DevWeb

Capítulo 09

A beleza dos estilos

Eu costumo sempre dizer que você pode ter o melhor conteúdo do mundo, mas se ele não for bem apresentado, o alcance dele diminui consideravelmente. Visitantes de sites gostam da beleza, mesmo que eles não conheçam nada de design. É uma sensação satisfatória ver um conteúdo organizado e bonito. Esse capítulo vai te mostrar os primeiros passos com o uso de CSS, aplicando esses conceitos em seus códigos.

Você se lembra do que falamos sobre as CSS?

Nós já falamos sobre folhas de estilo em cascata, as famosas CSS no **capítulo 3**. Se por acaso você não se lembra direito, vale a pena voltar lá e dar uma segunda olhada nas definições. Vou considerar que você lembra claramente o que são as **Cascading Style Sheets** para podermos prosseguir.

Outro conteúdo muito importante que vimos está no **capítulo 8**, onde falamos sobre a íntima relação da HTML5 com a **semântica** das tags. Lá foi comentado que todo e qualquer efeito visual é responsabilidade das CSS. Vou partir daí e vamos trabalhar com os estilos, ok?

Caso você sinta qualquer dúvida a partir daqui, não se esqueça de revisitar os capítulos anteriores, pois a base foi dada gradativamente até esse momento. Vamos lá!

A forma mais simples de aplicar estilos: HTML Style

Vamos começar pela técnica mais básica para aplicar estilos em áreas pontuais em nosso site, que é usando as CSS dentro de parâmetros de HTML5. Crie mais uma pasta dentro da sua área de **exercícios** chamada **ex009** e crie um arquivo `index.html` com aquele código base que já fizemos várias vezes. Dentro da área `<body>`, crie um código como apresentado a seguir:

```
 8  <body>
 9      <h1>Capítulo 1</h1>
10      <h2>Capítulo 1.1</h2>
11      <p>Lorem, ipsum dolor sit amet consectetur adipisicing elit. Beatae
        assumenda eveniet odit accusantium distinctio saepe.</p>
12      <h2>Capítulo 1.2</h2>
13      <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit.
        Necessitatibus doloribus pariatur deserunt in nobis labore aliquam
        eos.</p>
14      <h1>Capítulo 2</h1>
15      <h2>Capítulo 2.1</h2>
16      <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Totam,
        nobis quas! Eum saepe temporibus!</p>
17  </body>
```

Agora abra o arquivo recém criado no Google Chrome. O resultado visual deve ser semelhante ao apresentado a seguir:

Capítulo 1

Capítulo 1.1

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Beatae assumenda eveniet odit accusantium distinctio saepe.

Capítulo 1.2

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Necessitatibus doloribus pariatur deserunt in nobis labore aliquam eos.

Capítulo 2

Capítulo 2.1

Lorem ipsum dolor sit amet consectetur adipisicing elit. Totam, nobis quas! Eum saepe temporibus!

RELEMBRANDO: Não se esqueça que você pode criar esses parágrafos automáticos com texto "Lorem ipsum" apenas digitando o atalho `lorem` no VSCode.

Vamos começar nos focando na tag `<body>` e aplicando um estilo diferente ao corpo da página. Adicione o parâmetro `style` e digite as duas declarações de `font-family` e `color`, conforme apresentado a seguir:

```
<body style="font-family: Arial, Helvetica, sans-serif; color: blue;">
```

Muito cuidado na hora de digitar esse código. Tudo deve ser seguido exatamente como fizemos acima, inclusive com letras maiúsculas e minúsculas. Não esqueça de adicionar os ponto e vírgulas para separar as declarações. Seu resultado visual deve ser esse:

Capítulo 1

Capítulo 1.1

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Beatae assumenda eveniet odit accusantium distinctio saepe.

Capítulo 1.2

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Necessitatibus doloribus pariatur deserunt in nobis labore aliquam eos.

Capítulo 2

Capítulo 2.1

Lorem ipsum dolor sit amet consectetur adipisicing elit. Totam, nobis quas! Eum saepe temporibus!

Note que o formato da letra mudou (era Times e ficou em Arial) e a cor da fonte também foi alterado para azul. Se por acaso alguma dessas duas alterações não funcionou corretamente com você, confira seu código, pois algo foi digitado incorretamente. Lembre-se que o computador não é tão inteligente quanto você pode pensar. Temos que dar ordens bem claras e seguindo sempre as regras para que ele nos obedeça.

