



FUNDAMENTOS DA PROGRAMAÇÃO WEB

AULA 2



Profª Margarete Klamas



CONVERSA INICIAL

Agora chegou a vez de iniciarmos nossos estudos em CSS.

O objetivo desta etapa é apresentar as formatações básicas em CSS, que servem de base para o desenvolvimento de aplicações Web para os mais variados fins. Ao concluir os cinco temas propostos, você será capaz de aplicar estilos (formatações) em páginas HTML, bem como poderá fazer alterações em páginas criadas por outros desenvolvedores.

Abordaremos sobre o *Cascading Style Sheets* (CSS): definição, sintaxe, seletores e como aplicar CSS e hierarquia CSS. No tema 2 iremos ver comentários em CSS, cores, fundo (*background*), bordas, *margin* e preenchimento (*padding*). No tema 3 estudaremos as propriedades de largura (*width*), altura (*height*), formatação de texto/fonte e modelo de caixa (*box model*). No tema 4 estudaremos *float*, *clear*, elementos *block* e *inline*. No tema 5 abordaremos sobre *display*, *position*, *z-index* e *overflow*.

No desenvolvimento das explicações você irá encontrar QR Codes com vídeos que contêm exemplos.

Agora é o momento de praticar CSS.

Bons estudos!

TEMA 1 – CASCADING STYLE SHEETS (CSS)

1.1 Definição

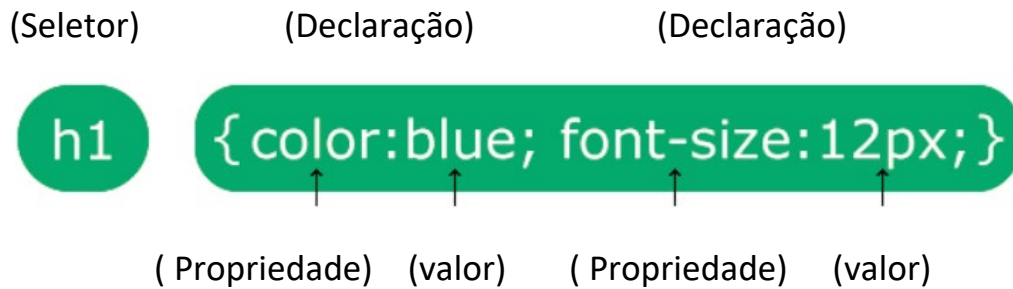
HTML foi desenvolvido para ser uma linguagem de marcação e estruturação do conteúdo de páginas. Não utilizamos HTML para formatar o visual das páginas.

Para especificar a parte visual temos o *Cascading Style Sheets* (CSS) ou, em português, Folhas de Estilo em Cascata. Segundo o W3C: “Folha de estilo em cascata é um mecanismo simples para adicionar estilos (por exemplo: fontes, cores, espaçamentos) aos documentos web”.

O CSS tem como finalidade especificar a apresentação da página HTML + CSS: “HTML para estruturar os conteúdos e CSS para apresentá-los” (Silva, 2015).



1.2 Sintaxe



Descrição da imagem: a imagem mostra a sintaxe do CSS interno.

```
h1{color: blue; font-size:12px;}
```

<h1> Seletor

A formatação deve estar entre {} (chaves)

color: blue; font-size:12px; são declarações, separadas por ponto e vírgula

color e font-size são declarações

blue e 12px são valores

O seletor aponta para o elemento HTML que você deseja estilizar. No exemplo anterior, o elemento `<h1>` da página será estilizado.

O bloco de declaração (*declaration*) contém uma ou mais declarações separadas por ponto e vírgula. Cada declaração inclui um nome de propriedade (*property*) CSS e um valor (*value*), separados por dois-pontos. Várias declarações CSS são separadas por ponto e vírgula e os blocos de declaração são cercados por chaves.

No exemplo anterior, a 1ª declaração *color: blue*; vai deixar a cor da fonte azul. A 2ª declaração *font-size: 12px*; irá deixar o tamanho da fonte em 12 pixels.

Na(s) página(s) em que essa CSS for aplicada, todos os elementos `<h1>` serão afetados por essa formatação.

1.3 Seletores

Os tipos de seletores são:

- seletores simples: definem elementos com base no nome, id e classe;
- seletores de combinação: selecionam elementos com base em um relacionamento específico entre eles;
- seletores de pseudoclassem: selecionam elementos com base em determinado estado;



- seletores de pseudoelementos: selecionam e estilizam parte de um elemento;
- seletores de atributo: selecionam elementos com base em um atributo ou valor de atributo.

1.3.1 Seletores ID

O seletor id usa o atributo id de um elemento HTML para selecionar um elemento específico. O id de um elemento é único dentro de uma página, então, o seletor de id é usado para selecionar um elemento único! Para selecionar um elemento com um id específico, escreva um caractere hash (#), seguido do id do elemento.

Exemplo:

```
#paragrafo{
    text-align: justify;
    color: green;
}
```

1.3.2 Seletores *class*

O seletor de classe seleciona elementos HTML com um atributo de classe específico. Para selecionar elementos com uma classe específica, escreva um caractere ponto (.), seguido do nome da classe.

