

INTRODUÇÃO ÀS CSS



INTRODUÇÃO ÀS CSS

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 tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.

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: CSS inline 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** 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
12     assumenda eveniet odit accusantium distinctio saepe.</p>
13   <h2>Capítulo 1.2</h2>
14   <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit.
15     Necessitatibus doloribus pariatur deserunt in nobis labore aliquam
16     eos.</p>
17   <h1>Capítulo 2</h1>
18   <h2>Capítulo 2.1</h2>
19   <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Totam,
20     nobis quas! Eum saepe temporibus!</p>
21 </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

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

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

Capítulo 1.2

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

Capítulo 2

Capítulo 2.1

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

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 `` 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!

A screenshot of a web browser window. The main content area has a solid purple background. On the left, the text "Soluções digitais para negócios" is displayed in large white font. On the right, there is a QR code. At the bottom center, there is a logo for "hostnet" featuring a stylized cloud icon and the word "hostnet" in white. The browser's title bar and control buttons are visible at the top.

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 internal 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, e adicione o código dentro de `<head>`.

```
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width,
6          initial-scale=1.0">
7      <title>Estilos pontuais</title>
8      <style>
9          body {
10              font-family: Arial, Helvetica, sans-serif;
11              background-color: #lightcyan;
12              color: #blue;
13          }
14          h1 {
15              color: #green;
16      }
17  </style>
18 </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!

Cursos grátis de tecnologia
que te preparam para o
mercado de trabalho

RECODE





QUERO MAIS CORES: Se você está achando que essa coisa de colocar cor pelo nome em Inglês é algo limitado, você está coberto de razão! No próximo capítulo, vou te mostrar outras técnicas de representar cores em CSS. Aquarede e confie!

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

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!

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.

A técnica mais versátil: CSS external style

Manter as folhas de estilo fora do código HTML, além de uma maior organização faz com que tudo seja reaproveitado de maneira mais eficiente nas outras páginas do nosso site. Para isso, utilizamos a tag <link> especialmente configurada para trabalhar com arquivos externos de estilo. Essa tag deve ser colocada dentro da área <head> do seu documento HTML.

```

<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CSS external</title>
    <link rel="stylesheet" href="style.css">
</head>

```

A linha com o `<link>` pode estar em qualquer linha, contanto que seja dentro da área `<head>`. Particularmente, sempre procuro adicionar essa configuração após a tag `<title>` do documento atual.

Dica para criar CSS externo com VSCode

O Visual Studio Code sempre trás algumas facilidades para o nosso dia-a-dia. Vá até o final da linha com o `<title>` e pressione Enter para criar uma nova linha. Depois comece digitando a palavra `link`, sem as marcas de tag (veja a imagem ao lado).

No menu de contexto que vai aparecer, escolha a opção `link:css` e a linha apresentada abaixo será magicamente preenchida.

```

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content=
    <title>CSS external</title>
    link
</head>
    ↴ link
    ↴ link:atom
    ↴ link:css
    ↴ link:favicon
    ↴ link:im
    ↴ link:import

```

`<title>CSS external</title>` Siga o link (cmd + clique)
`<link rel="stylesheet" href="style.css">`

Agora passe o mouse sobre o nome do arquivo `style.css` e veja que existe um atalho "*Siga o link*", bastando pressionar Ctrl+clique (ou Cmd+clique, se estiver usando o sistema MacOS). A primeira vez que você segurar o Ctrl/Cmd e clicar sobre o link, o VSCode vai perguntar se você quer que ele crie o arquivo pra você. Clique em Sim ou Ok para aceitar a ajuda e seu arquivo será criado automaticamente.

Agora é só adicionar os seletores e todas as suas respectivas declarações nesse arquivo separado para que elas possam ser aplicadas ao documento que contiver um `<link>` para ele.



