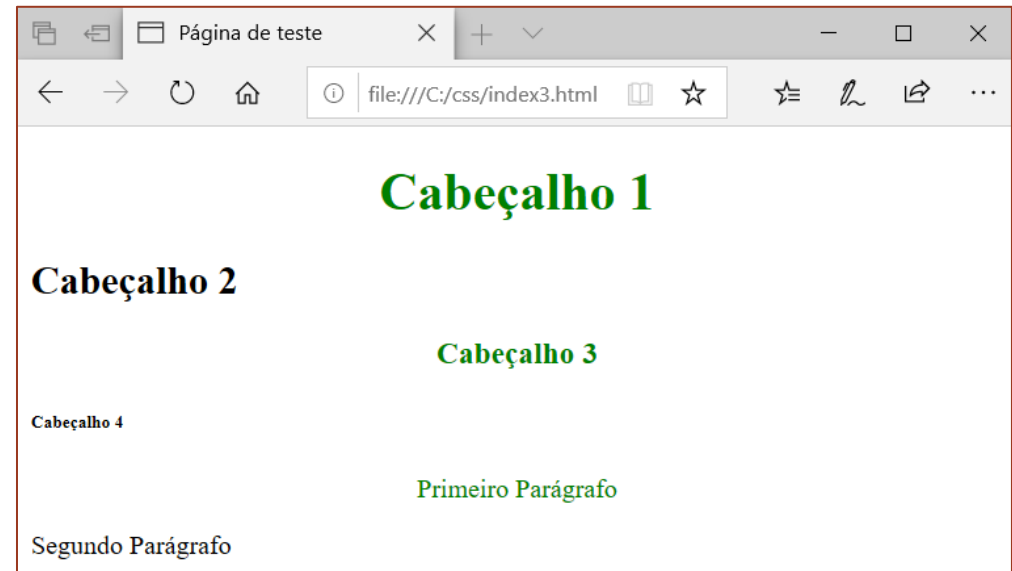
```

```

1  .centro {
2      text-align: center;
3      color: green;
4  }

```

Identificador `.` antes do nome da `class`



Seletores por classe (.)

Seleciona elementos HTML específicos com base na sua `class`

```

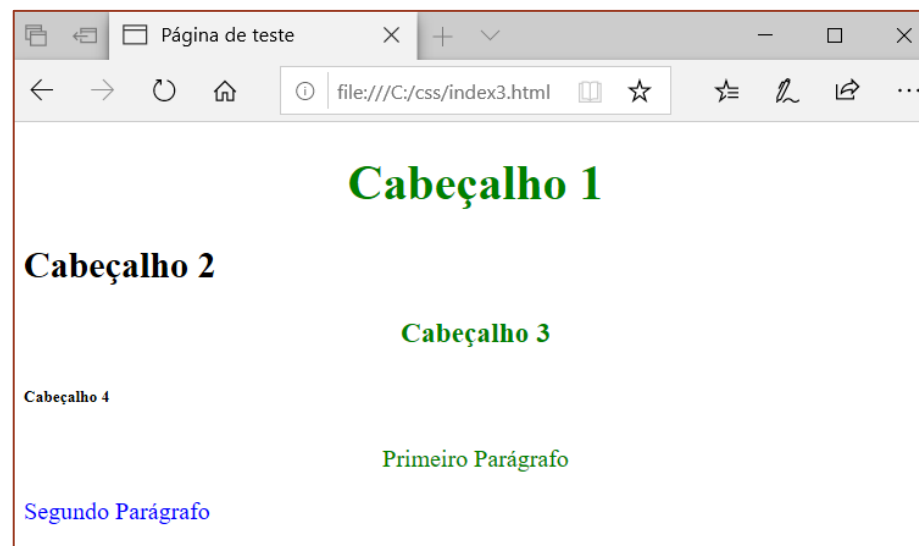
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Página de teste</title>
6      <link rel="stylesheet" href="estilo3.css">
7  </head>
8
9  <body>
10     <h1 class="centro">Cabeçalho 1</h1>
11     <h2 class="esquerda">Cabeçalho 2</h2>
12     <h3 class="centro">Cabeçalho 3</h3>
13     <h6 class="esquerda">Cabeçalho 4</h6>
14     <p class="centro">Primeiro Parágrafo</p>
15     <p class="esquerda">Segundo Parágrafo</p>
16
17 </body>
18 </html>

```