Exemplo:

```
.p.dois{
    text-align: center;
    color: gray;
}
```

Pode-se aplicar mais de uma *class* em um elemento:

```
<p class="p.dois large"> Este parágrafo receberá duas class.
</p>
```

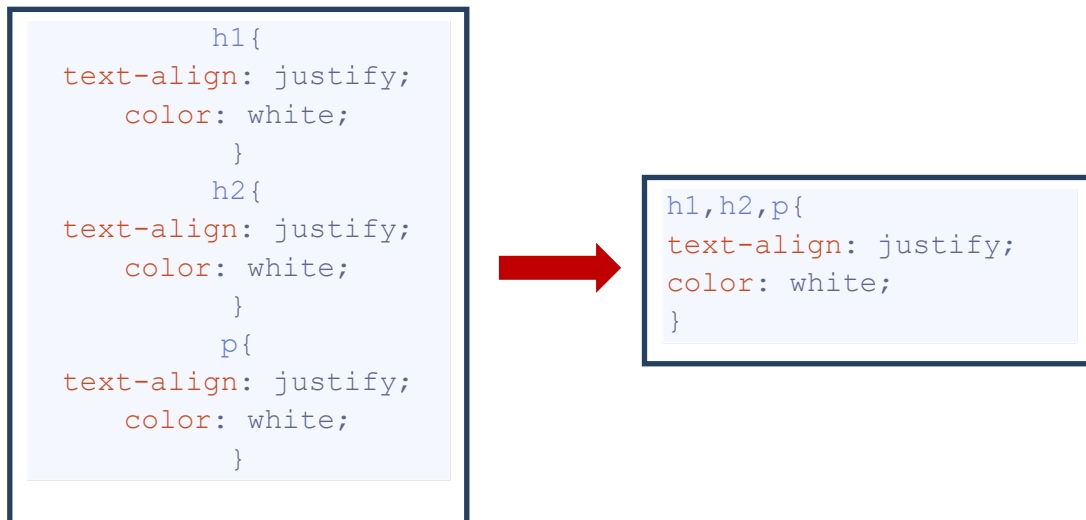
1.3.3 Seletor universal

Seleciona todos os elementos HTML da página, para aplicar a formatação.

```
*{ color: white;}
```

1.3.4 Agrupamento de seletores

Observe a seguinte formatação:



As formatações da imagem à esquerda serão aplicadas aos elementos `<h1>`, `<h2>` e `<p>` e são as mesmas, texto justificado e cor de fonte branca. Nessa situação, pode-se aplicar a formatação agrupando os elementos como se apresentou na imagem à direita.

1.3.5 Como aplicar CSS

Existem três maneiras de inserir uma folha de estilo:

- CSS externo;
- CSS interno;
- CSS embutido (*inline*).

1.3.5.1 CSS Externo

Com uma folha de estilo externa, você pode alterar a aparência de um site inteiro alterando apenas um arquivo. Cada página HTML deve incluir uma referência (`href`) ao arquivo de folha de estilo externo dentro do elemento `<link>`, dentro da seção `<head>`.



```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="estilo.css">
  <title>CSS Externo</title>
</head>
<body>
  <h1>Título</h1>
  <p> Aqui tem um parágrafo</p>
</body>
</html>
```

No exemplo anterior, no `<head>`, há um link a um arquivo `estilo.css`, onde há declarações CSS (a seguir).

```
body{
  background-color:
chocolate;
}
h1 {
  color: navy;
}
```

1.3.5.2 CSS interno

Uma folha de estilo interna pode ser usada se uma única página HTML tiver um estilo exclusivo. O estilo interno é definido dentro do elemento `<style>`, dentro da seção *head*.



```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <title>CSS Interno</title>
  <style>
    *{
      color: white;
    }
    body{
      background-color: black;
    }
  </style>
</head>
<body>
  <h1>Título</h1>
  <p> Aqui tem um parágrafo</p>
</body>
</html>
```

1.3.5.3 CSS *inline*

Um estilo *inline* pode ser usado para aplicar um estilo exclusivo para um único elemento, aplicado diretamente na *tag*. Para usar estilos *inline*, adicione o atributo *style* ao elemento. O atributo *style* pode conter qualquer propriedade CSS.

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <title>CSS Inline</title>
</head>
<body>
  <h1 style="color:blue; text-align: center;">Título</h1>
  <p style="color:red;"> Aqui tem um parágrafo</p>
</body>
</html>
```



1.4 Hierarquia CSS

Quando houver mais de um estilo aplicado a um elemento, é utilizada a seguinte regra:

1. CSS *inline* (aplicado diretamente na *tag*);
2. folhas de estilos internas e externas;
3. padrão do *browser*.

TEMA 2 – CSS (COMENTÁRIOS, CORES, FUNDO, BORDAS, MARGEM E PREENCHIMENTO)

2.1 Comentários em CSS

Os comentários são usados para explicar o código e podem ajudar você a editar o código-fonte posteriormente. Os comentários não serão exibidos pelos navegadores.

Um comentário CSS é colocado dentro do **<style>** e começa com **/*** e termina com ***/**:

```
<style>
  /*assim se marca comentário*/
  p{
    color:red;
  }
</style>
```

2.2 Cores em CSS

Podemos aplicar cores em textos, fundos e bordas.

As cores podem ser especificadas usando nomes de cores predefinidos ou valores RGB, HEX, HSL, RGBA ou HSLA.



2.2.1 Cores predefinidas em CSS (alguns exemplos)



Descrição da imagem: a imagem mostra retângulos coloridos nas cores predefinidas CSS: *Tomato*, *Orange*, *DodgerBlue*, *MediumSeaGreen*, *Gray*, *SlateBlue*, *Violet*, *LightGray*.

A tabela completa pode ser obtida pelo link disponível em: https://www.w3schools.com/colors/colors_names.asp. Acesso em: 19 dez. 2022.

2.2.2 Cores RGB

Um valor de cor RGB representa fontes de luz *red* (vermelha), *green* (verde) e *blue* (azul).

Para se especificar uma cor com um valor RGB, usando esta sintaxe: **RGB (vermelho, verde, azul)**.

Cada parâmetro (vermelho, verde e azul) define a intensidade da cor entre 0 e 255.

Por exemplo, RGB (255, 0, 0) é exibido como vermelho, porque vermelho é definido como seu valor mais alto (255) e os outros são definidos como 0.

Para exibir preto, defina todos os parâmetros de cor para 0, assim: RGB (0, 0, 0).

Para exibir branco, defina todos os parâmetros de cor para 255, assim: RGB (255, 255, 255).

2.2.3 Cores HEX

Uma cor hexadecimal é especificada com: #RRGGBB, em que os inteiros hexadecimais RR (vermelho), GG (verde) e BB (azul) especificam os componentes da cor.

Em CSS, uma cor pode ser especificada usando um valor hexadecimal na forma:

#rrggb _

em que rr (vermelho), gg (verde) e bb (azul) são valores hexadecimais entre 00 e ff.



Por exemplo, `#ff0000` é exibido como vermelho, porque o vermelho é definido com o valor mais alto (ff) e os outros são definidos com o valor mais baixo (00).

Para exibir preto, defina todos os valores para 00, assim: `#000000`.

Para exibir branco, defina todos os valores para ff, assim: `#ffffff`.

2.2.4 Cores HSL

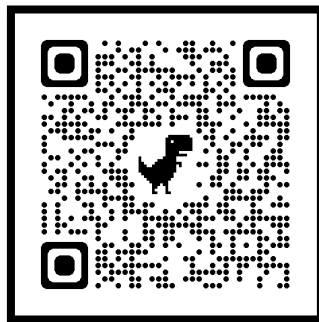
HSL significa **hue** (matiz), **saturation** (saturação) e **lightness** (leveza).

Matiz é um grau na roda de cores de 0 a 360. 0 é vermelho, 120 é verde e 240 é azul.

A saturação é um valor percentual. 0% significa um tom de cinza e 100% é a cor completa.

A leveza também é uma porcentagem. 0% é preto, 50% não é claro nem escuro, 100% é branco.

Vamos ver uma paleta de cores?



Saiba mais

Disponível em: <https://n-cpuninter.github.io/FUNDAMENTOS-DE-DESENVOLVIMENTO-WEB/html/cores.html>.

Paleta:

<https://color.adobe.com/pt/create/color-wheel>. Acessos em: 20 dez. 2022.

2.3 Background (plano de fundo)

Podemos ter para *background*:

- *background-color*: a cor de plano de fundo pode ser aplicada a qualquer elemento.

Exemplo para digitar:



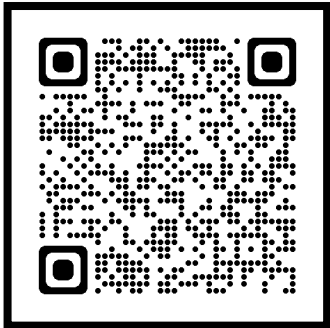
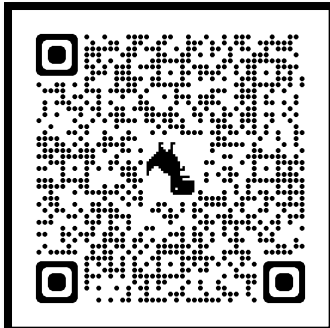
```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <title>CSS Interno</title>
  <style>
    body{
      background-color:
black;
      color: white;
    }
  </style>
</head>
<body>
  <h1>Título</h1>
  <p> Aqui tem um parágrafo</p>
</body>
</html>
```

No código anterior inserimos o CSS interno, o que significa que foi aplicado na página. O que identifica o CSS interno é que está dentro da *tag* `<head>`. A página foi estilizada com cor de fundo (*background-color*) preta e cor (*color*) de texto branco.

A propriedade *background-image* especifica uma imagem a ser usada como plano de fundo de um elemento. Por padrão, a imagem é repetida para cobrir todo o elemento.