```
1 @charset "UTF-8";
2 /* Regra @charset caso haja problemas com acentuação */
3
4 body {
5     background-color: #lightgoldenrodyellow;
6     color: #chocolate;
7     font-size: 12pt;
8     font-family: Arial;
9 }
10
11 h1 {
12     color: #brown;
13 }
14
15 a {
16     color: #brown;
17     text-decoration: none;
18 }
```

Na **linha 1**, colocamos uma **regra** em CSS, que vai indicar a compatibilidade de codificação com o padrão UTF-8, assim como fizemos com o arquivo HTML5. Essa linha não é obrigatória e normalmente nem vai aparecer na maioria dos seus arquivos de configurações de estilo, mas caso você comece a ter problemas de compatibilidade com alguns caracteres, saiba que ela existe.

Na **linha 2**, adicionamos um **comentário** para facilitar a documentação do arquivo. Os comentários - assim como vimos em HTML - só servirão de explicação para que o desenvolvedor entenda o funcionamento de uma determinada linha ou trecho de código. O navegador não vai considerar nada que está entre os símbolos /* e */ em CSS.

Nas demais linhas, fizemos as configurações dos seletores, da mesma maneira que criamos com as outras duas técnicas apresentadas no capítulo.



TÁ CONFUSO? Se você não está entendendo claramente todas as declarações, não se preocupe! Nos próximos capítulos nós vamos nos aprofundar nelas. Foque agora nas técnicas de uso das CSS.

Qual técnica eu escolho pra usar?

Nesse capítulo, aprendemos as três técnicas de uso de folhas de estilo em cascata: **inline**, **interna** e **externa**. Mas em que situações devemos escolher cada um dos formatos?

Para falar com toda a sinceridade, a técnica **CSS inline style** deve ser evitada ao máximo. Ela acaba deixando seu código meio confuso, misturando a parte HTML e CSS em uma mesma linha. Mas se é para citar um momento específico em que podemos aplicar estilos inline em nossos códigos, use apenas em configurações muito pontuais e que não serão mais usadas em nenhum outro momento.



Já a técnica **CSS internal style** organiza melhor seu código, separando conteúdo e estilo em duas áreas bem definidas do seu documento. Use essa técnica quando for criar páginas isoladas com estilos próprios, que não serão replicados em outras páginas. Opte também por essa técnica apenas se a quantidade de configurações de estilo for pequena/média. Usar muitos seletores com muitas declarações vai fazer com que seu arquivo `.html` fique muito grande e seu conteúdo seja visualmente jogado lá pra baixo, dificultando manutenções futuras.

Por fim, opte sempre pela técnica **CSS external style** sempre que seu estilo for usado em várias páginas dentro do seu site. Usando a tag `<link>` em várias páginas, você pode compartilhar o mesmo estilo entre elas e não vai precisar ficar alterando vários arquivos quando o seu cliente solicitar uma pequena mudança no tom de uma determinada cor, por exemplo.

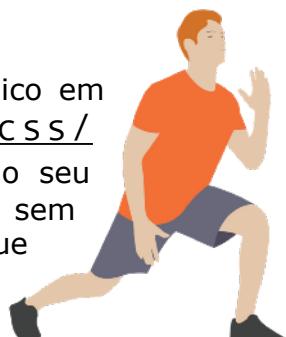
De forma resumida, guarde isso na sua cabeça:

CSS externo = use sempre que puder
CSS interno = use para pequenas configurações
CSS inline = procure evitar

Ainda é possível misturar as três técnicas, criando um CSS externo para as configurações globais, CSS interno para as configurações locais de um documento e CSS inline para pequenas configurações pontuais.

Hora de exercitar

Chegou a hora de acessar o endereço do nosso repositório público em <https://gustavoguanabara.github.io/html-css/exercicios/> e executar os **exercícios 013, 014 e 015** 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**.



Quer acompanhar tudo em vídeo?

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 faz parte da playlist completa onde você encontra os **Módulos 1 e 2** do **Curso de HTML5 e CSS3**, completamente gravado com base nesse material.



Módulo 1 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dkZ9-atkcmcBaMZdmLHft8n

Módulo 2 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



1. A sigla CSS significa:

- Content Style Setup
- Content Smart Sheets
- Cascading Style Sheets
- Cascading Setup Style

2. Ao configurar os estilos com a técnica inline, as propriedades CSS são colocadas em que local?

- Dentro da própria tag HTML, em um parâmetro chamado style
- Dentro da área <head> do documento HTML, na área delimitada por <style>
- Em um arquivo separado dentro da própria pasta onde o documento HTML está
- Dentro de uma pasta especial chamada /style que fica no diretório atual do site

3. O que aconteceu com configurações como <body bgcolor="blue"> e que eram usadas em sites mais antigos?

- Continuam funcionando perfeitamente e podem ser usadas normalmente
- Continuam funcionando, mas não é nada recomendável pois as CSS são mais atuais
- Não funcionam mais de forma alguma
- Ainda são a única opção de mudar cores

4. Qual das opções a seguir é a única que configura corretamente a cor da letra de um título usando CSS inline?

- A <h1 color="blue">Título</h1>
- B <h1 style="blue">Título</h1>
- C <h1 style="font-color: blue">Título</h1>
- D <h1 style="color: blue">Título</h1>

5. Para usar a técnica de estilos internos, a tag <style> deve estar em que local?

- A logo depois da tag HTML que queremos configurar
- B dentro da área <head> do seu documento HTML
- C no final do arquivo, logo depois do </body>
- D em qualquer lugar do arquivo HTML

6. Em um mesmo documento HTML, utilizamos as técnicas de CSS interno e CSS inline para mudar a cor de um elemento e acabamos usando duas cores diferentes. Quais configurações vão prevalecer na versão final do site?

- A as configurações duplicadas feitas nas CSS inline vão prevalecer
- B as configurações duplicadas feitas nas CSS internas vão prevalecer
- C as duas configurações vão prevalecer e o texto será exibido duplicado com cores diferentes
- D o seu navegador vai decidir qual cor utilizar. Não existe uma regra para esses casos.

7. Para utilizar estilos CSS externos, devemos usar a tag _____ no documento HTML para fazer o carregamento:

- A <style>
- B <css>
- C <external>
- D <link>

8. A tag de carga de estilos externos deve estar localizada em que local do documento HTML?

- A deve ser a primeira linha do arquivo HTML
- B deve estar dentro da área <body> do documento
- C deve estar dentro da área <head> do documento
- D pode estar em qualquer lugar, desde que esteja dentro do documento HTML

9. Para que os arquivos externos em CSS tenham compatibilidade com caracteres acentuados, devemos adicionar a regra:

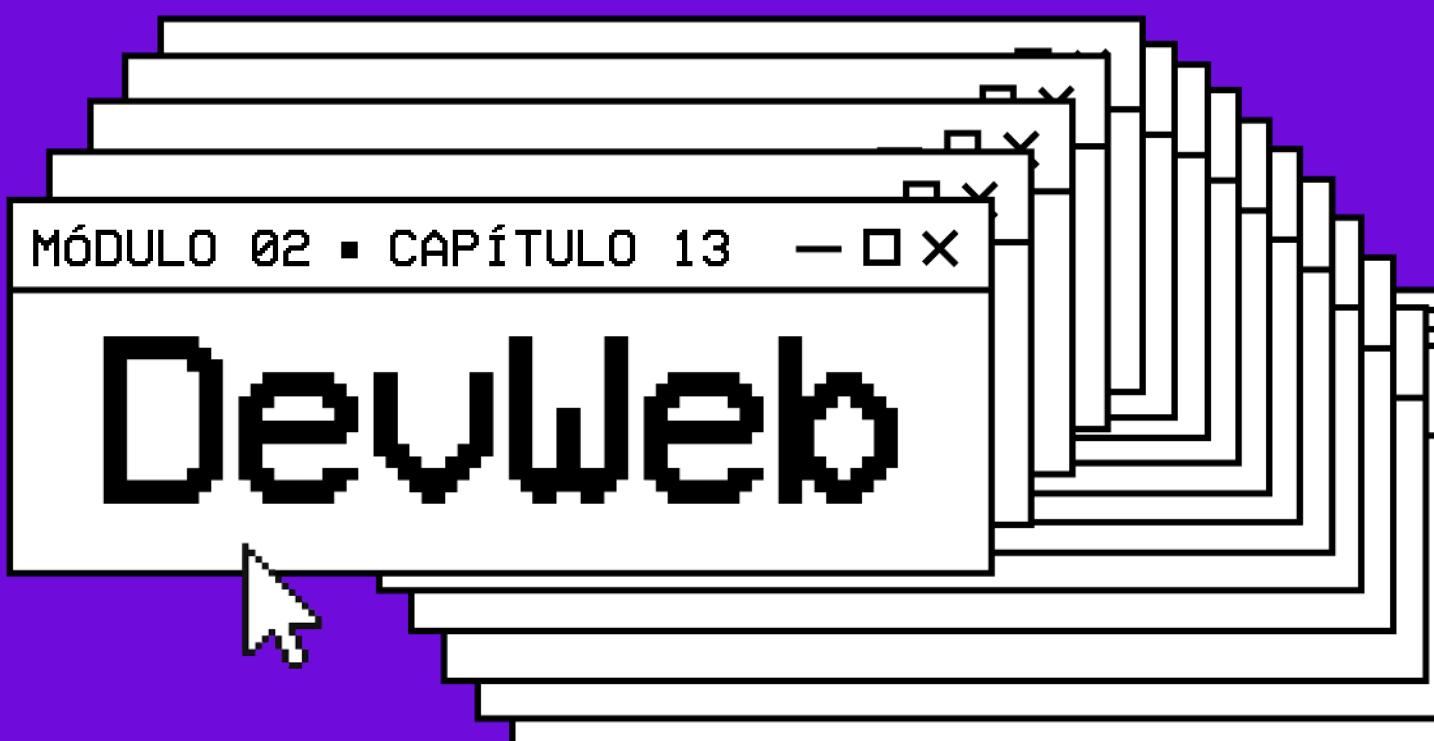
- A @UTF-8;
- B @charset "UTF-8";
- C @charset "PT-BR";
- D @PT-BR;

10. No fim das contas, vamos dar preferência à técnica de _____ sempre que for possível e vamos tentar evitar a técnica de _____ em nossos sites. A opção que apresenta os termos que completam as lacunas na ordem correta é:

- A CSS externo / CSS interno
- B CSS inline / CSS interno
- C CSS externo / CSS inline
- D CSS interno / CSS externo

Suas anotações

Não guarde conhecimento. Ele é livre. Compartilhe o seu e veja ele se espalhando pelo mundo 



O PODER DAS CORES



M02C13

O PODER DAS CORES

Boa parte da apresentação de um determinado conteúdo parte da escolha das cores e fontes de uma página. Quando sabemos escolher bem uma paleta de cores que harmonize com o nosso conteúdo, já conseguimos dar o primeiro passo no caminho de um site bonito. Nesse capítulo, vou te mostrar as principais dicas para você começar a aplicar estilo aos seus projetos.

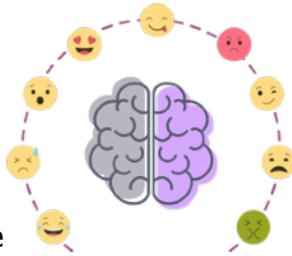


Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.

— □ ×

A emoção das cores

Jamais subestime o poder das cores, elas podem influenciar na quantidade de tempo que seu visitante passa visitando o seu site e pode até mesmo ser um poderoso critério de decisão para uma compra.



Segundo um dos grandes especialistas na área de otimização de conteúdos **Neil Patel** (guarde bem esse nome, você provavelmente vai voltar a ouvir sobre ele) em seu artigo "*Como cores afetam conversões*" afirma que as pessoas levam cerca de 90 segundos para decidir se querem ou não um produto, e que 90% dessa decisão se baseia na sua cor.

As cores passam emoção para o subconsciente das pessoas, mesmo que na maioria dos casos isso não seja feito de forma totalmente consciente. Percebemos as cores e sentimos a sua emoção mesmo sem ter a plena certeza de que alguém usou a **psicologia das cores** para modelar um site ou produto.

Se você fizer uma breve busca pelo termo "*psicologia das cores*", vai ver várias sugestões de emoções para determinada cor. O **azul**, por exemplo, acaba nos remetendo a *harmonia, equilíbrio, confiança, profissionalismo, integridade e segurança*. Agora dê uma breve olhada nos logotipos do Facebook, Twitter, LinkedIn, Dell, HP e Intel. O que elas têm em comum? Será que isso é só coincidência?

Peguei o azul como exemplo principal, pois ela é citada como a cor favorita entre homens (46%) e mulheres (44%) e também é a cor com a menor taxa de rejeição (entre 1% e 2%).



APRENDA MAIS. A seguir, vou colocar alguns links para artigos onde você vai poder ver mais sobre a emoção das cores. Consuma esses conteúdos para entender melhor o poder das cores.

- <https://rockcontent.com/blog/psicologia-das-cores/>
- <http://www.matildefilmes.com.br/psicologia-das-cores-guia-avancado-para-profissionais/>
- <https://neilpatel.com/br/blog/psicologia-das-cores-como-usar-cores-para-aumentar-sua-taxa-de-conversao/>

Mas **muita atenção** ao seguir guias de cores e artigos, pois eles não devem ser considerados como uma verdade absoluta para todos os mercados e situações. Constantemente vemos casos de marcas que adotam uma determinada paleta de cores totalmente não recomendada por esses padrões e acabam fazendo muito sucesso. Meu sincero conselho: considere as recomendações, mas não se prenda a elas. Com isso na mente, acompanhe algumas sugestões de aplicação de algumas das cores mais usadas em sites.



Cor	Associada a	Usar em	Evitar
vermelho	amor, emoção, energia, raiva, perigo	comida, moda, entretenimento, serviços de emergência e saúde	luxo, natureza, serviços em geral
amarelo	felicidade, alegria, otimismo, covardia	dar luz, dar calma e felicidade, chamar atenção	pode indicar que algo é barato ou spam
laranja	divertimento, ambição, calor, cautela	comércio eletrônico, entretenimento, call-to-action	pode se tornar cansativo se muito explorado
verde	saúde, natureza, dinheiro, sorte, inveja	relaxamento, turismo, financeiros, meio ambiente	luxo, tecnologia, meninas adolescentes
azul	competência, sabedoria, calma, frio	tecnologia, medicina, ciências, governo	comida (reduz apetite)
roxo	criatividade, poder, sabedoria, mistério	produtos de beleza, astrologia, ioga, espiritualidade, adolescente	não prende muito a atenção, indiferente
marrom	terra, robustez, estabilidade, amizade	alimentação, imobiliária, animais, finanças	cor considerada conservadora
preto	elegância, autoridade, mistério, morte	luxo, moda, marketing, cosméticos	desconforto e medo
branco	pureza, limpeza, felicidade, segurança	medicina, saúde, tecnologia, luxo (com preto, ouro, cinza)	não chama atenção, deve ser combinado
cinza	formalidade, sofisticação, frieza, indiferença	bens de luxo, efeito calmante	dá a sensação de frieza
rosa	amor, romance, sinceridade, cuidados	produtos femininos e cosméticos	pode tornar muito sentimental e doce

— □ ×

Cursos grátis de tecnologia que te preparam para o mercado de trabalho



RECODE

Achei bonito, mas não sei explicar o motivo

Você provavelmente já olhou para um belo site ou para uma peça de propaganda bem produzida, teve aquela sensação de que tudo está em perfeita harmonia, mas não sabe explicar o porquê do seu cérebro perceber toda essa beleza e te fazer se sentir bem.

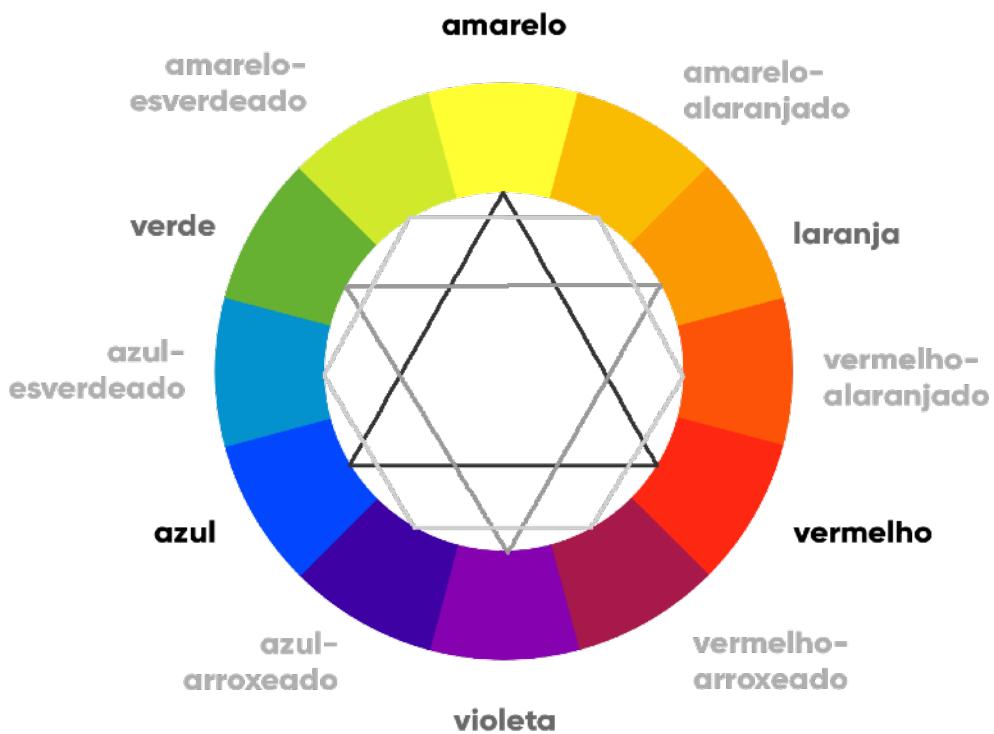
Pois saiba que boa parte de toda essa percepção que temos é por conta das cores e da simetria geométrica que aconteceu durante o planejamento desse site/propaganda. Um designer vai decidir qual será a **paleta de cores** usada e fazer tudo fazer sentido quando as pessoas olharem para o resultado. Por exemplo, olhe para as cores apresentadas em cada linha ao lado. Elas são cores que fazem sentido quando usadas em conjunto. Você é capaz de perceber que elas possuem uma certa "harmonia" e talvez não saiba que existe toda uma ciência por trás disso. E é exatamente sobre isso que começaremos a falar daqui pra frente. Preciso antes te apresentar um amigo meu: o círculo cromático.



O círculo cromático

Dentro da teoria das cores, precisamos separá-las em grupos para que possamos decidir se as escolhas que vamos fazer para o nosso site vão fazer um sentido harmônico e para que os nossos visitantes olhem para o nosso projeto e instinctivamente pense: "- nossa, que bonito!".

A base para isso é conhecer o círculo cromático e compreender as suas sub-divisões. E ele está logo aí abaixo, olhe atentamente, se possível para uma versão colorida. Se por acaso você está vendo uma versão impressa em preto-e-branco, acesse agora o meu repositório e veja o PDF diretamente na tela do seu computador ou celular. Vai ficar tudo mais claro pra você, pode acreditar!



Analisando atentamente o círculo cromático, percebemos as três **cores primárias**, que estão destacadas com o texto mais escuro: **amarelo, vermelho e azul**.

Da junção das cores primárias, temos as três **cores secundárias**, que são o **laranja** (amarelo+vermelho), o **violeta/roxo** (azul+vermelho) e o **verde** (azul+amarelo).

Da junção de uma cor primária com uma secundária, temos as seis **cores terciárias**:

- **Amarelo-esverdeado** (amarelo+verde)
- **Amarelo-alaranjado** (amarelo+laranja)
- **Vermelho-alaranjado** (vermelho+laranja)
- **Vermelho-arroxeadoo** (vermelho+roxo)
- **Azul-arroxeadoo** (azul+roxo)
- **Azul-esverdeado** (azul+verde)

— □ ×

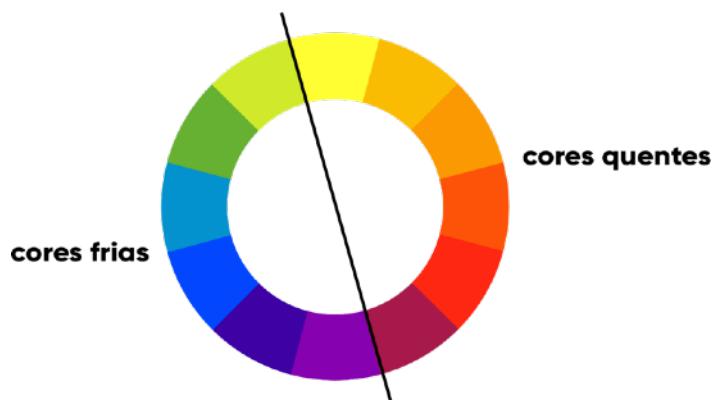
Soluções digitais para negócios





Temperatura e Harmonia

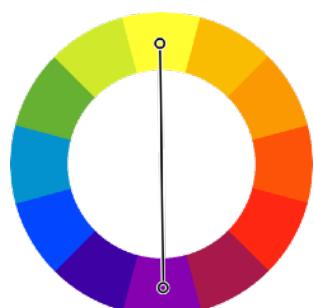
Olhando o círculo cromático, também conseguimos classificar as cores por sua temperatura. Dá só uma olhada na imagem a seguir:



As cores quentes, criam uma sensação de calor e proximidade. Já as cores frias, estão associadas a sensações mais calmas, de frescor e tranquilidade.

Além da classificação por temperatura, podemos classificar as cores por esquemas harmônicos.

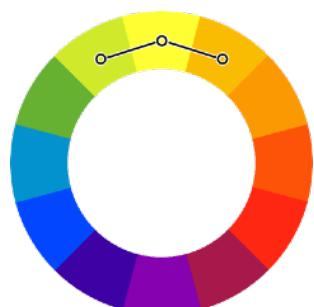
Cores complementares



São aquelas que apresentam o maior contraste entre si. Elas estão localizadas do lado imediatamente oposto do círculo cromático.

Se pegarmos qualquer cor primária, a sua cor complementar é sempre uma cor secundária. De forma similar, qualquer cor terciária tem uma outra cor terciária como complementar. Quando juntamos duas cores complementares, sempre obtemos o cinza.

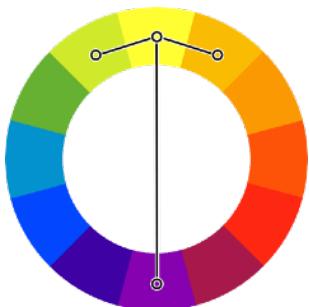
Cores análogas



Diferente das cores complementares, que estão do lado extremo oposto no círculo cromático, as cores análogas são aquelas que são imediatamente vizinhas entre si.

Por serem cores consecutivas, as cores análogas possuem um baixo contraste entre elas, mas criam uma bela harmonia quando combinadas em um mesmo design.

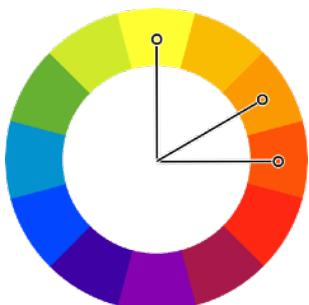
Cores análogas mais uma complementar



Dá pra notar que essa aqui é uma combinação dos dois tipos anteriores, não é?

Essa técnica quebra um pouco o ritmo semelhante das cores análogas, adicionando uma cor que cria um grande contraste com as três análogas.

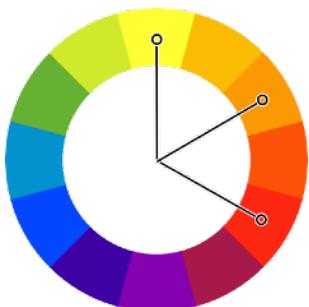
Cores análogas relacionadas



Nesse tipo de harmonia, escolhemos duas cores análogas (consecutivas) e depois pulamos uma terceira cor (em qualquer direção) e escolhemos a quarta.

Com essa técnica, conseguimos um resultado parecido com o das cores análogas simples, mas com um pouco mais de contraste sem ter que escolher uma cor complementar.

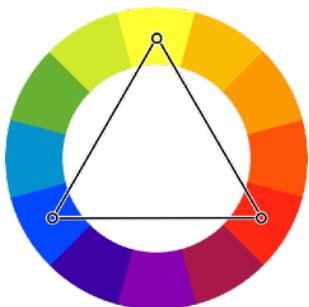
Cores intercaladas



Um tipo menos usado de harmonia, já que às vezes não funciona tão bem assim. Vamos escolher a primeira cor e depois mais duas com intervalo constante entre elas.

Na imagem ao lado, criei um exemplo onde o intervalo é constante entre as cores selecionados.

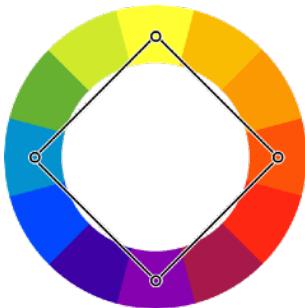
Cores triádicas



Técnica bastante utilizada e que garante uma grande riqueza de cores, onde escolhemos três pontos equidistantes no círculo cromático.

Esse esquema gera sempre um triângulo equilátero e cria uma opção que sempre possui um ótimo contraste entre as cores.

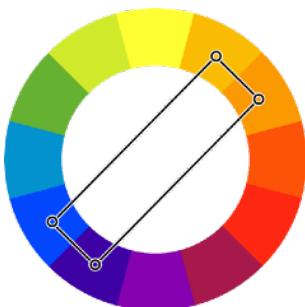
Cores em quadrado



Bastante semelhante ao esquema triádico, mas permite selecionar quatro cores com um contraste razoável entre as cores escolhidas.

Esse esquema gera sempre um quadrado perfeito com os pontos selecionados.

Cores tetrádicas



Com essa técnica, vamos escolher dois pares de cores complementares, que não serão necessariamente análogas ou consecutivas. Isso vai nos garantir dois pares de cores, com bastante contraste entre si.

Monocromia



Uma harmonia bem diferente das anteriores, que usa apenas uma cor e varia apenas a sua saturação e o seu brilho. Essa combinação geralmente gera pouquíssimo contraste entre as cores escolhidas, mas acaba gerando um resultado visual bem agradável aos olhos, conhecido como “degradê”.

— □ ×

Cursos que vão te levar ao próximo nível

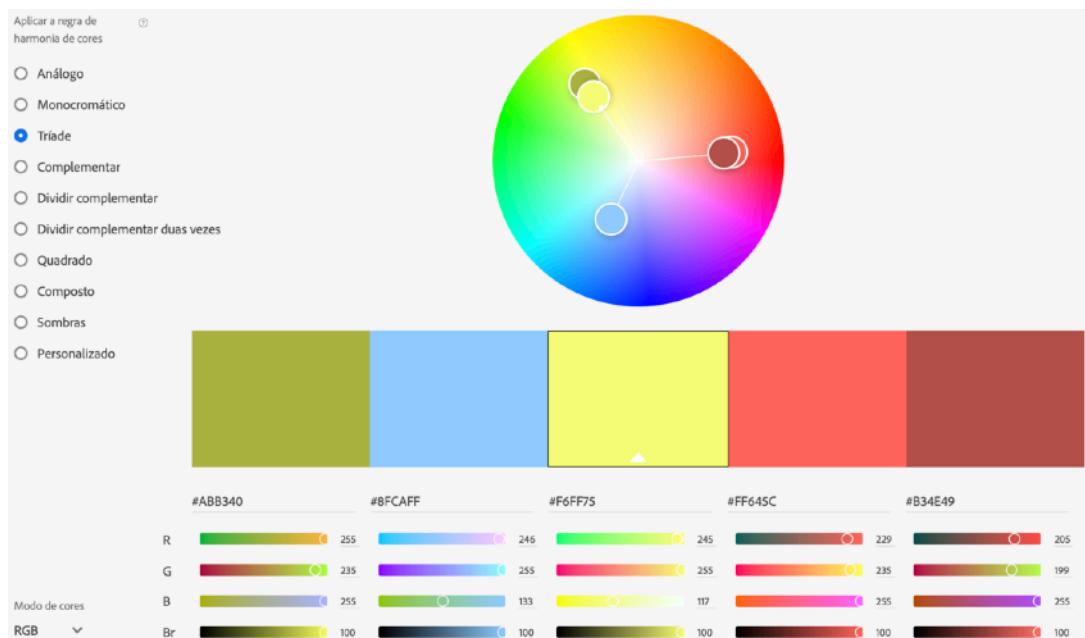
 estudonauta



E onde eu aplico esse conhecimento na prática?

Existem vários sites e serviços que vão te ajudar na escolha da paleta de cores do seu site. A que vai permitir mais opções, na minha opinião é o **Adobe Color** (disponível em <https://color.adobe.com/pt/>), que tem recursos gratuitos para te auxiliar na escolha das suas cores baseado nos esquemas de harmonia que vimos anteriormente.

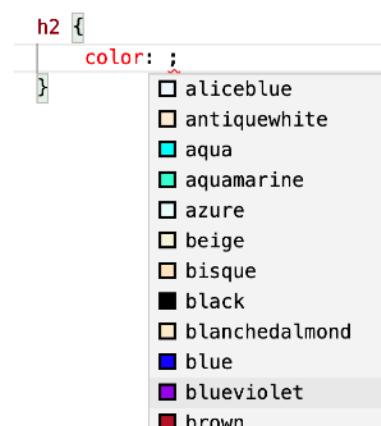
No modo **Criar**, você vai escolher o modo de cores (para monitores é o RGB) e também a regra de harmonia que você quer usar. A partir daí ele vai te sugerir uma paleta com cinco cores perfeitamente harmônicas. Para mudar as tonalidades sem mudar a regra, arraste qualquer uma das cores e a regra vai se aplicar aos outros pontos. Já no modo **Explorar**, você vai ser apresentado a várias paletas prontas e vai poder copiá-las na maior cara de pau, pois tudo é grátil e liberado!



Aplicando cores ao nosso site

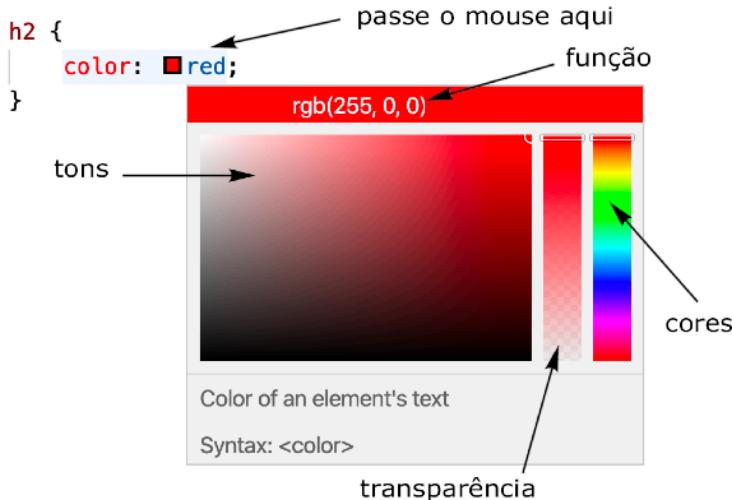
Nos códigos CSS do capítulo anterior, vimos declarações voltadas para cores. Até o momento, usamos valores textuais como `blue`, `red`, `lightcyan`, e muitas outras.

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.

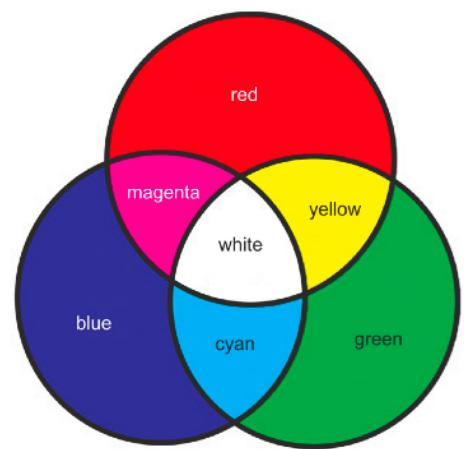
Na imagem ao lado, ao passar o mouse sobre a palavra `red`, a janela de seleção de cores vai aparecer, permitindo escolher a cor, tom e transparência. O sistema do VSCode vai sugerir a melhor função a ser utilizada, mas você pode mudá-la clicando sobre o nome da função, também indicado na imagem.

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 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** na tabela da página a seguir. 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

Lime #A4C400 RGB(164, 196, 0)	Green #60A917 RGB(96, 169, 23)	Emerald #008A00 RGB(0, 138, 0)	Teal #00ABA9 RGB(0, 171, 169)
Cyan #1BA1E2 RGB(27, 161, 226)	Cobalt #0050EF RGB(0, 80, 239)	Indigo #6A00FF RGB(106, 0, 255)	Violet #AA00FF RGB(170, 0, 255)
Pink #F472D0 RGB(244, 114, 208)	Magenta #D80073 RGB(216, 0, 115)	Crimson #A20025 RGB(162, 0, 37)	Red #E51400 RGB(229, 20, 0)
Orange #FA6800 RGB(250, 104, 0)	Amber #F0A30A RGB(240, 163, 10)	Yellow #E3C800 RGB(227, 200, 0)	Brown #825A2C RGB(130, 90, 44)
Olive #6D8764 RGB(109, 135, 100)	Steel #647687 RGB(100, 118, 135)	Mauve #76608A RGB(118, 96, 138)	Taupe #87794E RGB(135, 121, 78)

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;
  }
</style>
```