CUIDADO! Se por acaso você aprendeu em algum momento a tag `<font color="blue">` e acha muito mais prática usá-la, saiba que ela **NÃO É MAIS ACEITA** para as especificações da HTML5!

Nós falamos sobre especificações obsoletas de HTML no **capítulo 3**. Se precisar relembrar, volte lá e faça uma revisão do conteúdo.

Vamos fazer mais uma alteração, dessa vez na linha do primeiro título `<h1>` do nosso código:

```
<h1 style="color: red;">Capítulo 1</h1>
```

O resultado visual deve ser:

Capítulo 1

Capítulo 1.1

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Beatae assumenda eveniet odit accusantium distinctio saepe.

Capítulo 1.2

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Necessitatibus doloribus pariatur deserunt in nobis labore aliquam eos.

Capítulo 2

Capítulo 2.1

Lorem ipsum dolor sit amet consectetur adipisicing elit. Totam, nobis quas! Eum saepe temporibus!

Note que apenas o **Capítulo 1** ficou vermelho, o **Capítulo 2** - que também é um `<h1>` - não teve alteração alguma. Isso acontece pois estamos fazendo **configurações pontuais** usando CSS.

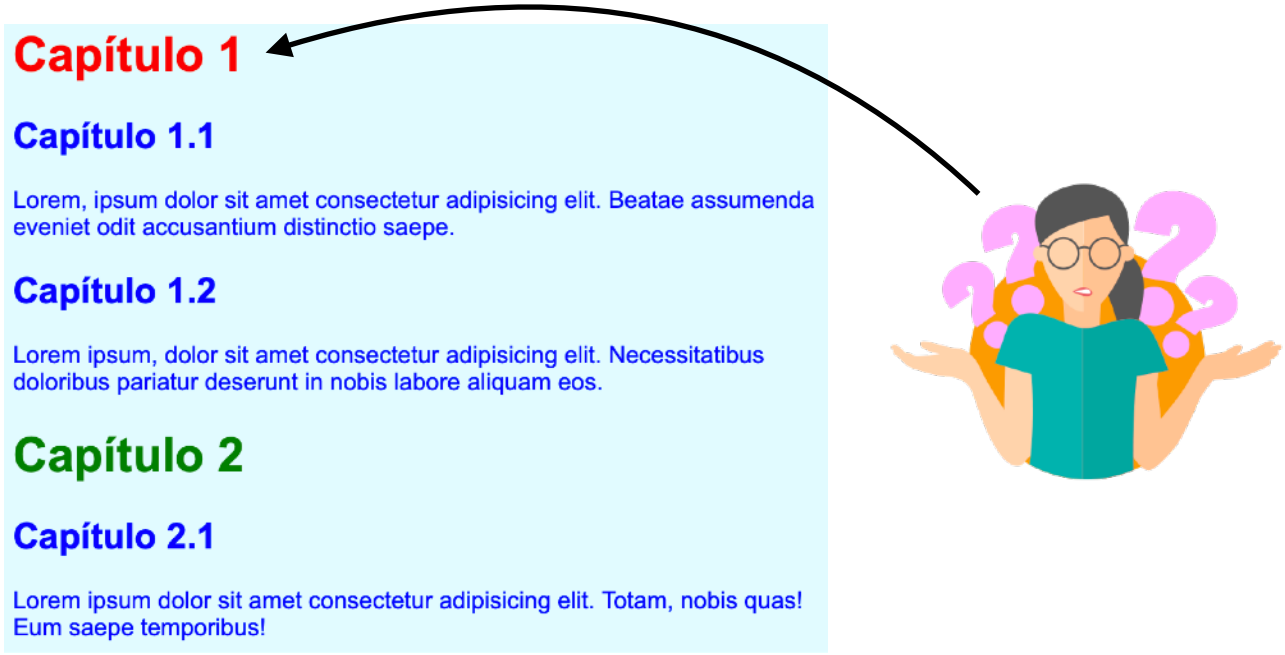
Estilizando de maneira mais interessante: CSS Local Style