```
body{ background-color: url("flor.gif");}
```

A seguir, veja vídeos com códigos para você digitar com explicações:

Cor de fundo em vários elementos.	Imagem como plano de fundo
	
Disponível em: < https://n-cpuninter.github.io/FUNDAMENTOS-DE-	Disponível em: < https://n-cpuninter.github.io/FUNDAMENTOS-DE-

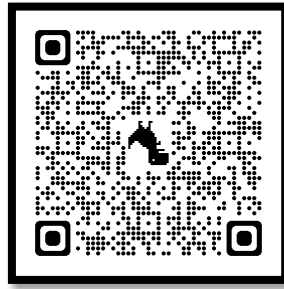


DESENVOLVIMENTO- WEB/html/plano_de_fundo.html>. Acesso em: 19 dez. 2022.	DESENVOLVIMENTO- WEB/html/plano_de_fundo_imagem.html>. Acesso em: 19 dez. 2022.
--	---

2.4 Bordas

A propriedade *border* define a espessura, o estilo e a cor das bordas do box. Pode-se aplicar as propriedades separadamente, se desejar, pois a borda tem quatro lados: *top*, *right*, *bottom* e *left*.

Propriedade	Descrição
border-width border-top-width: 1px; border-right-width: 2px; border-bottom-width: 3px; border-top-width: 4px;	Define a espessura da borda Sintaxe abreviada: border-width: 1px; 2px; 3px; 4px;
border-color border-top-color: red; border-right-color: green; border-bottom-color: cyan; border-top-color: black;	Define a cor da borda Sintaxe abreviada: border-color: red; green; cyan; black;
border-style border-top-style: solid; border-right-style: ridge; border-bottom-style: double; border-top-style: dotted;	Define o estilo da borda Sintaxe abreviada: border-style: solid; ridge; double; dotted;
Border border-top: 3px solid red; border-right: 2px dotted green; border-bottom: 1px double cyan; border-top: 5px inset black;	Definição abreviada para borda Sintaxe abreviada para bordas iguais nos quatro lados: border: 3px dotted black;
border-radius: Propriedade usada para adicionar bordas arredondadas border-radius: 25px; border-radius: 50px 20px; Exemplo no QR Code a seguir.	



Disponível em: <https://n-cpuninter.github.io/FUNDAMENTOS-DE-DESENVOLVIMENTO-WEB/html/Bordas_arredondadas.html>. Acesso em: 20 dez. 2022.

Propriedades de estilos de bordas:

Borda pontilhada (dotted).

Borda tracejada (dashed).

Borda Sólida (solid).

Borda dupla (double).

Borda tipo groove.

Borda tipo ridge .

Borda tipo inset.

Borda tipo outset.

Sem bordar.

Borda oculta.

Borda tipo mixed.

Descrição da imagem: a imagem apresenta modelos de tipos de bordas: pontilhada (dotted), tracejada (dashed), sólida (solid), dupla (double), groove, ridge, inset, outset, sem bordas e borda oculta. Código da imagem anterior disponível em: <https://github.com/n-cpuninter/fundamentos-de-desenvolvimento-web/blob/main/html/bordas_estilos.html>. Acesso em: 19 dez. 2022.

2.5 Margin

As margens são usadas para criar espaço ao redor dos elementos.

CSS tem propriedades para especificar a margem para cada lado de um elemento:

margin-top

margin-right

margin-bottom

margin-left

```
p{  
  margin-top: 100px;  
  margin-bottom: 100px;
```



```
margin-right: 150px;  
margin-left: 80px;  
}
```

Se a propriedade tiver quatro valores:

margin: 25px 50px 75px 100px;

- a margem superior é de 25px
- a margem direita é 50px
- a margem inferior é de 75px
- margem esquerda é 100px