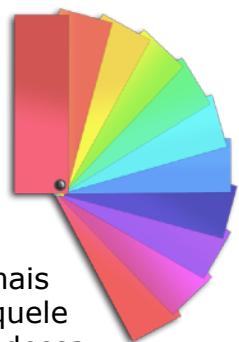
Pode parecer esquisito no início, mas um gradiente é considerado pelo navegador como se fosse uma imagem, por isso usamos a propriedade `background-image` na declaração CSS. A função `linear-gradient` é auto-explicativa e gera um gradiente linear angular. O primeiro parâmetro da função, indica o ângulo 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!

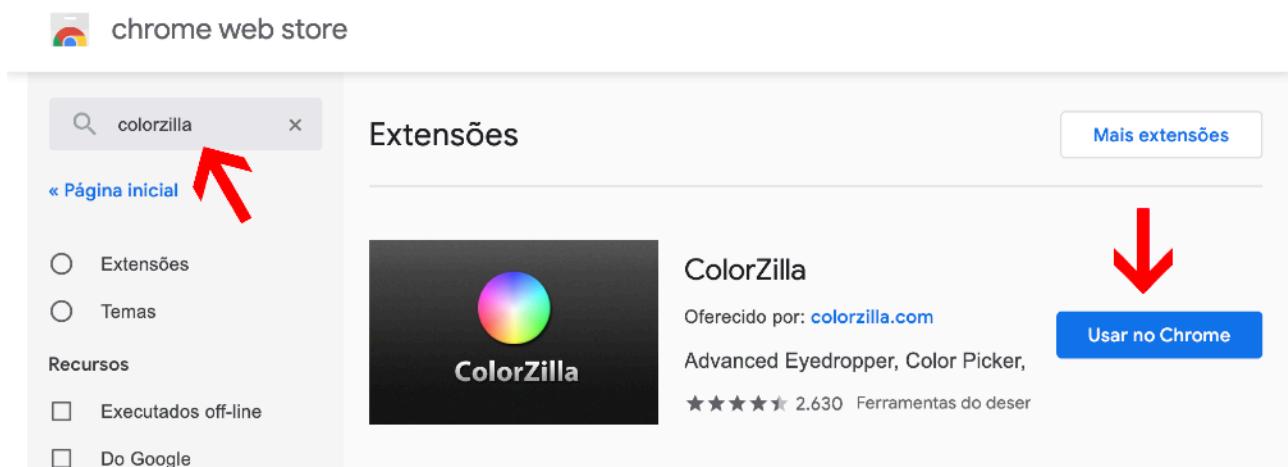
Encontrei uma cor maravilhosa! Qual é o código dela?



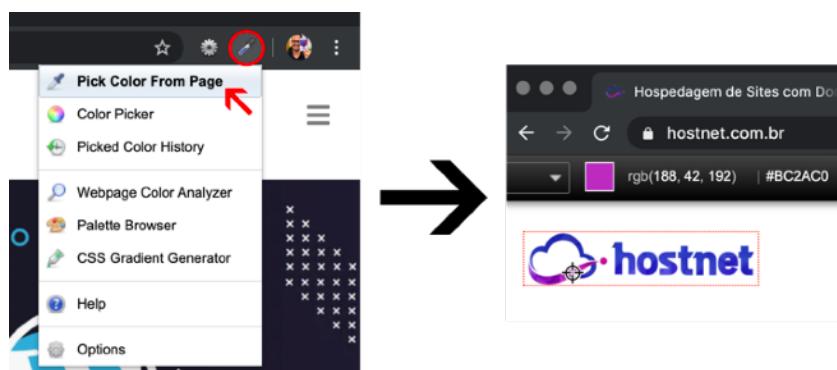
Com certeza essa situação vai aparecer na sua vida, mais cedo ou mais tarde. Você vai entrar em um site e vai descobrir um tom perfeito daquele amarelo que estava procurando. Como descobrir exatamente o código dessa cor?

Uma das maneiras bem práticas de executar essa tarefa é usando uma extensão gratuita do **Google Chrome** chamada **Colorzilla**.

Para instalar uma extensão, abra o **Google Chrome** e acesse a **Chrome Web Store** no endereço <https://chrome.google.com/webstore/>. Na caixa de pesquisa no canto superior esquerdo, digite **colorzilla** e pressione Enter. A extensão vai aparecer no lado direito da tela e você deve clicar sobre o botão “Usar no Chrome” e clicar no botão autorizando usar a extensão (veja os passos na imagem a seguir).

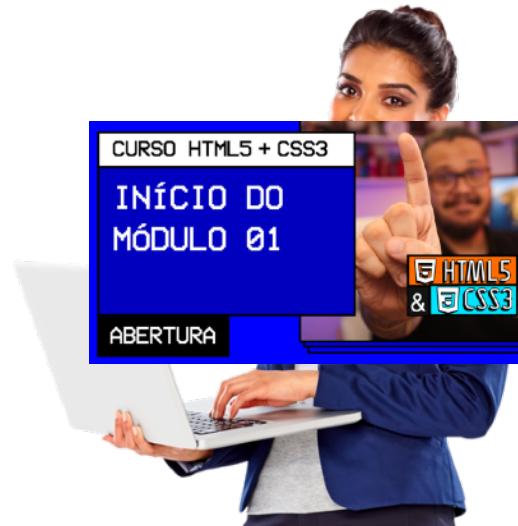


Agora você vai perceber que ao lado da barra de endereço do navegador, apareceu um pequeno conta-gotas. Abra um site qualquer e clique sobre esse ícone. Em seguida, clique em “*Pick Color From Page*” e aí é só clicar no local que deseja capturar. A cor vai aparecer em formato `rgb()` e com seu código hexadecimal. Ao clicar, o código será copiado para a sua área de transferência.



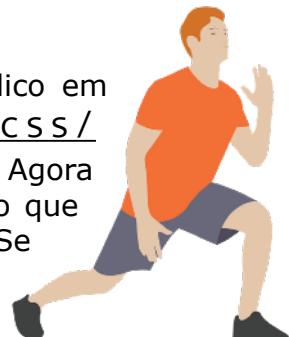
Agora já sou especialista em CSS?

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 continuar estudando alguns conceitos de design, nos focando especialmente nos conceitos sobre fontes.



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 016** 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**.



Quer acompanhar tudo em vídeo?

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 faz parte da playlist completa onde você encontra os **Módulos 1 e 2** do **Curso de HTML5 e CSS3**, completamente gravado com base nesse material.



Módulo 1 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dkZ9-atkcmcBaMZdmLHft8n

Módulo 2 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4d1UpEXkY1AyVLQGcpSgVF8s

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



1. Segundo levantamentos relacionados à psicologia das cores, qual é a cor com a maior taxa de aceitação e menor rejeição que existe?

- A vermelho
- B verde
- C azul
- D amarelo

2. De acordo com a tabela apresentada neste capítulo 13, qual é a cor que está associada a criatividade, poder, sabedoria e mistério?

- A preto
- B roxo
- C rosa
- D laranja

3. Qual cor deve ser evitada em sites de alimentação ou relacionados com comida, pois pode induzir a uma redução no apetite?

- A roxo
- B azul
- C vermelho
- D marrom

4. Ao construir um site, devemos definir qual será _____ utilizado(a), pois isso vai criar a sensação de que tudo faz sentido visualmente e o usuário vai ter a sensação de harmonia, mesmo sem saber do que se trata.

- A o círculo cromático
- B a cor análoga
- C a cor tetrádica
- D a paleta de cores

5. Qual das cores a seguir é a única que não está presente explicitamente no círculo cromático?

- A amarelo-alaranjado
- B vermelho-arroxeados
- C amarelo-arroxeados
- D azul-esverdeado

6. Qual dos itens a seguir é o único que não é considerado como uma cor quente?

- A roxo
- B vermelho
- C laranja
- D amarelo

7. Enquanto as cores _____ são aquelas que estão no extremo oposto do círculo cromático e por isso possuem o maior contraste entre si, as cores _____ são aquelas que estão localizadas imediatamente aos lados da cor considerada (vizinhas).