Para aplicar estilos de forma mais dinâmica e prática, podemos adicionar uma tag `<style>` dentro da área `<head>` do nosso documento HTML local. Volte lá no seu VSCode, ainda no **ex009**, e adicione o código dentro de `<head>`.

```
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width,
        initial-scale=1.0">
6       <title>Estilos pontuais</title>
7       <style>
8           body {
9               font-family: Arial, Helvetica, sans-serif;
10              background-color: lightcyan;
11              color: blue;
12          }
13          h1 {
14              color: green;
15          }
16      </style>
17  </head>
```

ATENÇÃO! A tag `<style>` deve estar dentro da área <head> do seu documento HTML5. Se você colocá-la em qualquer outro local, como dentro da tag `<body>`, o resultado até pode funcionar, mas seu código estará fora dos padrões estabelecidos pela W3C. Siga sempre as regras!

Feitas as alterações, vamos ver o resultado e uma dúvida vai surgir:



Você sabe explicar por que o **Capítulo 1** ficou **vermelho** e não **verde**, como solicitamos?

Isso acontece porque as configurações pontuais (HTML style) vão prevalecer sobre as configurações gerais (CSS style). Volte ao seu código e remova todas as configurações de estilo que fizemos nas tags `<body>` e `<h1>` no início do capítulo.

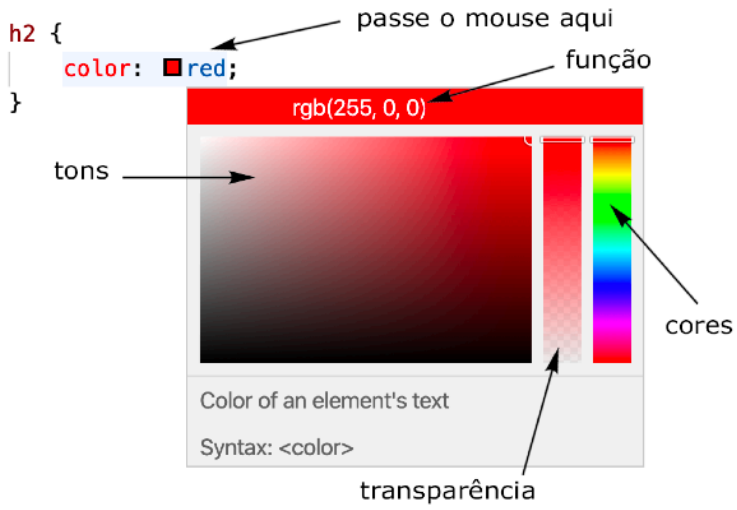
E já que falamos de cores...

Nos códigos CSS que vimos anteriormente, vimos declarações voltadas para cores. Até o momento, usamos valores textuais como `blue`, `red`, `lightcyan`, ...

No VSCode, ao criar uma propriedade relacionada a cores em CSS, podemos posicionar o cursor entre os dois pontos e o ponto e vírgula da declaração e pressionar `Ctrl+Espaço` para obter uma lista com os valores possíveis. Veja na imagem ao lado como esse recurso se comporta.

Porém, esse método de especificação de cores é muito limitado, pois uma tela moderna é capaz de exibir aproximadamente 65 milhões de cores.

Para conseguirmos mais possibilidades, devemos recorrer aos códigos hexadecimais ou então às funções CSS `rgb()`, `rgba()`, `hsl()` ou `hsla()`. Para usar esse recurso, adicione qualquer cor textual à sua propriedade e passe o mouse sobre o nome da cor (veja a imagem a seguir) e uma janela especial aparecerá.

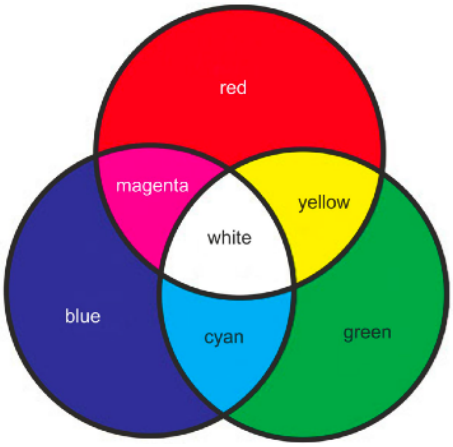


Na imagem ao lado, ao passar o mouse sobre a palavra `red`, a janela de seleção de cores vai aparecer, permitindo escolher a cor, tom e transparência. O sistema do VSCode vai sugerir a melhor função a ser utilizada, mas você pode mudá-la clicando sobre o nome da função, também indicado na imagem.