```

1  .centro {
2
3  }
4
5  p.esquerda {
6      color: blue;
7  }
8

```



Identificador `.` entre o nome da `tag` e o nome da `class`

Seletores por atributos

- Seleciona elementos HTML com base em dois critérios

```
index4.html x
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo4.css">
7   </head>
8
9   <body>
10    <a href="http://www.upt.pt" target="_blank">UPT nova janela</a><br/>
11    <a href="http://www.upt.pt" target="_self">UPT mesma janela</a>
12  </body>
13 </html>
```

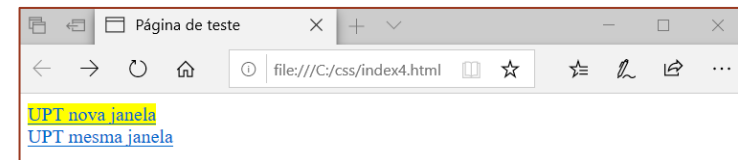
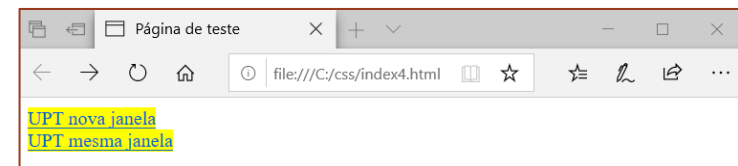
- **Atributos específicos:** é utilizada a sintaxe `seletor[atributo]` para selecionar elementos com um atributo específico.

```
estilo4.css x
1 a[target] {
2   background-color: yellow;
3 }
```

- **Valores de atributos:** é usada a sintaxe `seletor[atributo=valor]` para selecionar elementos com atributos e valores específicos.

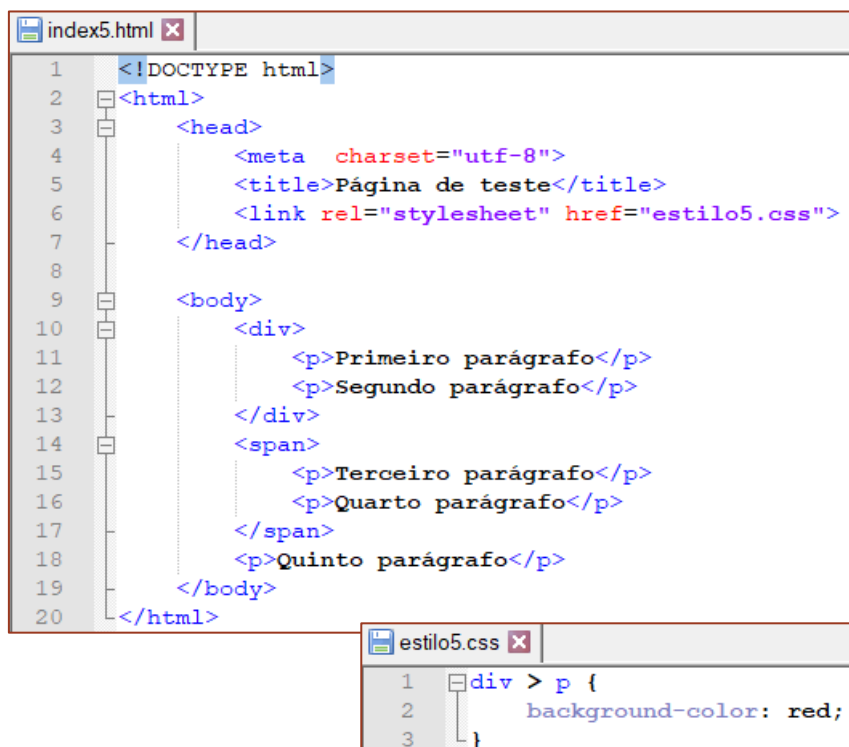
```
estilo4.css x
1 a[target="_blank"] {
2   background-color: yellow;
3 }
```

Seletor	Descrição
[atributo~="valor"]	Valor do atributo <u>contém</u> "valor" (com espaço antes e depois)
[atributo = "valor"]	Valor do atributo <u>começa</u> por "valor" (palavra completa)
[atributo^="valor"]	Valor do atributo <u>começa</u> por "valor"
[atributo\$="valor"]	Valor do atributo <u>termina</u> com "valor"
[atributo*="valor"]	Valor do atributo <u>contém</u> "valor"



Seletores baseados em combinadores

- É possível combinar vários seletores com os combinadores