```
p{ margin: 25px 50px 75px 100px;}
```

Se a propriedade tiver três valores:

margin: 15px 25px 55px;

- a margem superior é de 15px
- as margens direita e esquerda são 25px
- a margem inferior é de 55px

Se a propriedade tiver dois valores:

margin: 20px 25px;

- as margens superior e inferior são 20px
- as margens direita e esquerda são 25px

Se a propriedade tiver um valor:

margin: 25px;

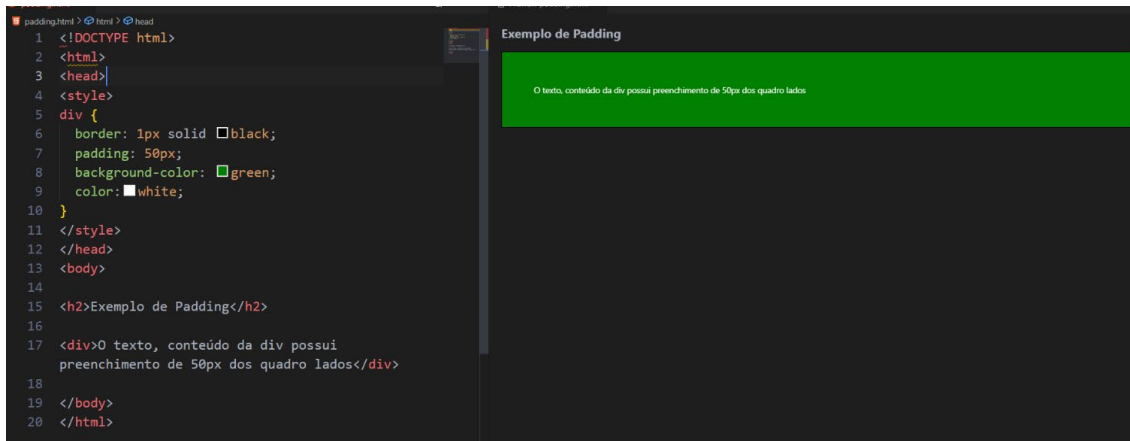
- todas as quatro margens são 25px

2.6 *Padding* (preenchimento)

O preenchimento é usado para criar espaço ao redor do conteúdo de um elemento, dentro de qualquer borda definida.

O preenchimento pode ter valores diferentes nos quatro lados, e se formata da mesma maneira que a *margin*. Veja, a seguir, um exemplo de *padding* com mesmo valor.

Exemplo:



Descrição da imagem: a imagem apresenta um exemplo de *padding*. No exemplo foi aplicado um *padding* de 50px numa div. O código do exemplo está disponível em: <<https://github.com/n-cpuninter/fundamentos-de-desenvolvimento-web/blob/main/html/padding.html>>. Acesso em: 20 dez. 2022.

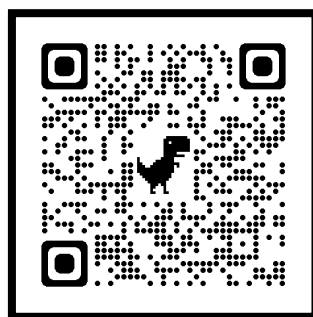
TEMA 3 – CSS (*WIDTH*, *HEIGHT*, FORMATAÇÃO DE TEXTO/FONTE. *BOX MODEL*)

3.1 *Width* (largura) e *height* (altura)

As propriedades *height* e *width* usadas para definir a altura e a largura de um elemento.

A propriedade *max-width* é usada para definir a largura máxima de um elemento.

Veja o exemplo no vídeo de *width* e *max-width*:



Disponível em: <https://n-cpuninter.github.io/FUNDAMENTOS-DE-DESENVOLVIMENTO-WEB/html/v_max_width.html>. Acesso em: 20 dez. 2022.

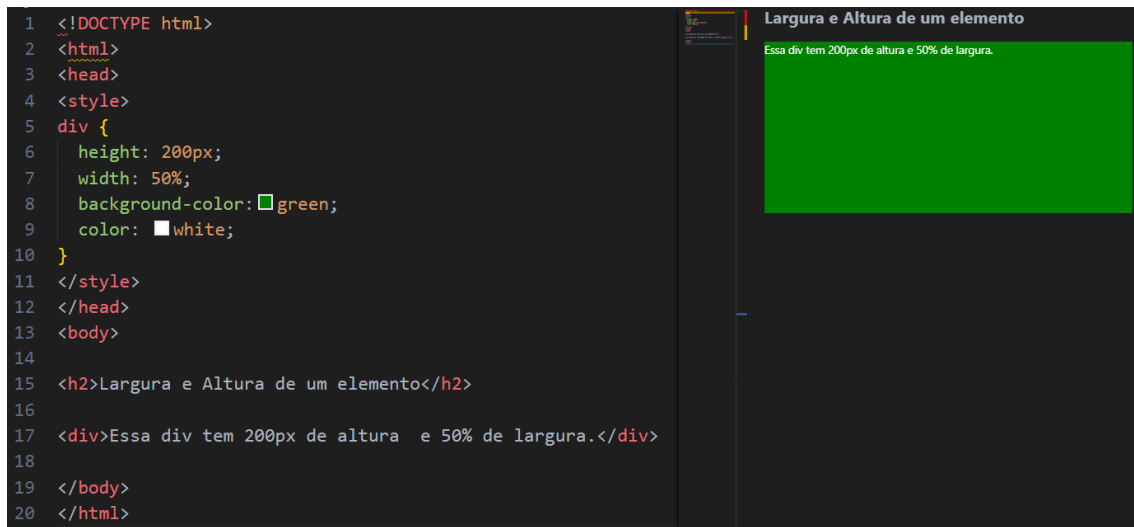
Podem ter os seguintes valores:

- *auto*- este é o padrão. O navegador calcula a altura e a largura;
- *length*- define a altura/largura em px, cm etc.;



- %- define a altura/largura em porcentagem do bloco que contém.

Exemplo:



Descrição da imagem: a imagem demonstra um exemplo de *height* (altura) com 200px e largura (*width*) de 50% aplicados em uma div. O código utilizado no exemplo está disponível em: https://github.com/n-cpuninter/fundamentos-de-desenvolvimento-web/blob/main/html/larg_altura.html. Acesso em: 20 dez. 2022.

3.2 Formatação de texto

Podemos aplicar diversas propriedades ao texto, como cor de texto, alinhamento, tipo de fonte, tamanho de fonte e decorações.

Propriedade	Descrição
color	Define a cor do texto
text-align	Define o alinhamento para o texto text-align: left; (<i>center, right, justify</i>)
text-decoration-line <style> h1 { text-decoration: overline;} h2 {text-decoration: line-through;}	adicionar uma linha de decoração ao texto. Exemplos:



```
h3 { text-decoration: underline;}  
p.ex {text-decoration: underline overline underline;}  
</style>
```

decoração de texto overline

~~Decoração de texto de linha~~

Sublinhar decoração de texto

Decoração de texto overline e sublinhado.

Nota: Não é recomendado sublinhar texto que não seja um link,

Obs.: pode adicionar cor a decoração:

```
h1 { text-decoration-line: overline; text-decoration-color: red;}
```

```
h2 { text-decoration-line: line-through; text-decoration-color: blue;}
```

text-decoration-style

Define o estilo da linha de decoração

text-decoration-style: solid;

text-decoration-style: dotted;

text-decoration-style: dashed;

text-decoration-style: wavy;

text-decoration-style: wavy;

Exemplo:


```
h2 {  
  text-decoration-line: underline;  
  text-decoration-style: double;  
}
```

Pode adicionar o text-decoration-color: blue; para testar



3.2.1 Formatação de fonte

Diferença entre fonte:

	
Sans-serif	Serif

Exemplos de fontes:

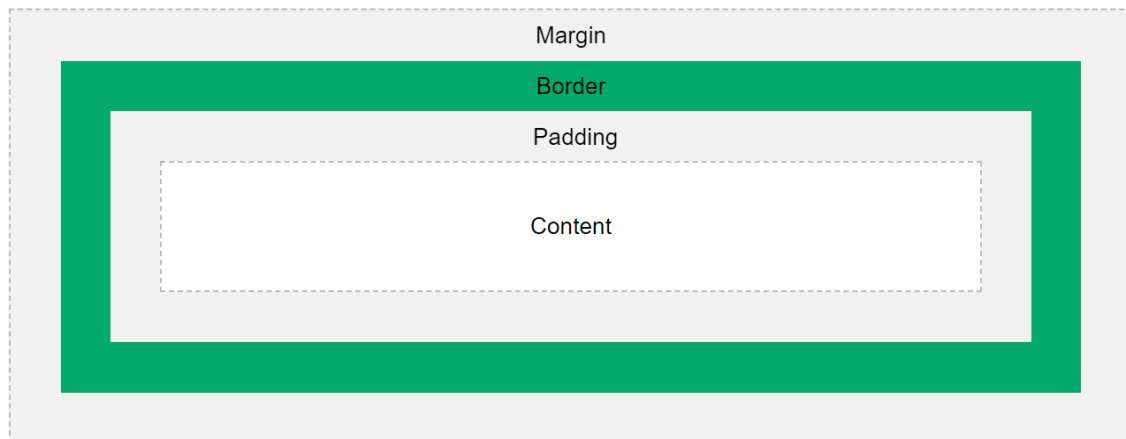
Tipos	Exemplos
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica
Propriedade	Descrição
font-family	Define a família de fonte Exemplo: <pre>h2 { font-family: Arial, Helvetica, sans-serif; }</pre>
font-size	Define o tamanho da fonte Exemplo: <pre>h2 { font-size: 30px; }</pre>

3.3 Box model

O "box model" facilita o entendimento de medidas para o *design* e o *layout*.



O *box-model* é basicamente uma caixa que envolve cada elemento HTML. A imagem seguinte ilustra o *box-model*.



Fonte: w3schools.

Descrição da imagem: a imagem mostra quatro retângulos, um dentro do outro. No retângulo externo temos a margem (*margin*), interno a este temos: a borda (*border*), depois o preenchimento (*padding*) e o conteúdo (*content*).

Se aplicar o seguinte estilo em uma `div`¹:

```
div {  
  width: 350px;  
  padding: 10px;  
  border: 5px dotted green;  
  margin: 0;  
}
```

A medida final será de 380px, pois:

350px (largura)
+ 20px (preenchimento
esquerdo + direito)
+ 10px (borda esquerda +
direita)
+ 0px (margem esquerda +
direita)
= 380px

div 380 x 48.4

Conteúdo

<style>

div {
 width: 350px;
 padding: 10px;
 border: 5px dotted green;
 margin: 0;
 background-color: burlywood;
}

</style>

¹ <div> Utilizamos <div> para criar containers. É um elemento sem valor semântico. Pode ser utilizado como um recurso visual, com CSS pode-se adicionar bordas, cores de fundo.



Descrição da imagem: a imagem demonstra o *box-model*. Exemplificando que, inicialmente, se planejarmos o tamanho do elemento, devemos acrescentar na medida final os valores da margem (*margin*), borda (*border*) e do preenchimento (*padding*). No exemplo foi atribuída a largura (*width*) de 350px, mas o valor da largura total da div será 380px, acrescentando os valores dos elementos citados.

TEMA 4 – CSS (*FLOAT, CLEAR, ELEMENTOS BLOCK E INLINE*)


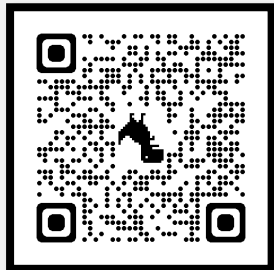
4.1 *Float e clear*

O **Float** auxilia no posicionamento de elementos e pode ter um dos seguintes valores:

- *Left*: o elemento flutua à esquerda de seu contêiner;
- *Right*: o elemento flutua à direita de seu contêiner;
- *None*: o elemento não flutua (será exibido exatamente onde ocorre no texto). É padrão;
- *Inherit*: o elemento herda o valor *float* de seu pai.

A propriedade **clear** pode ser utilizada quando desejar posicionar o próximo elemento abaixo (não à direita ou à esquerda).

Assista o exemplo de aplicação de *float*:

<i>Float</i>	<i>Float e clear</i>
	
Acesso em: < https://n-cpuninter.github.io/FUNDAMENTOS-DE-DESENVOLVIMENTO-WEB/html/v_float.html >. Acesso em: 20 dez. 2022.	Disponível em: < https://n-cpuninter.github.io/FUNDAMENTOS-DE-DESENVOLVIMENTO-WEB/html/v_clear.html >. Acesso em: 20 dez. 2022.

4.2 Elementos *block* e *inline*

Diagramar o *layout* de uma página pode se tornar trabalhoso. Assim, é fundamental o entendimento de como alguns elementos HTML são exibidos. Os



modos de exibição são *block* e *inline*. Os elementos do tipo *block* (bloco) são exibidos um abaixo do outro. Os elementos do tipo *inline* são exibidos um à esquerda do outro (na mesma linha).

4.3 Elementos do tipo *block*

<address>

<article>

<aside>

<canvas>

<dd>

<div>

<dl>

<dt>

<fieldset>

<figcaption>

<figure>

<hr>

<main>

<nav>

<noscript>

<output>

<p>

<pre>

<section>

<table>



4.4 Elementos do tipo *inline*

| | |
|------------------------------|-------------------------------|
| <code><a></code> | <code><script></code> |
| <code><acronym></code> | <code><select></code> |
| <code></code> | <code><small></code> |
| <code><bdo></code> | <code></code> |
| <code><big></code> | <code></code> |
| <code>
</code> | <code><sub></code> |
| <code><button></code> | <code><sup></code> |
| <code><cite></code> | <code><textarea></code> |
| <code><code></code> | <code><time></code> |
| <code><dfn></code> | <code><tt></code> |
| <code></code> | <code><var></code> |
| <code><i></code> | |

TEMA 5 – CSS (*DISPLAY, POSITION, Z-INDEX*)

5.1 *Display*

O *Display* especifica como um elemento será exibido na tela. Já vimos que os elementos têm propriedades padrões: *block* ou *inline*. Podemos utilizar a propriedade *display* para alterar esse padrão de exibição.

O *Display* pode ser definido como: *block*, *inline* ou *none*.

Os elementos `` e `` são elementos do tipo *block*, podemos alterá-los para o tipo *inline*.

Exemplo:

| | |
|--|--|
| <pre><!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <title>Document</title> <style> li { display: inline; } </style> </head> HTML CSS JavaScript </pre> | <p>Menu Horizontal:</p> <p>HTML CSS JavaScript</p> |
|--|--|



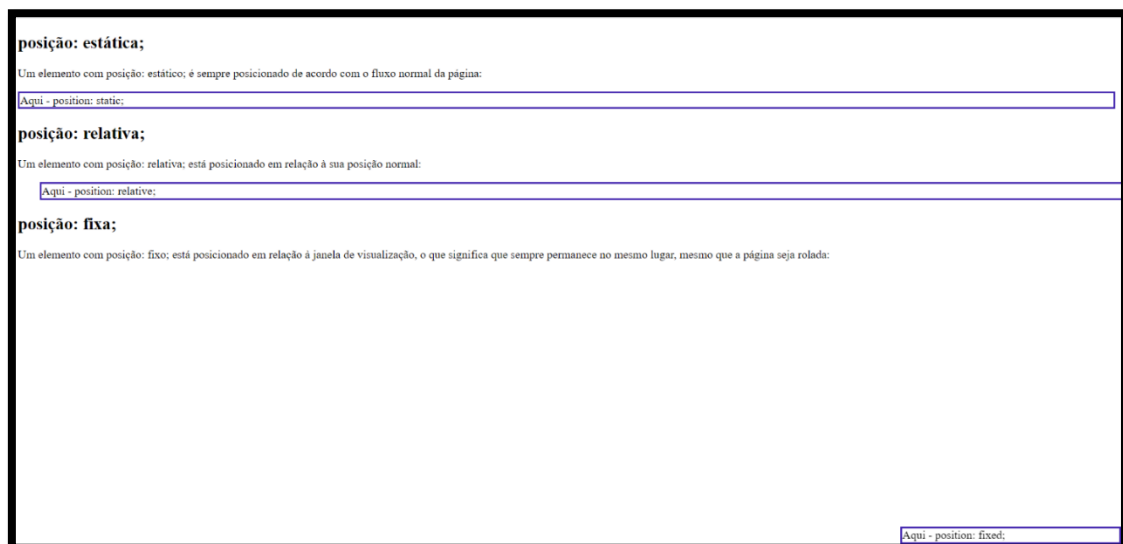
<pre></style> </head> <body> <p>Menu Horizontal:</p> HTML CSS JavaScript </body> </html></pre>	<p>Acima temos a visualização do menu horizontal</p> <p>Apague a linha <i>display: inline</i> e visualize no <i>browser</i></p>
--	---

5.2 Position e Z-index

5.2.1 Position

Assim como a propriedade *float*, *position* e *z-index* ajudam no posicionamento de elementos do *layout*. *Position* pode ter os valores: estático, fixo, absoluto e relativo. *Z-index* permite controlar o empilhamento de elementos que se sobrepõem.

O posicionamento dos elementos HTML é estático por padrão. Não são afetados pelas propriedades *top*, *bottom*, *left* e *right*.



Descrição da imagem: na imagem temos a visualização da propriedade *position*. O código está disponível em: <https://github.com/N-CPUninter/fundamentos-de-desenvolvimento-web/blob/main/html/position.html>. Acesso em: 20 dez. 2022.



Na imagem anterior, a primeira div foi definida com a posição estática. Mesmo se adicionarmos as propriedades *top*, *left*, *right* ou *bottom*, ela não será modificada.

A segunda div foi definida com a posição relativa, e será afetada pelas propriedades *top*, *left*, *right* ou *bottom*. No exemplo foi aplicado *left: 30px*;

A terceira div foi definida com a posição fixa e mesmo que a página tenha barra de rolagem ela permanecerá na mesma posição. No exemplo foi aplicado *bottom: 0*; e *right:0*;

Position com valor absoluto posiciona o elemento HTML em relação ao ancestral posicionado mais próximo (em vez de posicionado em relação à janela de visualização, como fixo).

5.2.2 Z-index

A propriedade *z-index* permite alterar a ordem de empilhamentos dos elementos, porém só funciona com elementos que possuem a propriedade *position*.

Exemplo:



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <style>
7     img{
8       position: absolute;
9       left:0px;
10      top:0px;
11      z-index: -1;
12    }
13  </style>
14 </head>
15 <body>
16   <h2>z-index</h2>
17
18   <p>Index da imagem = -1, está atrás do texto.</p>
19   
20 </body>
21 </html>
```