Faça testes, experimente, mude cores, use sua criatividade. A prática leva ao aprendizado sólido e duradouro!

Como representar uma cor?

Você já deve ter ouvido falar que as cores em uma tela são compostas da junção de três cores primárias: **vermelho** (red), **verde** (green) e **azul** (blue). Analisando a imagem ao lado, vemos que a junção de algumas cores primárias nos leva a a outras cores como o **magenta**, **amarelo** e **ciano**. Se usarmos todas as cores primárias no máximo, chegamos ao **branco**. Com todas as três no mínimo, obtemos o **preto**.



Cada cor primária pode ter um valor **entre 0 e 255**, totalizando **256 possibilidades** para cada elemento. Vejamos alguns exemplos de cores e seus respectivos códigos.

Vamos tomar como exemplo a cor **Teal** aqui ao lado. Seu código `rgb(0, 171, 169)` indica que existe quantidade **0** de vermelho nessa cor, **171** de verde e **169** de azul. No código de cores hexadecimal (iniciado sempre com #) indica que **00** é a quantidade de vermelho, **AB** é a quantidade de verde e **A9** é a quantidade de azul.

Esta mesma cor indicada acima, pode ser representada em CSS com um outro formato baseado na maneira como o olho humano enxerga as cores: o padrão **HSL**. A função `hsl(179, 100%, 34%)` indica que temos 179 de **hue** (matiz), 100% de **saturation** (saturação) e 34% de **lightness** (luminância).

Para obter versões de cores com transparência, basta arrastar a barra de transparência indicada à direita e perceber que mais um valor (**alpha**) será adicionado ao código.

Usando Gradientes em CSS

Podemos gerar gradientes e aplicarmos a componentes visuais usando folhas de estilo. Vamos usar um exemplo simples no nosso exercício atual. Vá até o documento e modifique a declaração do nosso seletor `body`.

```
<style>
    body {
        font-family: Arial, Helvetica, sans-serif;
        background-image: linear-gradient(90deg, yellow, red);
        color: black;
    }
```

Pode parecer esquisito no início, mas um gradiente é considerado pelo navegador como se fosse uma imagem, por isso usamos a propriedade `background-image` na declaração CSS. A função `linear-gradient` é auto-explicativa e gera um gradiente linear angular. O primeiro parâmetro da função, indica o ângulo de inclinação de 90 graus (*90deg*) e as seguintes indicam as cores do degradê a ser criado. Você pode indicar quantas cores quiser e o navegador vai saber se virar pra gerar seu degradê personalizado. Experimente na sua casa outros valores de ângulo também, incluindo negativos (45deg, -90deg, 25deg,...) e note as diferenças.

Também é possível gerar os chamados gradientes radiais, que também são meio auto-explicativos. Veja o exemplo:

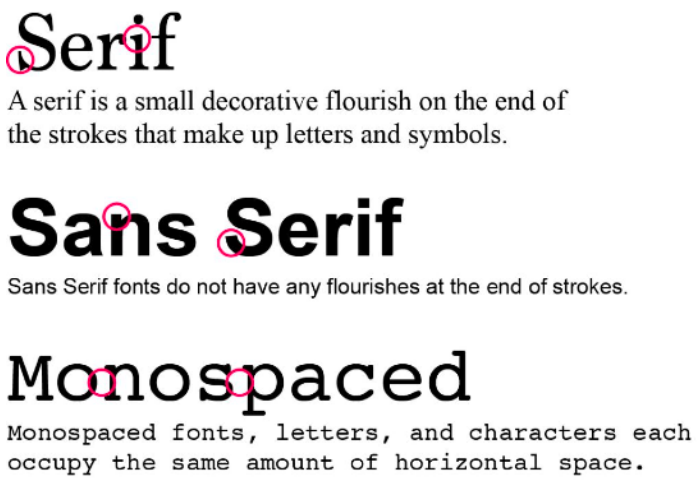
```
background-image: radial-gradient(circle, red, yellow, green);
```

Altere o tipo de gradiente do `body` para usar o formato radial circular e veja o resultado. Você também pode personalizar ainda mais seu degradê colocando uma porcentagem ao lado da cor como `red 10%, yellow 40%, green 50%`. Experimente!