```

index5.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo5.css">
7   </head>
8
9   <body>
10    <div>
11      <p>Primeiro parágrafo</p>
12      <p>Segundo parágrafo</p>
13    </div>
14    <span>
15      <p>Terceiro parágrafo</p>
16      <p>Quarto parágrafo</p>
17    </span>
18    <p>Quinto parágrafo</p>
19  </body>
20 </html>

estilo5.css
1 div > p {
2   background-color: red;
3 }
  
```

Seletores	Descrição
Descendentes ()	Todos os descendente do elemento
Filho direito (>)	Todos os filhos do elemento
Irmão direito (+)	Todos os irmãos a seguir ao elemento
Irmão geral (~)	Todos os irmãos



Seletores por pseudoclasses

- **Pseudoclasses** servem para definir estados de elementos (ex: passar o rato por cima, links visitados e não visitados, elementos de acordo com a sua posição)
- Sintaxe: `seletor:pseudoclasse{
 propriedade: valor;
}`

```

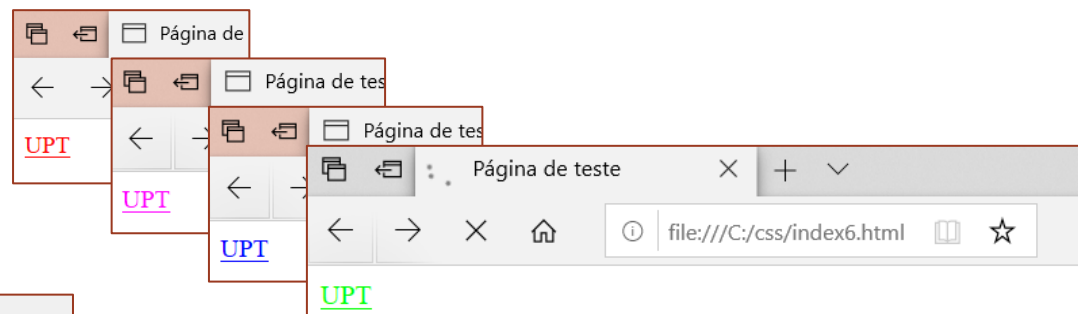
index6.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo6.css">
7   </head>
8   <body>
9     <a href="http://www.upt.pt">UPT</a>
10  </body>
11 </html>

```

```

estilo6.css
1 /*link não visitado*/
2 a:link {
3   color: #FF0000;
4 }
5
6 /*link visitado*/
7 a:visited {
8   color: #00FF00;
9 }
10
11 /*mouse over link*/
12 a:hover {
13   color: #FF00FF;
14 }
15
16 /*link selecionado*/
17 a:active {
18   color: #0000FF;
19 }