- A análogas / complementares
- B complementares / análogas
- C análogas / intercaladas
- D complementares / intercaladas

8. Qual dos itens abaixo é o único que não é uma função CSS para a representação de cores?

- A rgbl()
- B rgba()
- C hsl()
- D hsla()

9. Considerando a representação de cor hsl(179, 100%, 34%), os três valores indicados são respectivamente quantidades para:

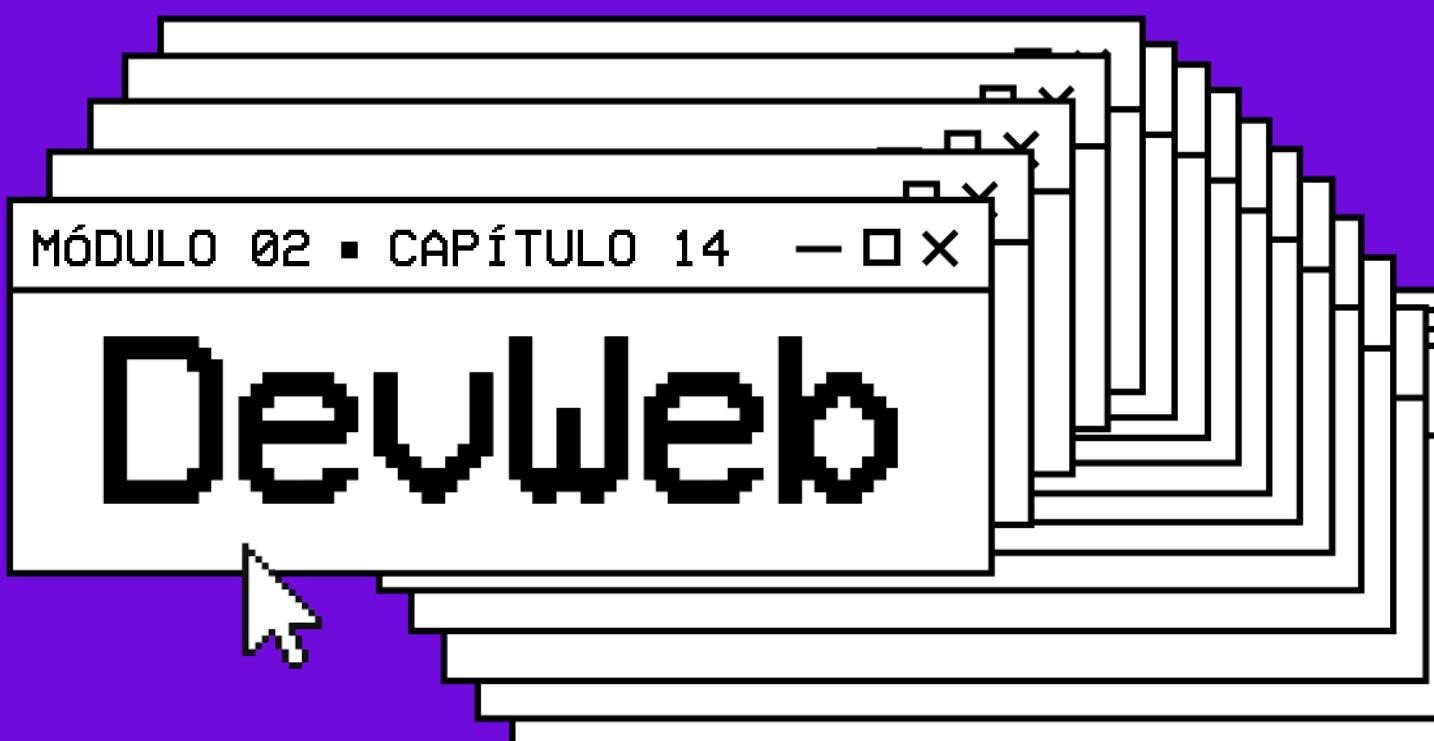
- A matiz, saturação e luminância
- B harmonia, saturação e luminosidade
- C hegemonia, salientação e lealdade
- D hierarquia, síntese e liberdade

10. Para criar um efeito degradê em CSS, podemos usar as seguintes funções:

- A linear-degradee e radial-degradee
- B derradee-linear e degradee-radial
- C gradient-linear e gradient-radial
- D linear-gradient e radial-gradient

Suas anotações

Não guarde conhecimento. Ele é livre. Compartilhe o seu e veja ele se espalhando pelo mundo



VAMOS FALAR
DAS FONTES



M02C14

VAMOS FALAR DAS FONTES

Sem dúvidas as cores são muito poderosas, como pudemos conferir no capítulo anterior. Mas em conjunto com elas, temos as fontes, que são um ótimo recurso visual para criar a identidade da página e mostrar a ideia que queremos passar com o nosso design. Vamos aprender um pouco mais sobre fontes e como aplicá-las aos nossos sites. Venha comigo.



— □ ×

Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.

Tipografia? Que bicho é esse?

Você se lembra de que falamos no capítulo anterior que as cores podem transmitir emoções? Pois as fontes também possuem essa mesma capacidade. E se você é uma pessoa atenta, vai entender que quando somamos essas emoções, podemos ter resultados ainda mais fortes.

Para entender mais sobre as fontes, precisamos estudar os fundamentos básicos da **tipografia**, que é uma arte antiga que estuda técnicas de **escrita** (do Grego, *graphía*) para a apresentação de forma impressa (do Grego, *týpos*). Essa preocupação surgiu na época em que as grandes prensas físicas eram usadas para produzir livros/jornais. Os **tipos móveis** são aquelas peças de metal/madeira/argila (ao lado) que são usados para "carimbar" o papel e fazer as letras.



E o mundo da tipografia se inicia em 1450, com o inventor Alemão **Johannes Gutenberg** (foto ao lado), criador da prensa mecânica de tipos móveis. Na verdade, os Chineses foram os primeiros a criarem o conceito de prensa com tipos móveis, mas Gutenberg acabou sendo reconhecido como aquele que deu início à **Revolução da Imprensa**. Antes disso tudo, cada exemplar de um livro era reproduzido através de material manuscrito devidamente copiado, palavra por palavra, até atingir o resultado desejado.

Fonte, letra e família

Glifos, letras, caracteres

São os signos alfabéticos projetados para reprodução mecânica. O exemplo a seguir representa os glifos de **a** até **h**.

abcdefghijklmnopqrstuvwxyz

Família tipográfica

É o conjunto de glifos que possuem as mesmas características anatômicas, independente das suas variações.

Vou exemplificar esse conceito com o exemplo a seguir: a família tipográfica **Open Sans** possui várias configurações de peso (de 300 a 800). Mesmo parecendo representações bem diferentes, todos eles fazem parte da mesma família tipográfica.

Light 300

Curso em Vídeo

Regular 400

Curso em Vídeo

Semi-bold 600

Curso em Vídeo

Bold 700

Curso em Vídeo

Extra-bold 800

Curso em Vídeo

Fontes

As fontes são conjuntos de glifos que formam uma família tipográfica. O termo fonte também é aplicável ao arquivo digital que armazena todos os formatos de glifos que compõem uma determinada família tipográfica.

A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77
A	B	C	D	E	F	G	H	I	J	K	L	M
N 78	O 79	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109
A	B	C	D	E	F	G	H	I	J	K	L	M
n 110	o 111	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0 48	1 66	2 67	3 68	4 69	5 70	6 71	7 72	8 73	9 74			
0	1	2	3	4	5	6	7	8	9			
\$ 36	* 42	+ 43	- 45	/ 47	& 38	~ 71	= 61	% 37	" 34	' 39	Ø 157	# 35
\$	*	+	-	/	&	~	=	%	"	'	ÿ	#
@ 64	_ 66	(40) 41	, 44	. 46	; 59	: 58	? 63	! 33	\ 92	124	{ 123
(@	_	()	,	.	;	:	?	!	\		{
} 125	< 60	> 62	[91] 93	^ 96	~ 94	® 166	° 167	½ 171	¼ 172	µ 230	Ø 157
}	<	>	[]	^	~	TM	∫	Ω	º	μ	ÿ
À 183	Á 181	Â 182	Ã 199	à 133	á 160	â 131	ã 198	É 144	Ê 210	Í 214	í 161	Ó 224
ô 226	ö 229	ó 162	ô 147	õ 228	ú 233	û 154	ñ 165	ñ 164	ç 128	ç 135		
'	÷	1	^	~	€	>	"	Û	«	Ê		

As fontes falam?

No capítulo anterior nós falamos sobre a importância de escolher uma boa paleta de cores para o nosso projeto. Também precisamos saber escolher as famílias tipográficas que utilizaremos em um site. As fontes também podem passar emoções. Vamos a um exemplo?

Imagine que eu tenha que representar a palavra “amor” através de uma determinada tipografia. Qual das opções abaixo você escolheria?

Amor
Amor

Com toda certeza, a maioria das pessoas escolheria a segunda opção. Sabe o por quê? A palavra “amor” tem mais a ver com uma representação mais suave e fluida, não algo mais robusto e forte. E mesmo sem te dizer isso, provavelmente a escolha do tipo fez isso por mim.

O problema é que a escolha não é tão simples assim. Na representação acima, com uma palavra curta e isolada, conseguimos ler facilmente a palavra “amor” em qualquer uma das opções de fontes escolhidas. Chamamos isso de **legibilidade**. Mas basta colocarmos uma frase maior para as coisas ficarem um pouco confusas.

*“Amor quando é amor não definha. É
até o final das eras há de aumentar.
Mas se o que eu digo for erro, e o meu
engano for provado, então eu nunca terei
escrito ou nunca ninguém terá amado.”*
William Shakespeare

Cursos que vão te levar ao próximo nível

 estudonauta



No exemplo anterior, mesmo que todas as palavras tenham uma **legibilidade** razoável (até dá pra entender), a **leiturabilidade** não é tão boa assim. Essa segunda característica diz respeito à fluidez que conseguimos ter na leitura.

Sendo assim, escolher um bom tipo é essencial para cada caso. Não povoar nosso site com tipos diferentes também é uma ótima ideia. No máximo dois ou três tipos já estaria ótimo.

Vamos conhecer agora algumas características anatômicas dos tipos para nos ajudar a escolher boas fontes para nosso site.

Anatomia do Tipo

Vamos analisar cada uma das partes de um tipo. Volte sempre para essa página ao ler a descrição de cada elemento:



A - **Altura das maiúsculas**: Altura que as letras maiúsculas vão ocupar. Geralmente um pouco menor que a soma da ascendente com a mediana ($< B + D$)

B - **Ascendente**: Parte das letras maiúsculas que se ergue acima da linha mediana

C - **Descendente**: Parte das letras minúsculas que passa por baixo da linha de base.

D - **Altura-X**: Também chamada de mediana, define o tamanho das letras minúsculas. Tem esse nome, pois se baseia no tamanho da letra x minúscula.

E - **Corpo**: É a soma de quatro medidas: ascendente + altura-x + descendente + espaço de reserva. É o tamanho total da letra. É o valor que escolhemos ao configurar o tamanho da fonte em um texto.

1 - **Arco**: presente em letras minúsculas. Uma linha curva que nasce em na haste principal.

2 - **Barriga**: curva em uma letra maiúscula ou minúscula, fechada, ligada à haste vertical em dois pontos.

3 - **Braço**: traço horizontal ou inclinado, ligado à haste vertical principal de uma letra maiúscula ou minúscula.

4 - **Cauda**: apêndice do corpo de algumas letras (g, j, J, K, Q, R), que fica abaixo da linha base.

5 - **Enlace**: a forma como uma haste, linha ou filete se liga a um arremate, a uma serifa ou a um terminal. Pode ser angular ou curvilíneo.

6 - **Espinha**: curva e contracurva estrutural da letra S.

7 - **Esporão**: uma projeção que encontramos nas letras b e G.

8 - **Filete**: haste horizontal ou inclinada, fechada nas duas extremidades, por duas hastas ou por uma curva.

9 - **Haste**: traço principal de uma letra, geralmente vertical.

10 - **Olho**: espaço em branco, fechado, dentro de uma letra.

11 - **Orelha**: apêndice presente na letra **g**, que pode ser em gota, botão, bandeira ou gancho.

12 - **Pé**: terminal ou serifa horizontal que arremata uma perna na parte de baixo.

13 - **Perna**: haste vertical ou inclinada com um extremidade livre (ou com um pé) e outra extremidade ligada ao corpo da letra.

14 - **Serifa**: também chamada de apoio ou patilha. Pequenas retas que ornamentam as hastas de alguns tipos.

15 - **Terminal**: forma que arremata a extremidade de uma linha curva de uma letra.

16 - **Vértice**: também chamada de ápice. Formada pela convergência de duas hastas que se encontram. Pode ser pontiagudo, oblíquo, plano ou redondo.



Categorias de fontes

Os tipos ou fontes tipográficas também são classificados por suas categorias. Elas são baseadas principalmente na presença ou ausência da serifa, o item 14 da lista anterior. As demais categorias, acabam derivando das duas principais (com e sem serifa) ou não se encaixam nessas características e por isso geram novas categorias.

Fontes Serifadas

Esta é a categoria mais clássica de fontes, surgida lá na época das prensas que eu citei no início do capítulo. Tipicamente, os caracteres serifados sempre foram aplicados em grandes blocos de textos impressos em papel e se aproveitam de uma característica da nossa percepção: nós nunca lemos as palavras letra por letra, e sim por um conjunto. As serifas têm a capacidade de guiar nossos olhos graças aos pequenos prolongamentos que elas criam e fazem as letras “se juntarem” em palavras. A seguir, vemos quatro exemplos de fontes serifadas:

Soluções digitais
para negócios

hostnet

A QR code is located in the bottom right corner of the purple advertisement.

Bree Serif
TypeTogether

Noto Serif
Google

Curso em Vídeo Curso em Vídeo

EB Garamond
Georg Duffner

Bitter
Huerta Tipográfica

Curso em Vídeo Curso em Vídeo

Atualmente, não usamos fontes serifadas para apresentar textos longos na Web pois as tendências atuais nos levam a usar fontes um pouco mais leves visualmente. Porém, as fontes serifadas são bastante usadas em títulos, pois acabam chamando mais atenção por conta das características que citei.



APRENDA MAIS SOBRE FONTES: As classificações não param por aqui. Existem também sub-categorias para cada fonte serifada, como as *old style*, *transitional*, *didone*, *slab*, *clarendon* e *glyphic*, que apresentam características detalhadas para cada uma. Nesse material, nós vamos nos limitar apenas às classificações gerais por motivos práticos. Provavelmente o seu professor de *design* vai falar sobre o assunto de maneira mais aprofundada.

Fontes não Serifadas

Mais conhecidas por seu “nome chique” em Francês *sans-serif* (significa “sem serifa”), são fontes que, como você já pode imaginar, não apresentam serifas. As primeiras fontes dessa categoria surgiram em 1816, mas foram consideradas demais para a época. Anos depois, ressurgiram em versão melhorada e vieram pra ficar, principalmente para a Web. Isso acontece porque elas são ótimas para a exibição em telas/monitores pois transmitem a sensação de limpeza, clareza e organização. Veja a seguir alguns exemplos de fontes não serifadas:

Open Sans
Steve Matteson

Oswald
Vernon Adams, Kalapi Gajjar, Cyreal

Curso em Vídeo

Titillium Web
Multiple Designers

Curso em Vídeo

Montserrat
Julieta Ulanovsky, Sol Matas, Juan Pablo del Peral,

Curso em Vídeo

Curso em Vídeo

A grande maioria dos textos que você está lendo nesse material desde o início do curso estão sendo escritos com uma fonte não serifada muito popular: a **Verdiana**.

Fontes Monoespacadas

Essa é uma das categorias de fontes que vieram derivadas das duas categorias que vimos anteriormente, por isso existem fontes monoespacadas com e sem serifas. A principal diferença desse tipo de fonte é o espaço horizontal (largura) ocupado por cada letra. Na maioria das fontes, a letra **i** ocupa muito menos espaço lateral do que a letra **M**, não é? Não para as fontes monoespacadas. Elas possuem a mesma largura para todas as letras.