Mais um pouco sobre fontes

🎶 Família, família. Papai, mamãe, titia 🎶

Olhe atentamente para a imagem ao lado. Ela mostra as diferenças entre as famílias genéricas de fontes. Fontes ditas **serifadas** apresentam pequenas decorações nas extremidades dos caracteres. Essas serifas servem basicamente para guiar nossa visão enquanto estamos lendo, mas podem atrapalhar no design e poluir visualmente um componente.



Para melhorar isso, foi criada a família genérica **sans serif** (sem serifa) que não apresenta essas decorações nas pontas que citamos anteriormente.

Uma outra família genérica é a **monoespaçada**, bastante usada por nós para apresentar comandos ou códigos. A diferença desse tipo de fonte é que cada letra ocupa exatamente o mesmo espaçamento horizontal, diferente das duas acima, onde a letra **S** ocupa muito mais espaço lateral do que a letra **i**, por exemplo.

Para configurar a família de uma fonte, usamos a propriedade `font-family` das CSS. Se indicarmos mais de uma fonte na sequência, estamos indicando ao navegador que dê preferência para a primeira. Caso ela não seja encontrada, tente a próxima. E essa estratégia se seguirá até a última, que geralmente é a família genérica `serif`, `sans-serif` ou `monospaced`.

Vamos fazer alguns exemplos aplicando famílias bem simples às nossas fontes. Vá até o seu exercício atual e aplique algumas declarações de `font-family` aos seletores de cada componente formatável do seu documento HTML.

```
<style>
    body {
        font-family: Arial, Helvetica, sans-serif;
        color: black;
    }
    h1 {
        font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
        color: rgb(24, 97, 126);
    }
    h2 {
        font-family: 'Times New Roman', Times, serif;
        color: rgb(33, 136, 161);
    }
    p {
        font-family: 'Courier New', Courier, monospace;
    }
</style>
```

No código acima, seus títulos principais `<h1>` usarão preferencialmente a fonte **Franklin Gothic Medium**, uma fonte sem serifa e que tem seu espaço horizontal bem limitado. Porém, essa fonte geralmente não existe em smartphones, que possuem a fonte Arial Narrow que é bem parecida mas é menos densa. Caso nenhuma delas seja encontrada no aparelho do visitante, o navegador vai selecionar a fonte Arial normal. Em último caso, se tudo der errado, o sistema selecionará uma fonte genérica sem serifa.

SEQUÊNCIAS SEGURAS: Existem as chamadas sequências seguras para especificações de famílias de fontes. Para ver quais são elas, abra o Google e faça uma rápida busca por `CSS Web Safe Font Combinations`.

Vamos falar de tamanhos

Além da família, podemos configurar tamanhos e estilos extras de qualquer componente textual do nosso documento HTML5.

Para especificar tamanho de fontes, existem várias medidas como **cm** (centímetros), **in** (polegadas), **pt** (pontos), **pc** (paicas), **px** (pixels), etc. Para tamanhos de fonte a serem exibidos na tela, o W3C recomenda o uso do **px** ou do **em**.

EU GOSTO DE USAR PT, MAS: A medida **pt** é aquela usada em editores de texto como o **Microsoft Word**. A recomendação oficial é de usar **pt** apenas para referenciar conteúdos que serão impressos.

A medida **em** é uma das que gera mais dúvida nos alunos. Ela é uma medida referencial em relação ao tamanho original da fonte. O tamanho padrão de uma fonte é geralmente **16px**, isso equivale a **1em**. A partir daí, podemos configurar o tamanho de um título, por exemplo, como sendo 2 vezes maior que a fonte padrão usando o valor **2em** para a propriedade.

```
h1 {
    font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
    font-size: 2em;
}
h2 {
    font-family: 'Times New Roman', Times, serif;
    font-size: 1.5em;
}
```