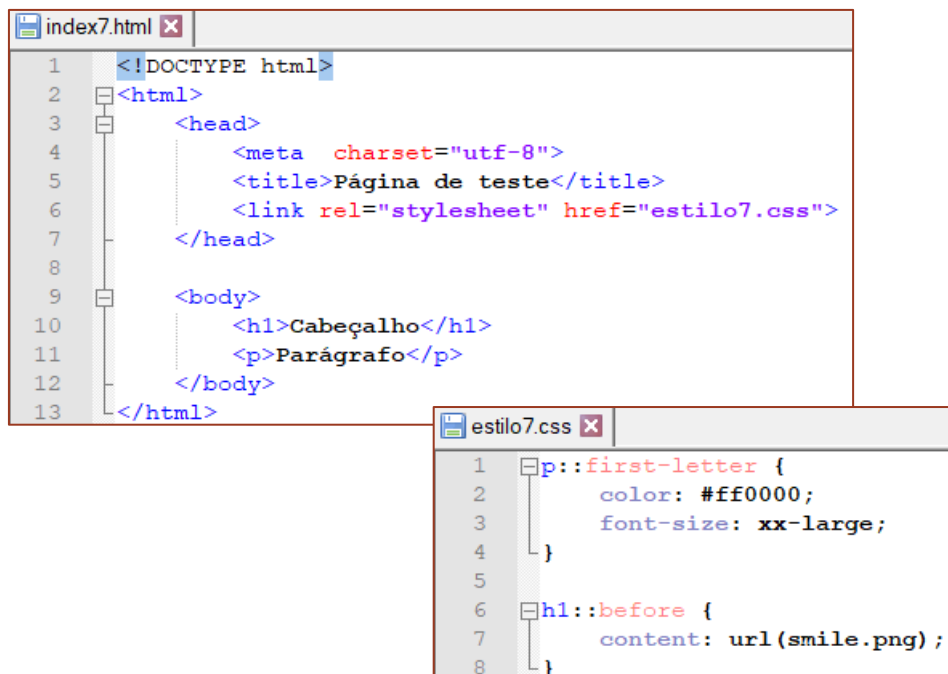
```



Seletores por pseudoelementos

- Pseudoelementos: usados para estilizar partes de um elemento (ex: primeira linha ou letra do elemento, inserir conteúdo antes/depois do elemento)
- Sintaxe:

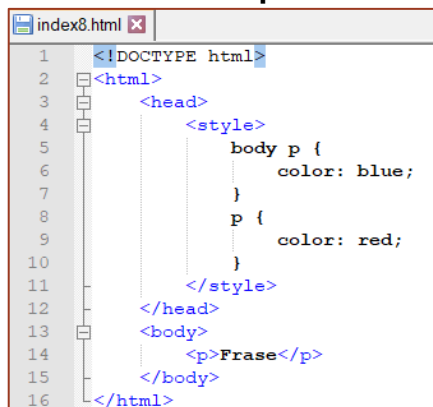
```
seletor:pseudoelemento{  
    propriedade: valor;  
}
```



Cascata

Quando é definida mais que uma regra para o mesmo elemento, há 3 métodos de “desempate”:

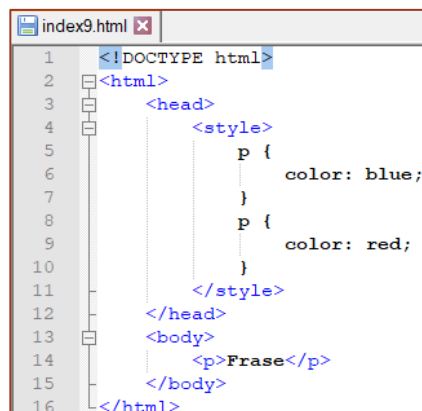
- **Cálculo de especificidade:** “ganha” a mais específica



```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <style>
5       body p {
6         color: blue;
7       }
8       p {
9         color: red;
10      }
11    </style>
12  </head>
13  <body>
14    <p>Frase</p>
15  </body>
16 </html>
  
```

- **Ordem de especificação:** “ganha” a última



```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <style>
5       p {
6         color: blue;
7       }
8       p {
9         color: red;
10      }
11    </style>
12  </head>
13  <body>
14    <p>Frase</p>
15  </body>
16 </html>
  
```

- **Anotação !important:** “ganha” a que tem a anotação



```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <style>
5       p {
6         color: blue !important;
7       }
8       p {
9         color: red;
10      }
11    </style>
12  </head>
13  <body>
14    <p>Frase</p>
15  </body>
16 </html>
  
```

Herança

- O HTML é formalizado de forma hierárquica
 - Um documento HTML pode ser representado pela sua árvore do documento (**DOM: Document Object Model**)
 - Estabelece relacionamentos (pais, filhos, irmãos, ancestrais, descendentes, ...)
- A herança permite que uns elementos herdem propriedades de outros
 - Reduz necessidade de escrita de código e ter de carregar documentos muito pesados
- Nem todas as propriedades podem ser herdadas (ex: background, border, width)
- Para determinar herança:
 - inherit: define que o valor deve ser herdado
 - initial: define que o elemento deve assumir o definido para o antecessor pelas pré-definições do browser. Se não estiver definido, herda o estilo definido por CSS para o antecessor
 - unset: redefine a propriedade para o seu valor natural (herdada se for naturalmente herdada)