Crédito: Margarete Klamas.

Descrição da imagem: do lado esquerdo temos a ilustração de uma imagem de uma árvore com uma coruja, que foi posicionada atrás do texto. Do lado direito da tela temos o código CSS/HTML que foi utilizado. Foi aplicado um CSS interno na *tag* *img*, com a *position absolute* e *z-index: -1*. Faça testes comentando a propriedade *Z-index*, e alterado o seu valor para 1.

O código está disponível em: <https://github.com/n-cpuninter/fundamentos-de-desenvolvimento-web/blob/main/html/z_index.html>. Acesso em: 20 dez. 2022.



5.3 Overflow

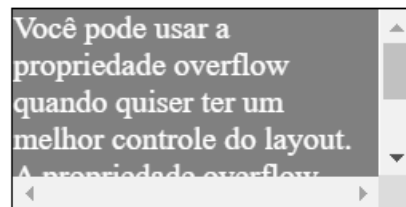
A propriedade **overflow** permite controlar o que acontece com o conteúdo grande demais para caber em uma área.

A propriedade tem os seguintes valores:

- **visible**: é o valor-padrão. O estouro não é contido. O conteúdo é renderizado fora da caixa do elemento;
- **hidden**: o estouro é contido e o restante do conteúdo ficará invisível;
- **scroll**: o estouro é contido e uma barra de rolagem é adicionada para ver o restante do conteúdo;
- **auto**: semelhante a **scroll**, mas adiciona barras de rolagem apenas quando necessárias.

Exemplo:

```
<style>
div {
  background-color: gray;
  width: 250px;
  height: 100px;
  border: 1px solid black;
  overflow: scroll;
  color: white;
}
</style>
```



Descrição da imagem: do lado direito há uma imagem que mostra a renderização de uma div com um conteúdo que excede a sua medida. Como a propriedade **overflow** está com o valor **scroll**, foi gerada barra de rolagem na div, tanto horizontal, quanto vertical.

FINALIZANDO

Agora você já é habilitado a estilizar (formatar) páginas HTML. Vimos as formatações básicas em CSS. Você vai perceber nos seus estudos que para deixar o *layout* como desejamos não é tão rápido, e exige paciência. É importante se desafiar e praticar *layouts* diferentes para entender como os CSS funcionam. Agora você pode e deve criar páginas HTML e aplicar CSS para praticar.

Bons estudos!



REFERÊNCIAS

ALVES, William Pereira. **HTML e CSS**: aprenda como construir páginas web. [recurso eletrônico] São Paulo: Expressa, 2021. (Minha Biblioteca).

CSS. Disponível em: <<https://www.w3schools.com/>>. Acesso em: 2 nov. 2022.

GUIA de Referência em CSS. Disponível em: <<https://www.w3c.br/divulgacao/guiasreferencia/css2/>>. Acesso em: 16 out. 2022.

SILVA, Maurício Samy. **Fundamentos de HTML5 e CSS3** . Novatec Editora, 2015.

SILVA, Maurício Samy. **CSS3**. Novatec Editora, 2013.

TERUEL, Evandro Carlos. **HTML5**: Guia Prático. 2. ed. rev. atual. e ampl. São Paulo: Érica, 2014. (Minha Biblioteca).