Source Code Pro
Paul D. Hunt

IBM Plex Mono
Mike Abbink, Bold Monday

Curso em Vídeo Curso em Vídeo

Roboto Mono
Christian Robertson

Ubuntu Mono
Dalton Maag

Curso em Vídeo Curso em Vídeo

A principal vantagem no uso desse tipo de fonte é facilitar ao máximo a leitura das palavras, principalmente aquelas que requerem que você as reproduza. Usamos muito esse tipo de fonte para representar comandos de linguagens de programação de computadores. Por isso, nós também costumamos chamá-las de fonte de terminal ou fonte de console.

Fontes Script

Também chamadas de fontes *handwriting*, são aquelas que tentam imitar a escrita humana. Seu uso deve ser bem controlado e jamais será aplicado a textos muito longos, pois causam cansaço visual e tornam-se difíceis de ler, como já provamos anteriormente no início do capítulo, dentro do item “As fontes falam”.

Great Vibes
TypeSETit

Pacifico
Vernon Adams, Jacques Le Bailly, Botjo Nikoltchev

Curso em Vídeo

Curso em Vídeo

Dancing Script
Impallari Type

Reenie Beanie
James Grieshaber

Curso em Vídeo

Curso em Vídeo

Fontes Display

Toda fonte que foge completamente das definições feitas pelas classificações acima são consideradas fontes *display*. São fontes com bastante efeitos visuais, enfeitadas e até mesmo curiosas. Também são chamadas de fontes comemorativas e algumas delas sequer representam letras, podendo ser desenhos de animais, objetos, pessoas, personagens de quadrinhos, etc.

Lobster

Impallari Type, Cyreal

Bungee Outline

David Jonathan Ross

Curso em Vídeo

CURSO EM VÍDEO

Luckiest Guy

Astigmatic

Press Start 2P

CodeMan38

CURSO em VÍDEO

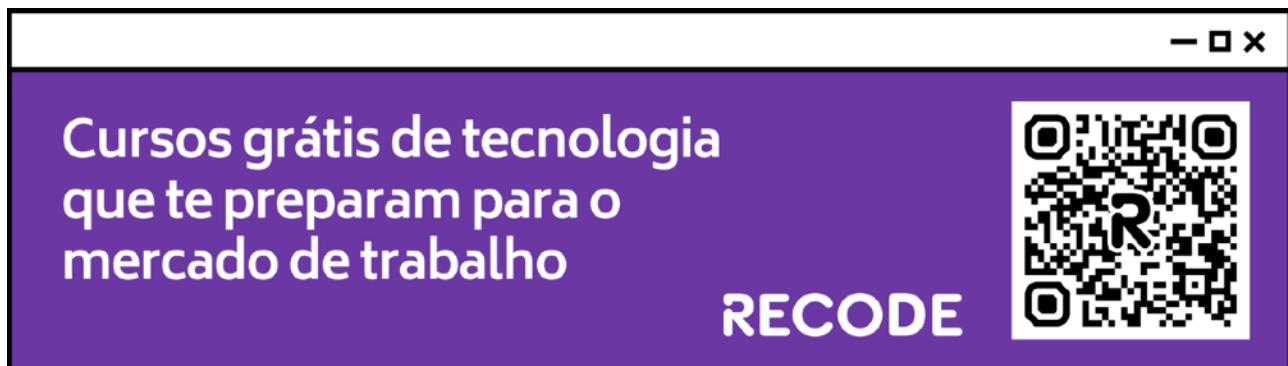
Curso em Vídeo

Essas fontes também são recomendadas para criar títulos em destaque e devem ser evitadas para textos médios ou longos.

Como aplicar isso na prática?

Para configurar a família tipográfica que será aplicada a um determinado texto, usamos a propriedade `font-family` das CSS. Se indicarmos mais de uma família 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>
```



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.

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.

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 suportadas 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`

Alinhamentos

Existem quatro tipos de alinhamento de textos:

`text-align: left;`

 Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

`text-align: right;`

 Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

```
text-align: center;  
  
Lorem Ipsum is simply dummy text of  
the printing and typesetting industry.  
Lorem Ipsum has been the industry's  
standard dummy text ever since the  
1500s, when an unknown printer took a  
galley of type and scrambled it to make  
a type specimen book.
```

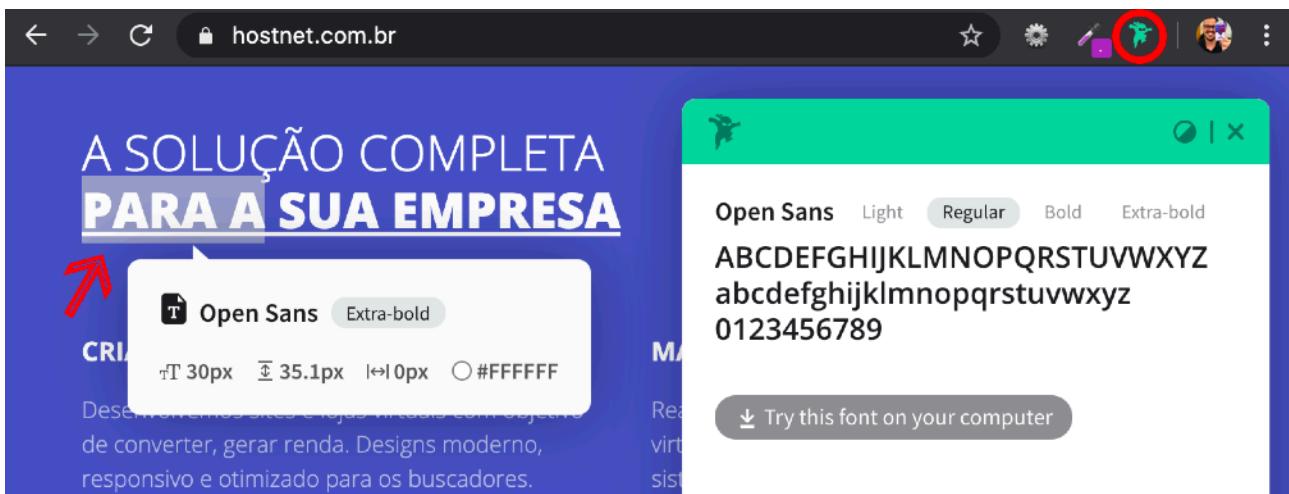
```
text-align: justify;  
  
Lorem Ipsum is simply dummy text of  
the printing and typesetting industry.  
Lorem Ipsum has been the industry's  
standard dummy text ever since the  
1500s, when an unknown printer took a  
galley of type and scrambled it to make  
a type specimen book.
```

Como descobrir uma fonte que está sendo usada em outro site?

No capítulo anterior, te ensinei a usar a extensão **Colorzilla** para pegar uma cor que estava sendo usada em outro site. Agora vou te ensinar a usar a extensão **Fonts Ninja** do Google Chrome para capturar a fonte usada em componentes de texto.

Acesse novamente o site do **Chrome Web Store** e procure pela extensão **Fonts Ninja** (ensinei como fazer isso no capítulo anterior). Uma vez instalada e ativa, a extensão ficará ao lado da barra de endereços, assim como o Colorzilla.

Abra um site qualquer, selecione o trecho de texto que quer identificar (recomendo selecionar poucas palavras) e clique sobre o botão do **Fonts Ninja** (veja na imagem a seguir).



Além de mostrar qual foi a família tipográfica utilizada no texto selecionado, a extensão vai te indicar o tamanho e peso da fonte, o espaçamento vertical e horizontal e a cor aplicada a ele. Para isso, basta mover o mouse sobre o texto e um balão aparecerá com todas essas informações.

Como usar fontes do Google Fonts

Além das famílias tipográficas e fontes padronizadas disponíveis para os navegadores, podemos usar fontes externas em nosso projeto sem a necessidade de baixar e instalar nenhuma fonte no computador do visitante.

Para isso, usaremos um serviço gratuito chamado **Google Fonts**, disponível em <https://fonts.google.com>. Ao acessar o site, algumas áreas são muito úteis:

The screenshot shows the Google Fonts homepage. At the top, there is a search bar (1) with a magnifying glass icon, a 'Custom' dropdown (2), a 'Curso em Vídeo' example text area (2), a '40px' size selector (3), and a 'Categories' dropdown (4). Below these are buttons for 'Language', 'Font properties', and 'Show only variable'. A checkbox for 'Show only variable' is also present. The main area displays 989 families of fonts. A font family named 'Roboto' is highlighted with a red box (5), showing 12 styles and an example of the text 'Curso em Vídeo'.

- 1 - Se você já sabe o nome de uma fonte, basta digitar nessa área.
- 2 - Na segunda área marcada, você pode escrever um texto de exemplo e vê-lo aplicado em várias fontes.
- 3 - É o tamanho da fonte que será apresentado na tela
- 4 - São as categorias das fontes que serão exibidas, suporta as opções Serif, Sans Serif, Display, Handwriting e Monospace. Você pode escolher mais de uma categoria.
- 5 - Uma lista com as fontes que satisfazem as configurações feitas e um exemplo do texto personalizado aplicado.

Uma vez escolhida a fonte, clique sobre o nome dela (como na área 5, acima) e uma outra tela será exibida, como a seguir. Clique sobre o botão **+ Select this style** e em seguida pressione o ícone superior, conforme marcado na próxima imagem.

This screenshot shows the 'Monoton' font details page. At the top, there is a navigation bar with a 'Google Fonts' logo and a 'Selected family' icon (circled in red). The main title is 'Monoton' with the subtitle 'Designed by Vernon Adams'. Below this, there is a section titled 'Styles' with a 'Regular 400' style listed. An example of the text 'CURSO EM VÍDEO' is shown in the font. To the right of the text is a large red arrow pointing right, followed by a blue button labeled '+ Select this style'.

Ao clicar no ícone superior direito, uma aba lateral chamada **Selected family** aparecerá.

Review

Embed

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

<link> **@import**

```
<style>
@import url('https://fonts.googleapis.com/css2?family=Monoton&
display=swap');
</style>
```

CSS rules to specify families

```
font-family: 'Monoton', cursive;
```

[API docs](#)

Monoton

Designed by Vernon Adams

Styles

Regular 400

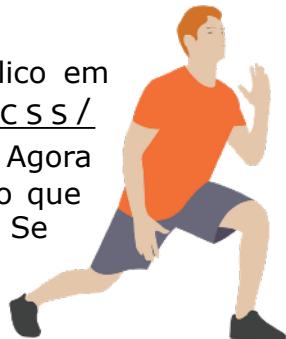
About Monoton

Monoton is a contemporary take on metalpress fonts like the ones used in 1931 by Rudolf Koch. Monoton is a pure display web font with 30 points. Monoton has been designed to be used freely.

Em primeiro lugar, clique em **Embed** e em seguida em **@import** para ter acesso aos códigos que serão colocados no seu arquivo CSS. O código de cima será colocado na primeira linha das suas declarações de estilo. Já o segundo código, especificado em **CSS rules** será colocado na propriedade font-family na declaração de todo seletor onde vamos querer aplicar a fonte.

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 017** 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**.



Quer acompanhar tudo em vídeo?

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 faz parte da playlist completa onde você encontra os **Módulos 1 e 2** do **Curso de HTML5 e CSS3**, completamente gravado com base nesse material.



Módulo 1 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dkZ9-atkcmcBaMZdmLHft8n

Módulo 2 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



1. A tipografia é uma arte antiga cujo nome vem da junção de duas palavras gregas. Marque a opção que descreve corretamente cada um destes termos.

- A týpos que significa impressão e graphía que significa escrita
- B tipus que significa letra e graphia que significa escrever
- C týpos que significa escrita e graphía que significa impressão
- D tipus que significa escrever e graphia que significa letra

2. A tipografia se inicia em 1450, com o inventor alemão chamado:

- A Edmund Germer
- B Konrad Zuse
- C Johannes Gutenberg
- D Alois Senefelder

3. Para os estudos em tipografia, uma letra também pode ser chamada de:

- A epístola
- B glifo
- C família
- D cunho

4. Qual das frases a seguir melhor define o que é uma família tipográfica?

- A é um grupo de palavras que podem ser consideradas harmonicamente organizadas em um texto
- B é um conjunto de letras que possuem variações nas suas características anatômicas
- C é o nome dado a uma fonte e que não pode ser replicado em nenhum serviço de disponibilidade de arquivos
- D é o conjunto de caracteres que possuem as mesmas características anatômicas, independente das suas variações

5. Em relação à anatomia do tipo, temos a medida da altura das letras minúsculas de uma determinada fonte, que chamamos de:

- A altura das minúsculas
- B altura-x
- C altura-y
- D altura-mini

6. As letras g, j e Q possuem uma pequena curva que geralmente é representada na área da descendente. Qual é o nome que se dá a essa curva nas letras indicadas?

- A cauda
- B enlace
- C esporão
- D perna

7. As fontes _____ são aquelas com bastante efeitos visuais, enfeitadas e até mesmo curiosas. Também são chamadas de fontes comemorativas.

- A serifadas
- B monoespaçadas
- C script
- D display

8. Em CSS, para definir que tipo de fonte será utilizado a um determinado elemento HTML, usamos a propriedade:

- A font-type
- B font-face
- C font-name
- D font-family

9. Na maioria dos casos, o tamanho padrão de uma fonte é de _____px, o que equivale a _____em na medida referencial.

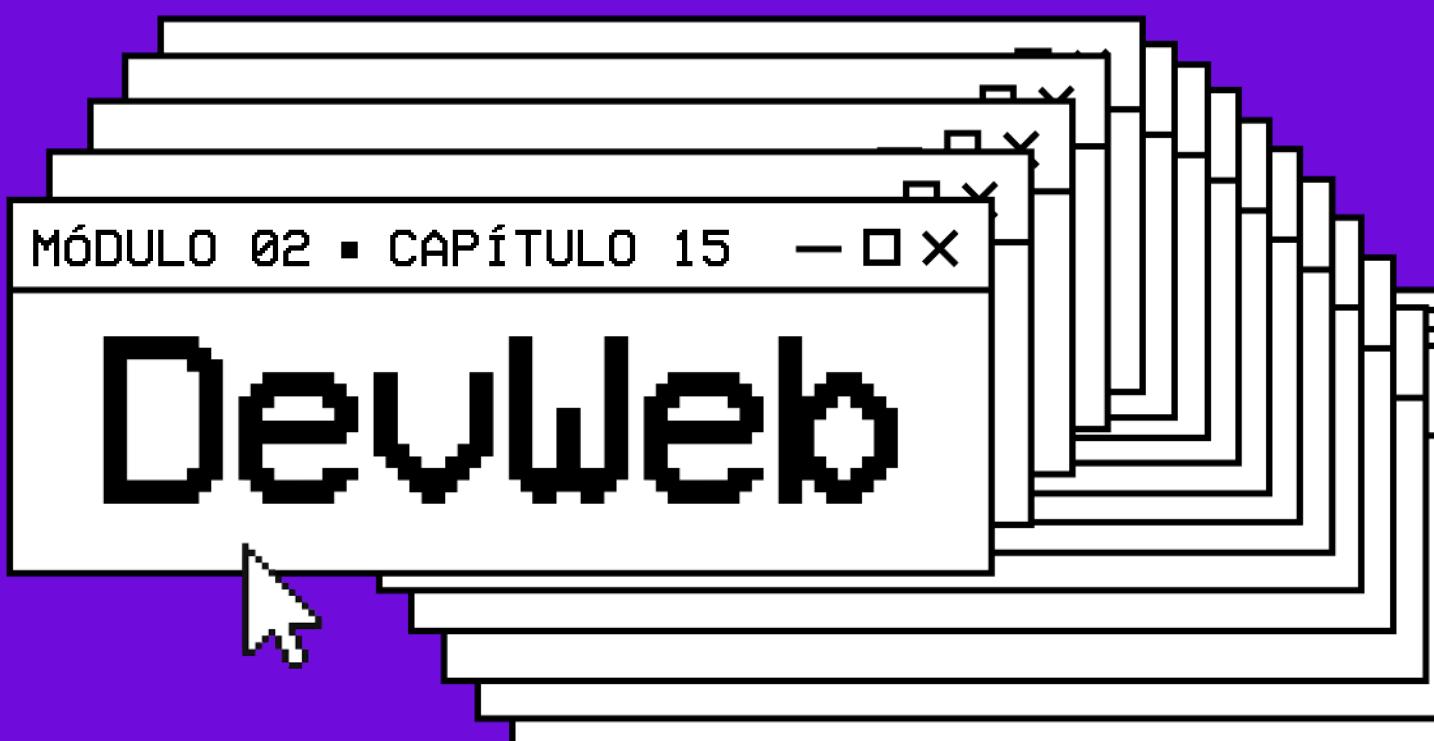
- A 10px / 10em
- B 30px / 2em
- C 16px / 1em
- D 12px / 10em

10. No exemplo de uso da *shorthand* font: italic bold 1em Arial, Helvetica, sans-serif; quais são as propriedades que estão sendo definidas, na ordem?

- A font-style, font-weight, font-size, font-family
- B font-variant, font-weight, font-size, font-style
- C font-style, font-variant, font-style, font-family
- D font-family, font-variant, font-size, font-style

Suas anotações

Não guarde conhecimento. Ele é livre. Compartilhe o seu e veja ele se espalhando pelo mundo 



SELETORES PERSONALIZADOS EM CSS



M02C15

SELETORES PERSONALIZADOS EM CSS

Agora que nós já passamos pelos elementos mais importantes do design, que são as cores, as imagens e as fontes, vamos falar um pouco sobre formatações especiais que vão permitir organizar esses conteúdos na tela, indicando seus tamanhos, espaçamentos, sombras e tudo mais. Eu te convido a aprender maneiras que vão facilitar no desenvolvimento web de sites eficientes e bonitos.



Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.

Personalizando Seletores

Para começar a dar mais poder às CSS, criando estilos personalizados, precisamos aprender a utilizar os seletores de id (#) e de class (.) de maneira eficiente. Ao criar nosso conteúdo em HTML, podemos **identificar** um determinado elemento único com um id, ou **agrupar** elementos múltiplos que tenham características semelhantes com um class. Vamos olhar um exemplo simples de documento HTML, com propriedades identificadoras:

```
22 <body>
23   <h1 id="titulo-principal">Aprenda Desenvolvimento Web</h1>
24   <h1>Aprenda HTML</h1>
25   <h2 class="topico">Semântica Web</h2>
26   <p class="texto">Lorem ipsum dolor sit, amet consectetur adipisicing elit.
    Optio, quibusdam? Tempore sequi commodi natus, vitae enim sed molestias. Rem,
    voluptas illum. Ut, debitis error quas explicabo corporis officia fugit.
    Doloribus.</p>
27   <h1>Aprenda CSS</h1>
28   <h2 class="topico">Estilos Web</h2>
29   <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Dolores, repellat
    dicta. In distinctio quasi nesciunt blanditiis. Quas expedita et architecto
    alias laudantium, facere neque quo facilis! Ut est preferendis sed!</p>
30 </body>
```



Olhando atentamente para a **linha 23**, vimos que o `<h1>` principal se diferencia dos demais que estão nas **linhas 24 e 27**, pois ele possui um id.

Agora foque sua atenção nas **linhas 25 e 28** e perceba que os dois títulos `<h2>` possuem o mesmo class.

Um id vai **identificar** um elemento único dentro da página atual. Essa identificação vai nos permitir criar um estilo especial para um elemento isolado. Já um class vai identificar uma **classe** à qual um ou mais elementos pertençam, compartilhando características em comum a todos os que façam parte desse grupo.



SE LIGA NA REGRA: Em um mesmo documento HTML, só podemos usar um id para um único elemento, o que significa que não podemos ter dois elementos com um mesmo id dentro de uma mesma página. Porém, podemos atribuir um mesmo class para vários elementos que possuam essa mesma característica.

Em seguida, baseado no código HTML visto anteriormente, vamos criar um estilo local criando uma área `<style>` dentro da seção `<head>` do documento atual.

```

7   <style>
8     body {
9       font: normal 1em Arial, Helvetica, sans-serif;
10    }
11
12    h1 {
13      font-size: 2em;
14      color: darkgreen;
15    }
16
17    h2 {
18      color: green;
19    }
20  </style>

```

O que fizemos foi criar configurações simples de estilo baseadas nos elementos body, h1 e h2 do nosso HTML. Até o momento, não fizemos nenhuma configuração personalizada para nenhum seletor. O resultado é bem previsível:

Aprenda Desenvolvimento Web

Aprenda HTML

Semântica Web

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Optio, quibusdam? Tempore sequi commodi natus, vitae enim sed molestias. Rem, voluptas illum. Ut, debitis error quas explicabo corporis officia fugit. Doloribus.

Aprenda CSS

Estilos Web

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Dolores, repellat dicta. In distinctio quasi nesciunt blanditiis. Quas expedita et architecto alias laudantium, facere neque quo facilis! Ut est preferendis sed!

No código ao lado, criamos seletores com simbologias novas. Isso porque não são seletores para elementos HTML, e sim seletores para id e class. Na **linha 21**, criamos um seletor personalizado para o identificador `titulo-principal`, que está atribuído ao h1. Você deve ter percebido que colocamos um símbolo `#` antes do id.

Já as **linhas 27 e 31** são seletores que receberão declarações para as classes `topico` e `texto`. Todos os seletores personalizados para uma class são identificados por um ponto que vem antes do nome da classe.

Perceba no resultado ao lado que os dois primeiros títulos possuem apresentações idênticas, mesmo que o primeiro deles possua um id.

Isso aconteceu porque não fizemos nenhuma configuração específica para ele. Não adianta colocar apenas `id` ou `class` em um elemento e não personalizar o resultado visual usando seletores personalizados.

Agora vamos adicionar algumas declarações e seletores personalizados, ainda dentro da área `<style>` que criamos:

```

21  #titulo-principal {
22    background-color: darkgreen;
23    color: white;
24    text-align: center;
25  }
26
27  .topico {
28    text-align: right;
29  }
30
31  .texto {
32    text-align: justify;
33  }

```



MAIS UMA REGRA: Todo id em HTML é identificado nas CSS por um símbolo #. Toda class em HTML é identificada nas CSS por um ponto.

Ao adicionar os seletores personalizados ao nosso estilo local, o resultado fica bem diferente:

ANTES...	...DEPOIS
<p>Aprenda Desenvolvimento Web</p> <p>Aprenda HTML</p> <p>Semântica Web</p> <p> Lorem ipsum dolor sit, amet consectetur adipisicing elit. Optio, quibusdam? Tempore sequi commodi natus, vitae enim sed molestias. Rem, voluptas illum. Ut, debitis error quas explicabo corporis officia fugit. Doloribus.</p> <p>Aprenda CSS</p> <p>Estilos Web</p> <p> Lorem ipsum dolor sit, amet consectetur adipisicing elit. Dolores, repellat dicta. In distinctio quasi nesciunt blanditiis. Quas expedita et architecto alias laudantium, facere neque quo facilis! Ut est preferendis sed!</p>	<p>Aprenda Desenvolvimento Web</p> <p>Aprenda HTML</p> <p>Semântica Web</p> <p> Lorem ipsum dolor sit, amet consectetur adipisicing elit. Optio, quibusdam? Tempore sequi commodi natus, vitae enim sed molestias. Rem, voluptas illum. Ut, debitis error quas explicabo corporis officia fugit. Doloribus.</p> <p>Aprenda CSS</p> <p>Estilos Web</p> <p> Lorem ipsum dolor sit, amet consectetur adipisicing elit. Dolores, repellat dicta. In distinctio quasi nesciunt blanditiis. Quas expedita et architecto alias laudantium, facere neque quo facilis! Ut est preferendis sed!</p>

Analise os dois resultados acima e perceba as diferenças. Note que apenas o título h1 identificado como titulo-principal ganhou uma formatação diferente e os demais títulos de mesmo nível se mantiveram como antes. Inclusive, compare o resultado diferente obtido entre o primeiro e o segundo parágrafos. Apenas o primeiro ficou justificado. Isso foi por conta da aplicação de uma class apenas ao primeiro (volte lá no código HTML inicial e confira que o segundo parágrafo não possui class).

The screenshot shows a web browser window with a purple-themed landing page. The main text on the page reads "Soluções digitais para negócios". Below this text is the Hostnet logo, which consists of a stylized white cloud icon followed by the word "hostnet". To the right of the text is a QR code. The browser interface includes standard window controls (minimize, maximize, close) at the top right.

Propriedades herdadas

No **Capítulo 12**, discutimos três técnicas usadas para aplicar estilos aos nossos documentos HTML: **inline**, **local** e **externa**. Durante os nossos vídeos de acompanhamento do material em PDF, demonstramos que as três técnicas podem ser usadas em conjunto, e os estilos de cada serão combinados nessa ordem:



- 1º lugar: CSS externo
- 2º lugar: CSS interno/local
- 3º lugar: CSS inline

Sendo assim, caso haja duplicidade nas configurações, o que fica valendo no final é a propriedade especificada nas configurações inline. Cada camada de aplicação de estilos vai adicionando propriedades novas e sobrescrevendo as configurações do nível anterior.

Quando nós aplicamos configurações personalizadas de `id` ou de `class`, também vai existir uma combinação de configurações, como você já deve ter percebido ao analisar o exemplo que apresentamos nas páginas anteriores.

Voltando ao exemplo dado anteriormente (que vou replicar aqui para facilitar o entendimento), logo no início do corpo, tínhamos dois elementos `h1`:

```
<body>
  <h1 id="titulo-principal">Aprenda Desenvolvimento Web</h1>
  <h1>Aprenda HTML</h1>
```

Logo em seguida, na área de estilo, criamos uma configuração especial para os elementos `h1`:

```
h1 {
  font-size: 2em;
  color: darkgreen;
}
```

Essas duas configurações de tamanho e cor da fonte serão aplicadas aos dois títulos, já que ambos são do tipo `h1`. Por fim, adicionamos uma configuração específica para o elemento que tem um `id` personalizado:

```
#titulo-principal {
  background-color: darkgreen;
  color: white;
  text-align: center;
}
```

Dessa maneira, o primeiro `h1` do código HTML (aquele que tem um `id`) ganhará mais três configurações - cor de fundo, cor da fonte e alinhamento do texto - graças a esse seletor extra. Note que na primeira configuração CSS dos elementos `h1`, dissemos que a cor era `darkgreen`.

Porém, quando aplicamos a segunda configuração de color no seletor `#titulo-principal`, ela vai sobrepor o valor da propriedade para `white`. As outras duas

propriedades (background-color e text-align) serão adicionadas às configurações desse primeiro título. O segundo título não sofrerá alterações, já que apenas o primeiro possui o id referido no seletor.



Aprenda Desenvolvimento Web

Aprenda HTML

Pseudo-classes e pseudo-elementos

Uma pseudo-classe CSS é uma palavra-chave adicionada às declarações de um seletor após um sinal de dois pontos e especificam um estado especial de um elemento. Existem várias pseudo-classes para estilos, podemos citar :hover, :visited, :active, :checked, :empty e :focus.

Já um pseudo-elemento CSS é uma palavra-chave adicionada às declarações de um seletor após dois sinais de dois pontos e permitem que você formate um pedaço específico do elemento referenciado. Os principais pseudo-elementos usados nas CSS são ::before, ::after, ::first-letter, ::first-line.

Vamos começar criando um exemplo bem simples e bastante usado para exemplificar o uso de pseudo-classes e pseudo-elementos: a personalização de links.

```
43  <body>
44      <h1>Personalizando um link</h1>
45      <p>
46          Para aprender HTML5 e CSS3, acesse o
47          <a href="https://gustavoguanabara.github.io">
48              GitHub do Guanabara
49          </a>
50      </p>
51      <p>
52          Se quiser conhecer mais sobre os padrões, visite o site do
53          <a href="https://www.w3c.br">Escritório do W3C no Brasil</a>
54      </p>
55      <p>Para acompanhar os cursos, acesse o
56          <a href="https://www.youtube.com/cursoemvideo/" class="especial">
57              canal do CursoemVideo
58          </a>
59      </p>
60  </body>
```

Começando pelo corpo do documento, contendo só o HTML, vamos criar um texto com um título e três parágrafos, com três links. Note que o último link recebeu a atribuição de uma classe específica. O resultado visual está sendo apresentado a seguir:

Personalizando um link

Para aprender HTML5 e CSS3, acesse o [GitHub do Guanabara](#)

Se quiser conhecer mais sobre os padrões, visite o site do [Escritório do W3C no Brasil](#)

Para acompanhar os cursos, acesse o [canal do CursoemVideo](#)

Na imagem acima, o primeiro link apareceu com a cor violeta pois já tínhamos visitado o perfil do GitHub anteriormente. Por padrão, os navegadores mostram links inéditos em azul e os visitados em violeta. Todos os links também aparecem sublinhados por padrão.

Vamos alterar essa apresentação padrão com nossas configurações de estilo. Crie a área `<style>` dentro de `<head>` e coloque os seguintes seletores.

```
8   body {
9     font: normal 1em Arial, Helvetica, sans-serif;
10    }
11
12   a {
13     font-weight: bold;
14     text-decoration: none;
15     color: red;
16    }
17
18   a:visited {
19     color: darkred;
20    }
21
22   a:hover {
23     text-decoration: underline;
24 }
```

Nas declarações acima, criamos três configurações para os links: a primeira (**linha 12**) para links inéditos (não visitados), colocando o texto em negrito, removendo o sublinhado e colocando-os em cor vermelha.



No segundo seletor para links (**linha 18**), configuramos as âncoras já visitadas (com a pseudo-classe :visited). Nele, apenas mudamos a cor para vermelho escuro.

Já na terceira declaração (**linha 22**), fizemos configurações para todos os links, quando passarmos o mouse por cima (pseudo-classe :hover) e o sublinhado volta a aparecer.

O resultado visual está apresentado abaixo, compare com o resultado anterior e veja as diferenças:

Personalizando um link

Para aprender HTML5 e CSS3, acesse o [GitHub do Guanabara](#)

Se quiser conhecer mais sobre os padrões, visite o site do [Escritório do W3C no Brasil](#)

Para acompanhar os cursos, acesse o [canal do CursoemVideo](#)

Por fim, vamos adicionar algumas configurações relacionadas à classe especial, criada no documento HTML, para o terceiro link.