```

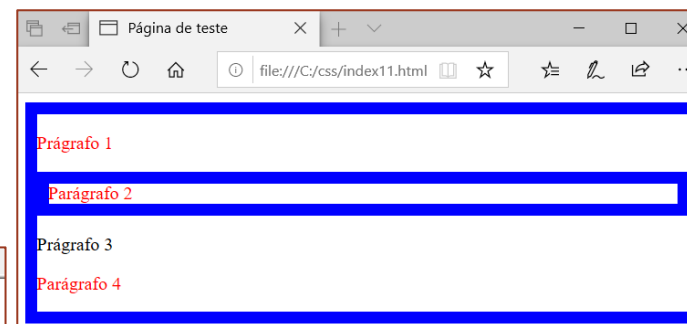
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo11.css">
7   </head>
8
9   <body>
10    <div class="container">
11      <p>Parágrafo 1</p>
12      <p class="inherit">Parágrafo 2</p>
13      <p class="initial">Parágrafo 3</p>
14      <p class="unset">Parágrafo 4</p>
15    </div>
16  </body>
17 </html>

```

```

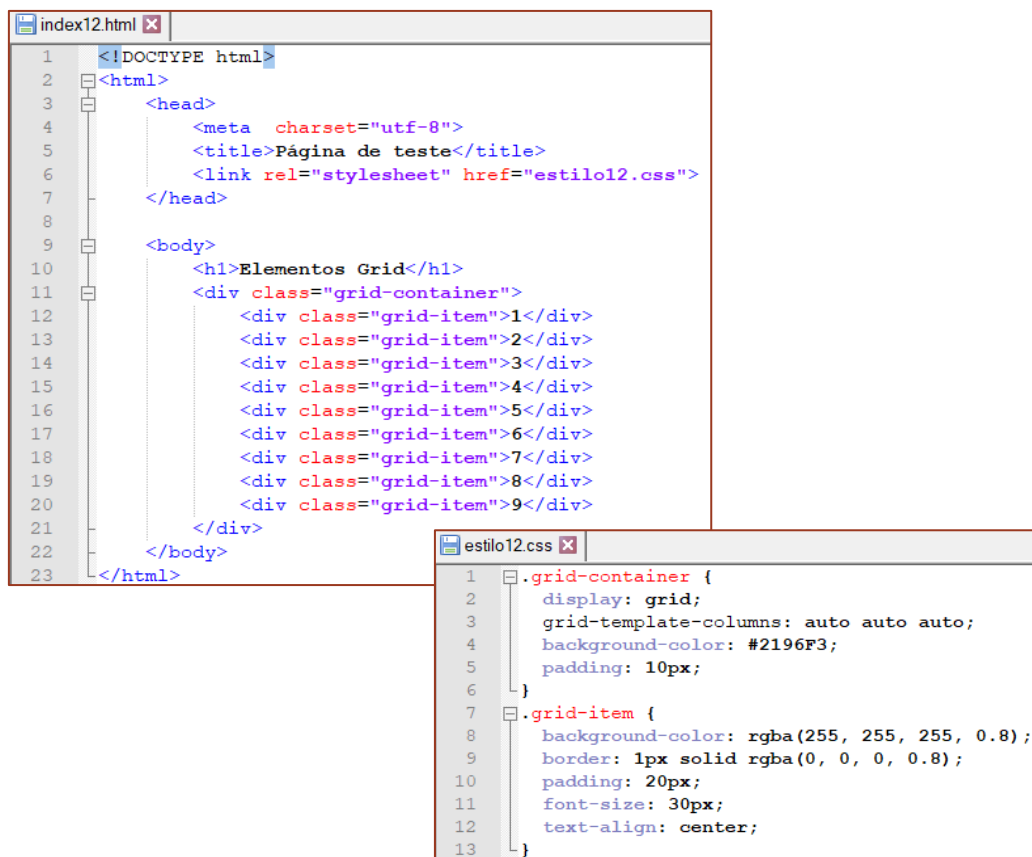
1 .container {
2   color: red;
3   border: 10px solid blue;
4 }
5 .inherit {color: inherit; border: inherit;}
6 .initial {color: initial; border: initial;}
7 .unset {color:unset; border: unset}

```



Grid CSS

- Um *layout* em grelha (*grid*) deve ter um elemento pai com a propriedade **display** definida como **grid** ou **inline-grid**.
- Filhos diretos desse elemento tornam-se automaticamente em elementos da grelha.



The image shows a code editor with two files: `index12.html` and `estilo12.css`.

index12.html:

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Página de teste</title>
6     <link rel="stylesheet" href="estilo12.css">
7   </head>
8
9   <body>
10    <h1>Elementos Grid</h1>
11    <div class="grid-container">
12      <div class="grid-item">1</div>
13      <div class="grid-item">2</div>
14      <div class="grid-item">3</div>
15      <div class="grid-item">4</div>
16      <div class="grid-item">5</div>
17      <div class="grid-item">6</div>
18      <div class="grid-item">7</div>
19      <div class="grid-item">8</div>
20      <div class="grid-item">9</div>
21    </div>
22  </body>
23 </html>

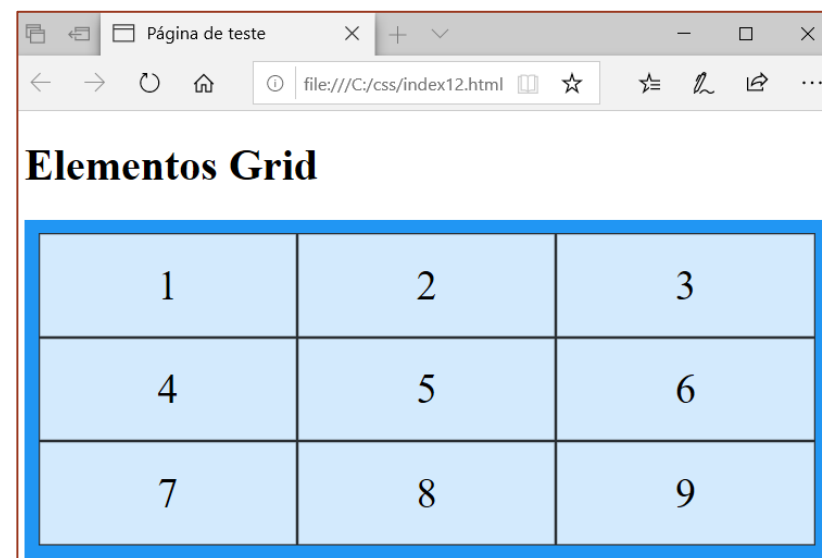
```