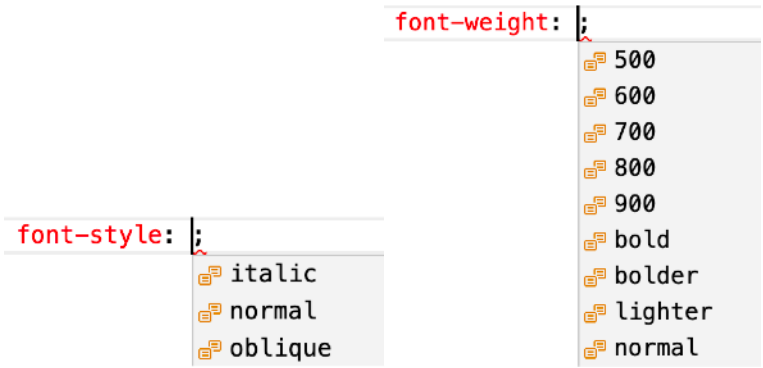
No exemplo acima, todo título `<h2>` do nosso documento será 1.5x o tamanho padrão da fonte de referência.

MAIS INFORMAÇÃO: Para saber mais sobre as medidas suportas pelas CSS, acesse o site oficial da W3C em:

https://www.w3.org/Style/Examples/007/units.pt_BR.html

Outros estilos

Existem outras formatações muito usadas em CSS, que são as propriedades `font-style` para aplicar o itálico e `font-weight` para aplicar o negrito, sem contudo existir o fator semântico discutido no **capítulo 08**.



O padrão para essas duas propriedades é o valor `normal`, mas podemos aplicar o valor itálico ao `font-style` usando `italic` (mais compatível) ou `oblique` (menos compatível). Já o negrito, pode ser aplicado por nomes como `lighter`, `bold` e `bolder` ou pelo peso numérico, como indicado na imagem.

Me dá uma mãozinha 🖐?

As formatações de fontes são tão importantes e tão usadas em CSS, que existem "atalhos" para usá-las. São as chamadas *shorthands*.

Existe uma shorthand para fontes que é a propriedade `font`. No lugar de fazer várias configurações em múltiplas linhas, podemos simplificar tudo de maneira muito simples.

Por exemplo, no lugar de configurar o estilo dos parágrafos do nosso site desse jeito:

```
p {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 1em;
    font-style: italic;
    font-weight: bold;
}
```

Podemos usar a shorthand `font` que vai simplificar tudo:

```
p {
    font: italic bold 1em Arial, Helvetica, sans-serif;
}
```

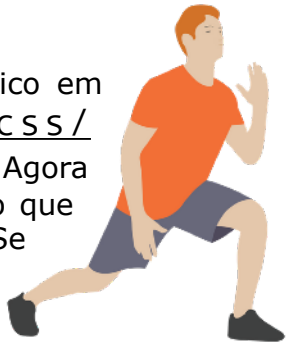
A ordem dos atributos de uma *shorthand* em CSS é importante. No caso da propriedade `font`, devemos informar, na ordem:

- `font-style`
- `font-variant`
- `font-weight`
- `font-size/line-height`
- `font-family`

Agora já sou especialista em CSS?

Ufa! Esse capítulo finalmente chegou ao fim. Mas não fique pensando que agora já sabe 100% das CSS. O caminho ainda é muito longo, nós só iniciamos! Nos próximos capítulos, vamos voltar um pouco à linguagem HTML, pois ainda temos um pouco mais a aprender.

Hora de exercitar



Chegou a hora de acessar o endereço do nosso repositório público em https://gustavoguanabara.github.io/html-css/exercicios/ e executar o **exercício 009** no seu computador. Agora tente atingir esse mesmo resultado em casa, sem copiar o código que eu criei. Nesse momento, a prática é algo que você mais precisa. Se por acaso ficar difícil, pode acessar o repositório público de HTML e CSS e dar uma olhada nos comandos, mas **EVITE COPIAR**.

Eu já falei sobre isso no YouTube?

Eu sei que às vezes as pessoas gostam mais de assistir vídeos do que ler livros, e é por isso que eu lanço há anos materiais no canal Curso em Vídeo no YouTube. O link que vou compartilhar contigo tem o conteúdo explicado como você leu aqui, só que de forma mais ilustrada. Reserve um tempo dos seus estudos para assistir esse vídeo todo.

Curso em Vídeo: https://www.youtube.com/playlist?list=PLHz_AreHm4dlAnJ_jJtV29RFxnPHDuk9o