estilo12.css:

```

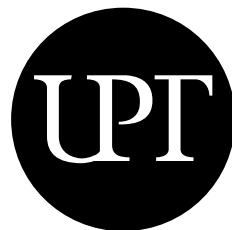
1 .grid-container {
2   display: grid;
3   grid-template-columns: auto auto auto;
4   background-color: #2196F3;
5   padding: 10px;
6 }
7
8 .grid-item {
9   background-color: rgba(255, 255, 255, 0.8);
10  border: 1px solid rgba(0, 0, 0, 0.8);
11  padding: 20px;
12  font-size: 30px;
13  text-align: center;
14 }

```



Boas Práticas

- Usar metodologias para nomeação de classes em CSS. Exemplos:
 - **OOCSS**: *Object Oriented CSS* <http://oocss.org/>
 - **BEM**: *Block, Element, Modifier* <http://getbem.com/introduction/>
 - **ACSS**: *Atomic CSS* <http://github.com/nemophrost/atomic-css>
 - **SMACSS**: *Scalable and Modular Architecture for CSS* <http://smacss.com/>
- Usar regras externas para:
 - Ser possível reutilização de código entre documentos e consistência em sites com mais de uma página
 - Separar o que é estrutural (HTML) do que é formatação (CSS)
 - Melhorar legibilidade, manutenção e modularização do código
- Dominar o conceito de *cascading* para compreender as precedências de regras
- Aplicar o conceito de herança para reduzir a quantidade de regras necessárias



UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.