```
26      .especial:hover {  
27          color: white;  
28          background-color: black;  
29          text-decoration: none;  
30      }  
31  
32      .especial::before {  
33          content: '»';  
34          font-weight: lighter;  
35      }  
36  
37      .especial::after {  
38          content: '«';  
39          font-weight: lighter;  
40      }
```

Na primeira declaração do código acima (**linha 26**), dizemos que o link da classe especial vai ter letra branca, fundo preto e perderá o sublinhado apenas quando movermos o mouse sobre ele.

Nas próximas declarações (**linhas 32 e 37**), vamos adicionar um símbolo » antes e outro símbolo « depois usando os pseudo-elementos ::before e ::after, respectivamente. O resultado dessas declarações está apresentado a seguir:

Personalizando um link

Para aprender HTML5 e CSS3, acesse o [GitHub do Guanabara](#)

Se quiser conhecer mais sobre os padrões, visite o site do [Escritório do W3C no Brasil](#)

Para acompanhar os cursos, acesse o » [canal do CursoemVideo](#) «

Mais um exemplo

Vamos criar mais um exemplo para o uso de pseudo-classes. Nosso HTML vai ter o seguinte corpo:

```
18 <body>
19     <h1>Conteúdo especial</h1>
20     <div>
21         Passe o mouse aqui
22         <p>SURPRESAAAA!</p>
23     </div>
24     <p>Veja o conteúdo oculto aparecendo!</p>
25 </body>
```

Na **linha 20**, criamos um bloco especial com a tag `<div>`. Uma das grandes vantagens em usar divs é que elas podem ter outras tags dentro dela, assim como o parágrafo interno que criamos na **linha 22**.

Vamos criar nosso estilo agora, dentro da área `<head>`:

```
7     <style>
8         div > p {
9             display: none;
10        }
11
12         div:hover > p {
13             display: block;
14             background-color: yellow;
15        }
16     </style>
```

O primeiro seletor, na **linha 8**, vai esconder o parágrafo que está dentro da div (representado em CSS como `div > p`) através da propriedade `display` com o valor `none`.

O segundo seletor, na **linha 12**, vai fazer o parágrafo escondido reaparecer, com o fundo pintado de amarelo apenas quando passarmos o mouse sobre ele.

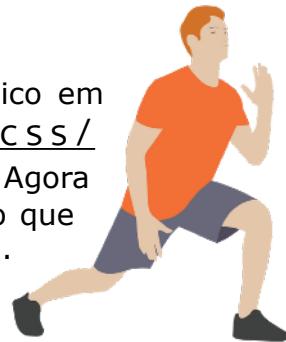


MAIS UM SÍMBOLO: Toda vez que usamos o símbolo > em um seletor, indicamos os filhos (*children*) imediatos de um elemento. No exemplo HTML que criamos, dizemos que o primeiro parágrafo é *direct children* da div.

SEM MOUSE	COM MOUSE
<h2>Conteúdo especial</h2> <p>Passe o mouse aqui Veja o conteúdo oculto aparecendo!</p>	<h2>Conteúdo especial</h2> <p>Passe o mouse aqui  SURPRESAAAA! Veja o conteúdo oculto aparecendo!</p>

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 017** 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.**



— □ ×

Cursos grátis de tecnologia
que te preparam para o
mercado de trabalho

RECODE



Quer acompanhar tudo em vídeo?

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 faz parte da playlist completa onde você encontra os **Módulos 1 e 2** do **Curso de HTML5 e CSS3**, completamente gravado com base nesse material.



Módulo 1 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dkZ9-atkcmcBaMZdmLHft8n

Módulo 2 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



1. Em um mesmo documento HTML, vários elementos podem possuir o mesmo _____ para criarmos formatações em grupo com CSS, mas só podemos ter um elemento com um _____ único para criar formatações individuais.

- A id / class
- B class / id
- C style / class
- D code / id

2. Toda configuração de folhas de estilo que fizermos para um determinado id, deve ser feita iniciando pelo símbolo _____ antes do identificador. Já as configurações para um determinado class serão feitas usando o símbolo _____ antes do nome da classe.

- A # e .
- B . e #
- C # e \$
- D \$ e .

3. Como já sabemos, existem três técnicas para inserir configurações CSS a um código HTML5: inline, local e externa. Se uma página usa as três técnicas ao mesmo tempo e cria uma configuração duplicada para uma determinada propriedade, qual delas vai se sobrepor às demais?

- A CSS local
- B CSS externo
- C CSS inline
- D Não existe ordem de prioridades para casos de duplicidade

4. Uma pseudo-classe CSS é uma palavra-chave adicionada às declarações de um seletor após um sinal de

- A dois pontos (:)
- B ponto-e-vírgula (;)
- C porcentagem (%)
- D arroba (@)

5. Qual dos itens a seguir é o único que não se refere a uma pseudo-classe CSS?

- A hover
- B before
- C checked
- D empty

6. Para realizar configurações de formatação em um pedaço específico do elemento HTML referenciado (a primeira letra de um texto, por exemplo), devemos aprender a criar seletores para _____.

- A pseudo-classes
- B pseudo-textos
- C pseudo-formatos
- D pseudo-elementos

7. Qual dos itens a seguir é o único que não se refere a um pseudo-elemento CSS?

- A first-line
- B first-letter
- C active
- D before

8. Considere que temos o seguinte link em uma mesma página: `Página 2`. Qual será o nome do seletor para configurarmos o que vai acontecer quando o usuário mover o cursor do mouse por cima do link?

- A a#link:hover
- B a.hover:link
- C a.hover#link
- D a.link:hover

9. Se quisermos que todos os links de uma página tenham um símbolo » depois do texto, devemos usar a formatação:

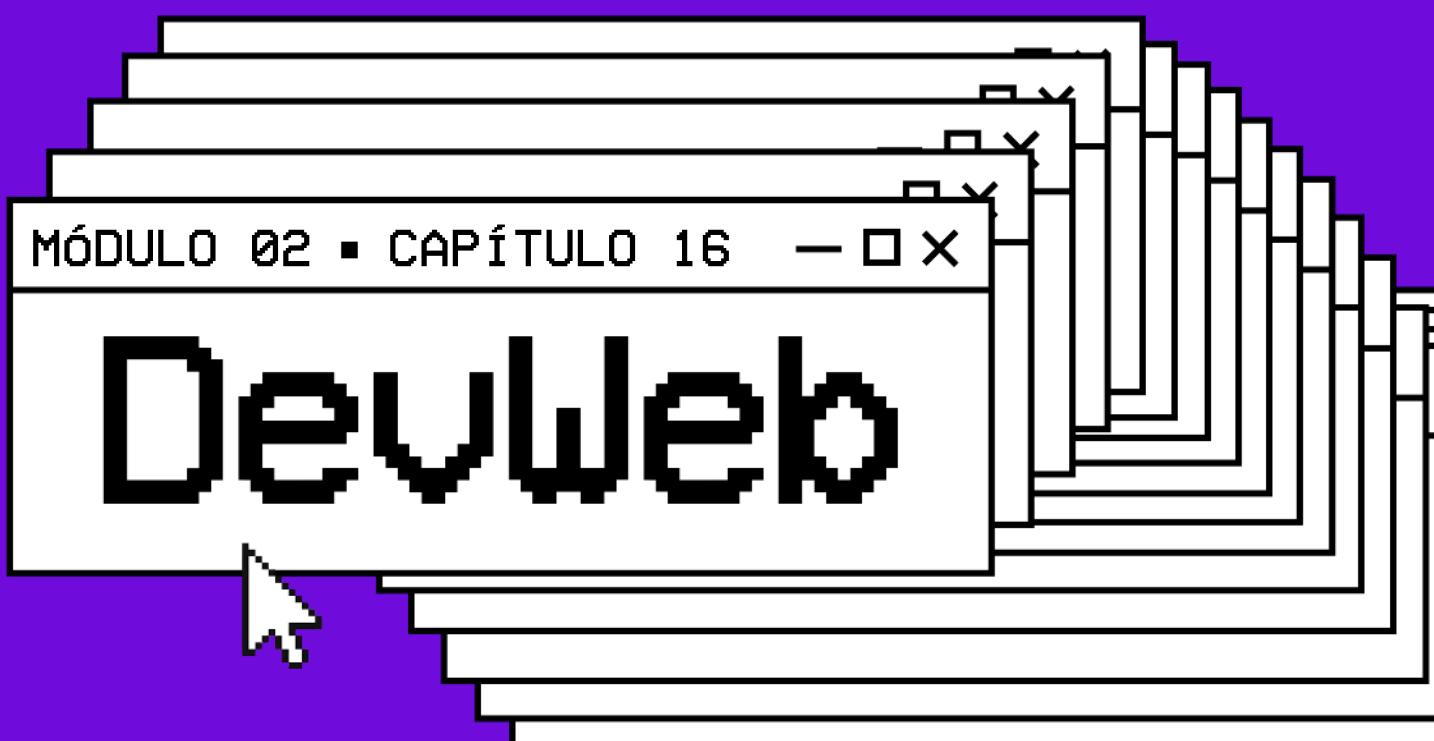
- A a::after { content: '»'; }
- B a { content: '»'; }::after
- C after::a { content: '»'; }
- D ::after { content: '»'; } a

10. Podemos definir seletores com o símbolo de maior que (>) para indicar o que chamamos de:

- A larger tag
- B next tag
- C direct children
- D alternative element

Suas anotações

Não guarde conhecimento. Ele é livre. Compartilhe o seu e veja ele se espalhando pelo mundo



O MODELO DE CAIXAS



M02C16

O MODELO DE CAIXAS

Entender o modelo das caixas é um dos primeiros passos para construir interfaces web e começar a dar forma aos seus sites. Nos últimos quatro capítulos, estudamos a essência das folhas de estilo e já sabemos criar seletores, identificá-los e personalizá-los. Agora chegou a hora de colocarmos tudo isso em caixas configuráveis e vamos começar a desenhar nossas primeiras páginas.



Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.

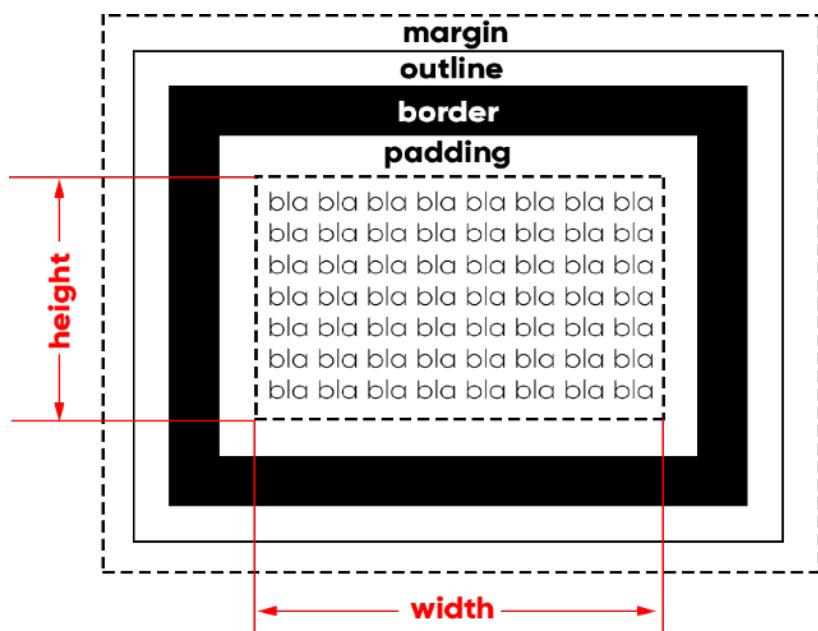
O que é uma caixa?

De forma simples e objetiva, baseado em um conceito chamado “*box model*”, a grande maioria dos elementos HTML que temos no nosso site são como caixas. Elas são *containers* que armazenam conteúdos ou até mesmo outras caixas.



Anatomia de uma caixa

Vamos analisar como uma caixa vai ser apresentada por todos os navegadores. Olhe atentamente o diagrama a seguir, que é exatamente o já citado **modelo de caixa**:



Tudo começa a partir do **conteúdo** (content), que representamos acima com o bla bla bla... Por padrão, toda caixa é composta apenas pelo conteúdo e não possui padding, nem border, nem outline e nem margin. Uma exceção curiosa é o elemento <body> que já vem com uma margin de 8px.

Todo conteúdo possui uma **largura** (width) e uma **altura** (height) e a esse conjunto de propriedades, damos o nome de box-size (tamanho da caixa). O tamanho da caixa não inclui as medidas de padding, border, outline e margin.

Depois do conteúdo e de seu tamanho, vamos nos focar na **borda** que fica em volta dele. Ela pode ter uma espessura, uma cor e um formato.

Entre a borda e o conteúdo - da borda para dentro - temos o **preenchimento** (padding) e da borda para fora, temos a **margem** (margin).

Entre a margem e a borda, podemos determinar o **contorno** (outline) que é muito pouco utilizado, mas existe. Ele é um traçado visual que podemos criar fora da borda e o cálculo da sua espessura faz parte da margem estabelecida.

Vamos criar um exemplo simples para exemplificar todos esses componentes, configurando as propriedades do modelo de caixa de um título `<h1>`. Acompanhe o trecho de código a partir das definições de estilo.

```
7 <style>
8   h1 {
9     width: 300px;
10    height: 50px;
11    background-color: lightgray;
12    border-width: 10px;
13    border-style: solid;
14    border-color: red;
15    padding: 20px;
16    outline-width: 30px;
17    outline-style: solid;
18    outline-color: blue;
19    margin: 50px;
20  }
21 </style>
22 </head>
23 <body>
24   <h1>Exemplo de Caixa</h1>
25 </body>
26 </html>
```

Todas as configurações serão aplicadas ao elemento `<h1>`, que é uma caixa e foi criado na **linha 24** do código acima. As **linhas 9 e 10** configuram o size da caixa (largura e altura, respectivamente) e fará com que ela tenha 300x50 pixels.

As **linhas de 12 a 14**, configuram uma borda sólida, vermelha e com 10 pixels de espessura.

A **linha 15** vai criar um espaço interno de preenchimento (da borda para dentro) de 20 pixels no elemento e a **linha 19** vai criar um espaço externo (da borda para fora) de 50 pixels.



As **linhas de 16 a 18** vão usar parte da margem para criar um contorno azul, sólido e com 30 pixels de espessura.



TAMANHO TOTAL: Para calcular a largura e altura total de um elemento na tela, some os tamanhos do **conteúdo + preenchimento + borda + margem**. O contorno não vai entrar nessa conta, pois utiliza parte da medida da margem.

O resultado visual do código anterior será:



Olhando de perto, podemos analisar as medidas configuradas no código apresentado. As medidas de height e width (300x50) são medidas apenas pela parte pontilhada do conteúdo.

A border de 10px ficou em vermelho e o outline de 30px ficou em azul. O padding de 20px fica da borda para dentro e a margin de 50px fica da borda para fora.

Sendo assim, a medida total que essa caixa vai ocupar é de $50 + 10 + 20 + 300 + 20 + 10 + 50 = \textbf{460px de largura}$ e $50 + 10 + 20 + 50 + 20 + 10 + 50 = \textbf{210px de altura}$.



NOVIDADE DAS CSS3: Existe a nova propriedade box-sizing onde podemos definir que as dimensões height e width não são medidas apenas a partir do conteúdo (content-box) e sim pela borda (border-box).

Dá pra simplificar?

As configurações de borda e contorno também possuem *shorthands* para simplificar o código anterior. A ordem para as duas configurações é sempre a mesma para as duas shorthands: largura (-width), estilo (-style) e cor (-color).

MODO COMPLETO	SHORTHAND
<code>border-width: 10px; border-style: solid; border-color: red; outline-width: 30px; outline-style: solid; outline-color: blue;</code>	<code>border: 10px solid red; outline: 30px solid blue;</code>

Preenchimento e margem personalizados

Todo elemento de caixa possui quatro valores para padding e quatro para margin, sempre nessa mesma ordem: superior (-top), direita (-right), inferior (-bottom), esquerda (-left). Quando colocamos um único valor de dimensão para o preenchimento ou margem, esse mesmo valor é aplicado simetricamente a todas as direções, mas também podemos fazer códigos como:

MODO COMPLETO	SHORTHAND
<code>padding-top: 10px; padding-right: 15px; padding-bottom: 20px; padding-left: 25px; margin-top: 0px; margin-right: 10px; margin-bottom: 20px; margin-left: 30px;</code>	<code>padding: 10px 15px 20px 25px; margin: 0px 10px 20px 30px;</code>

Também existe a opção de indicar cada *shorthand* das propriedades de preenchimento e borda usando apenas duas medidas:

MODO COMPLETO	SHORTHAND
<pre>padding-top: 10px; padding-right: 20px; padding-bottom: 10px; padding-left: 20px; margin-top: 0px; margin-right: 15px; margin-bottom: 0px; margin-left: 15px;</pre>	<pre>padding: 10px 20px; margin: 0px 15px;</pre>

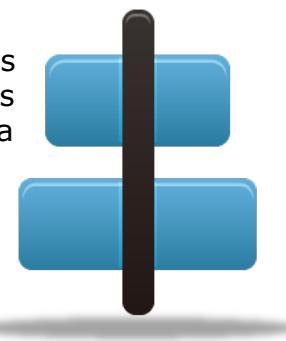
Essa simplificação só é possível quando as medidas -top e -bottom forem iguais entre si e o mesmo também ocorrer entre as medidas -right e -left.

Margens no automático

Um recurso que também vai ser muito usado em nossos exercícios é a centralização de blocos. Para que isso seja feito, devemos pedir que o navegador calcule automaticamente as margens da esquerda e da direita para que o bloco seja colocado no meio do navegador, independente do tamanho da janela.

Para centralizar uma caixa, use a seguinte declaração no seu seletor:

```
margin: auto;
```



Cursos que vão te levar ao próximo nível

estudonauta

QR code

Tipos de Caixa

Dependendo do comportamento da caixa, podemos classificar um elemento em uma de duas categorias:

Caixa do tipo block-level

Um elemento dito *block-level* sempre vai se iniciar em uma nova linha e vai ocupar a largura total do elemento onde ele está contido. Se não estiver contido em nenhuma outra caixa, ele vai ocupar 100% da largura do <body>.

O elemento *block-level* mais conhecido é o <div> e suas variações semânticas modernas da HTML5, como <main>, <section>, <aside>, etc.

Na lista a seguir, coloquei alguns elementos HTML que são block-level:

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>
<div>	<dl>	<dt>	<fieldset>	<figcaption>	<figure>
<footer>	<form>	<h1> - <h6>	<header>	<hr>	
<main>	<nav>	<noscript>		<p>	<pre>
<section>	<table>	<tfoot>		<video>	

Caixa do tipo inline-level

Um elemento do tipo *inline-level* não vai começar em uma nova linha, e sim no ponto exato onde foram definidos. E a largura dele vai ocupar apenas o tamanho relativo ao seu conteúdo.

Abaixo, listei alguns elementos *inline-level* usados pela HTML:

<a>	<abbr>	<acronym>		<bdo>	
<button>	<cite>	<code>	<dfn>		<i>
	<input>	<kbd>	<label>	<map>	<object>
<output>	<q>	<samp>	<script>	<select>	<small>
		<sub>	<textarea>	<tt>	<var>

Grouping Tags e Semantic Tags

A linguagem HTML padrão tinha apenas duas tags de agrupamento genérico: a `<div>` e a ``. A diferença básica entre elas é que a primeira é um elemento agrupador do tipo *block-level* e o segundo é *inline-level*. No mais, eles agem exatamente da mesma maneira, servindo para juntar vários outros elementos HTML.

Com o surgimento da HTML5, surgiram as tags semânticas de agrupamento. Isso não significa que as `<div>` e `` (agora chamadas de não-semânticas) deixaram de existir ou ficaram obsoletas, mas seu uso agora faz menos sentido, pois temos tags para dividir as partes do nosso documento HTML.

Vamos compreender a partir de agora os principais agregadores semânticos da HTML5.

Header

Cria áreas relativas a cabeçalhos. Pode ser o cabeçalho principal de um site ou até mesmo o cabeçalho de uma seção ou artigo. Normalmente inclui títulos `<h1>` - `<h6>` e subtítulos. Podem também conter menus de navegação.

Nav

Define uma área que possui os links de navegação pela estrutura de páginas que vão compor o website. Um `<nav>` pode estar dentro de um `<header>`.



Main

É um agrupador usado para delimitar o conteúdo principal do nosso site. Normalmente concentra as seções, artigos e conteúdos periféricos.

Section

Cria seções para sua página. Ela pode conter o conteúdo diretamente no seu corpo ou dividir os conteúdos em artigos com conteúdos específicos. Segundo a documentação oficial da W3C, “uma seção é um agrupamento temático de conteúdos, tipicamente com um cabeçalho”.

Article

Um artigo é um elemento que vai conter um conteúdo que pode ser lido de forma independente e dizem respeito a um mesmo assunto. Podemos usar um `<article>` para delimitar um post de blog ou fórum, uma notícia, etc.

Aside

Delimita um conteúdo periférico e complementar ao conteúdo principal de um artigo ou seção. Normalmente um conteúdo `<aside>` está posicionado ao lado de um determinado texto ou até mesmo no meio dele, exatamente como fizemos no bloco de texto apresentado anteriormente, falando sobre "MÚLTIPLOS NÍVEIS".



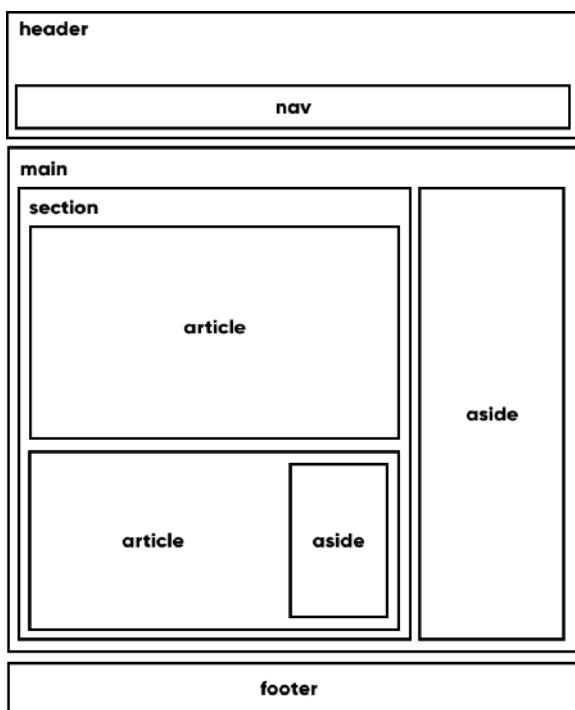
MÚLTIPLOS NÍVEIS: A sua criatividade e planejamento vai definir a estrutura do seu site. Sendo assim, é possível ter um ou mais `<article>` dentro de uma `<section>` ou até mesmo criar `<section>` dentro de um `<article>`. Não existem limitações quanto a isso.

Footer

Cria um rodapé para o site inteiro, seção ou artigo. É um conteúdo que não faz parte diretamente do conteúdo nem é um conteúdo periférico (o que caracterizaria um `<aside>`), mas possui informações sobre autoria do conteúdo, links adicionais, mapa do site, documentos relacionados.

A seguir, vou criar uma proposta de estrutura para um projeto de site. Não tome ela como a única possibilidade de criar o posicionamento de elementos de agrupamento semântico.

A screenshot of a website's footer. The background is purple. On the left, the text "Soluções digitais para negócios" is displayed in large white font. On the right, there is a QR code. At the bottom center, there is a logo for "hostnet" featuring a stylized cloud icon and the word "hostnet" in white.



```

9   <header>
10  <h1>Meu Site</h1>
11  <nav>link link link link...</nav>
12  </header>
13  <main>
14    <section>
15      <article>
16          <h2>Título</h2>
17          <p>Texto do artigo</p>
18      </article>
19      <article>
20          <h2>Título</h2>
21          <p>Texto do artigo</p>
22          <aside>
23              conteúdo periférico do artigo
24          </aside>
25      </article>
26  </section>
27  <aside>
28      conteúdo periférico do site
29  </aside>
30 </main>
31 <footer>
32     conteúdo do rodapé
33 </footer>

```

Analise o diagrama do lado esquerdo e o código do lado direito da imagem acima. Veja a hierarquia entre os elementos e quais deles estão dentro um do outro.

Sombras nas caixas

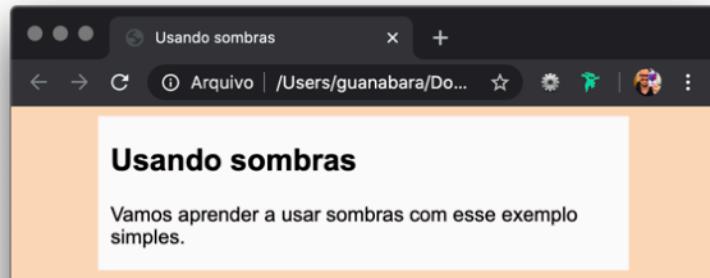
As sombras são muito úteis para dar volume, para dar a sensação de que as caixas estão ali realmente. Para exemplificar, vamos criar o seguinte código base:

```

7   <style>
8     body {
9       font: 1em Arial, Helvetica, sans-serif;
10      background-color: #peachpuff;
11    }
12    article#caixa {
13      background-color: #snow;
14      width: 400px;
15      margin: auto;
16      padding: 1px 10px;
17      /* sombra aqui */
18    }
19  </style>
20 </head>
21 <body>
22   <article id="caixa">
23     <h1>Usando sombras</h1>
24     <p>Vamos aprender a usar sombras com esse
25       exemplo simples. </p>
26   </article>
27 </body>
28 </html>

```

Olhando para o corpo da página, temos apenas um `<article>` (**linha 22**) com um breve conteúdo. A configuração de estilo (**linha 7** em diante) faz com que esse artigo seja configurado como uma pequena caixa centralizada. O resultado visual está apresentado a seguir:



Para criar uma sombra nessa caixa, vamos adicionar uma declaração especial na **linha 17**, substituindo o comentário que deixei lá.

```
box-shadow: 3px 5px 4px black;
```



Veja que uma sombra bem forte já pode ser percebida, assim que adicionamos a propriedade `box-shadow` e seus quatro valores. A ordem é sempre essa:

1. **Deslocamento horizontal** (*h-offset*): quanto a sombra vai andar para o lado direito (valores negativos causam deslocamento para a esquerda)
2. **Deslocamento vertical** (*v-offset*): quanto a sombra vai andar para baixo (valores negativos causam deslocamento para cima)
3. **Embaçamento** (*blur*): quanto a sombra vai se espalhar pelo fundo
4. **Cor** (*color*): cor da sombra. É possível usar transparência.



MUITO CUIDADO! Não exagere no uso de sombras, pois elas podem tornar o seu efeito visual muito pesado. Evite também usar sombras coloridas. Olhe ao seu redor e perceba que as sombras são sempre pretas. Use cores `rgba()` para obter uma transparência que cause efeitos mais suaves.

Bordas decoradas

As bordas das caixas não precisam ser sempre retangulares e podem ter alguns detalhes especiais. Vamos usar o mesmo exercício que estamos criando desde o item anterior onde aprendemos a usar sombras.

Vértices arredondados

Podemos arredondar os vértices usando uma declaração simples usando a propriedade border-radius. Adicione o seguinte comando ao seletor do artigo do exemplo que estamos criando:

border-radius: 10px;

Usando sombras

Vamos aprender a usar sombras com esse exemplo simples.

Na declaração acima, todos os vértices foram levemente arredondados (*10px*) de forma simétrica. Se for necessário, podemos indicar quatro medidas diferentes, uma para cada vértice. Olhe atentamente para o resultado abaixo e perceba que cada ponta está diferente.

border-radius: 10px 20px 30px 40px;

Usando sombras

Vamos aprender a usar sombras com esse exemplo simples.

Assim como fizemos com as margens, também é possível indicar apenas dois valores, o que vai agir em vértices intercalados, partindo do canto superior esquerdo.

border-radius: 10px 30px;

Usando sombras

Vamos aprender a usar sombras com esse exemplo simples.

Tenho desafios pra você!

Lá no repositório, além do material em PDF e dos códigos dos exercícios 100% disponíveis, também disponibilizamos alguns **desafios** que devem ser resolvidos. Esses desafios não incluem o código original e você deve tentar chegar à resposta sem copiar nenhum código.

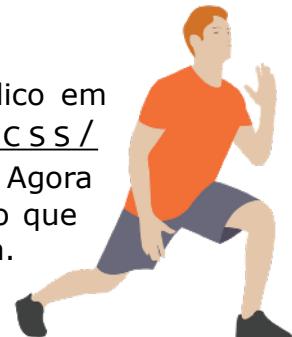
Com todo o conteúdo que vimos até essa aula, você já pode resolver o **desafio d010**. Acesse o repositório público, abra a área do curso de HTML+CSS e clique no link de acesso aos desafios. Manda ver! Só não fica pedindo a resposta! Você consegue resolver isso sozinho(a)!



Repositório em: <https://gustavoguanabara.github.io>

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 017** 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**.



Quer acompanhar tudo em vídeo?

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 faz parte da playlist completa onde você encontra os **Módulos 1 e 2** do **Curso de HTML5 e CSS3**, completamente gravado com base nesse material.



Módulo 1 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dkZ9-atkcmcBaMZdmLHft8n

Módulo 2 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



1. A grande maioria dos elementos HTML são considerados como contêineres que podem guardar conteúdos ou outros contêineres. Este conceito é conhecido pelo termo em Inglês:

- A container box
- B box model
- C container content
- D box content

2. Todo contêiner possui uma propriedade chamada box-size, que é composto por duas medidas: largura e altura, representados respectivamente por:

- A width e height
- B height e width
- C width e weight
- D weight e height

3. Todos os elementos HTML que são caixas podem ter um espaço interno, que fica entre a borda e o conteúdo e se chama _____. Eles também podem ter um espaço externo, que fica além da borda e se chama _____. Qual das opções abaixo é a única que preenche as lacunas na ordem correta?

- A margin / padding
- B outline / padding
- C outline / margin
- D padding / margin

4. A shorthand border agrupa a seguir as propriedades, nesta ordem:

- A border-size + border-type + border-color
- B border-width + border-type + border-color
- C border-width + border-style + border-color
- D border-height + border-style + border-color

5. A shorthand padding, quando especificada com quatro parâmetros, configura as seguintes propriedades, na ordem:

- A padding-top, padding-right, padding-bottom, padding-left
- B padding-top, padding-left, padding-bottom, padding-right
- C padding-top, padding-bottom, padding-left, padding-right
- D padding-left, padding-right, padding-top, padding-bottom

6. De acordo com o comportamento de uma caixa, ela pode ser classificada como:

- A box-level ou inline-level
- B box-level ou block-level
- C block-level ou inline-level
- D container-level ou box-level

7. Qual dos elementos a seguir é o único que não é considerado grouping tag?

- A <aside>
- B <nav>
- C <head>
- D <footer>

8. "Delimita um conteúdo periférico e complementar ao conteúdo principal de um artigo ou seção". Esta descrição se encaixa melhor no conceito de qual elemento HTML5?

- [A] <main>
- [B] <article>
- [C] <session>
- [D] <aside>

9. Para aplicar sombra em um elemento HTML5 exibido como uma caixa, usamos qual propriedade?

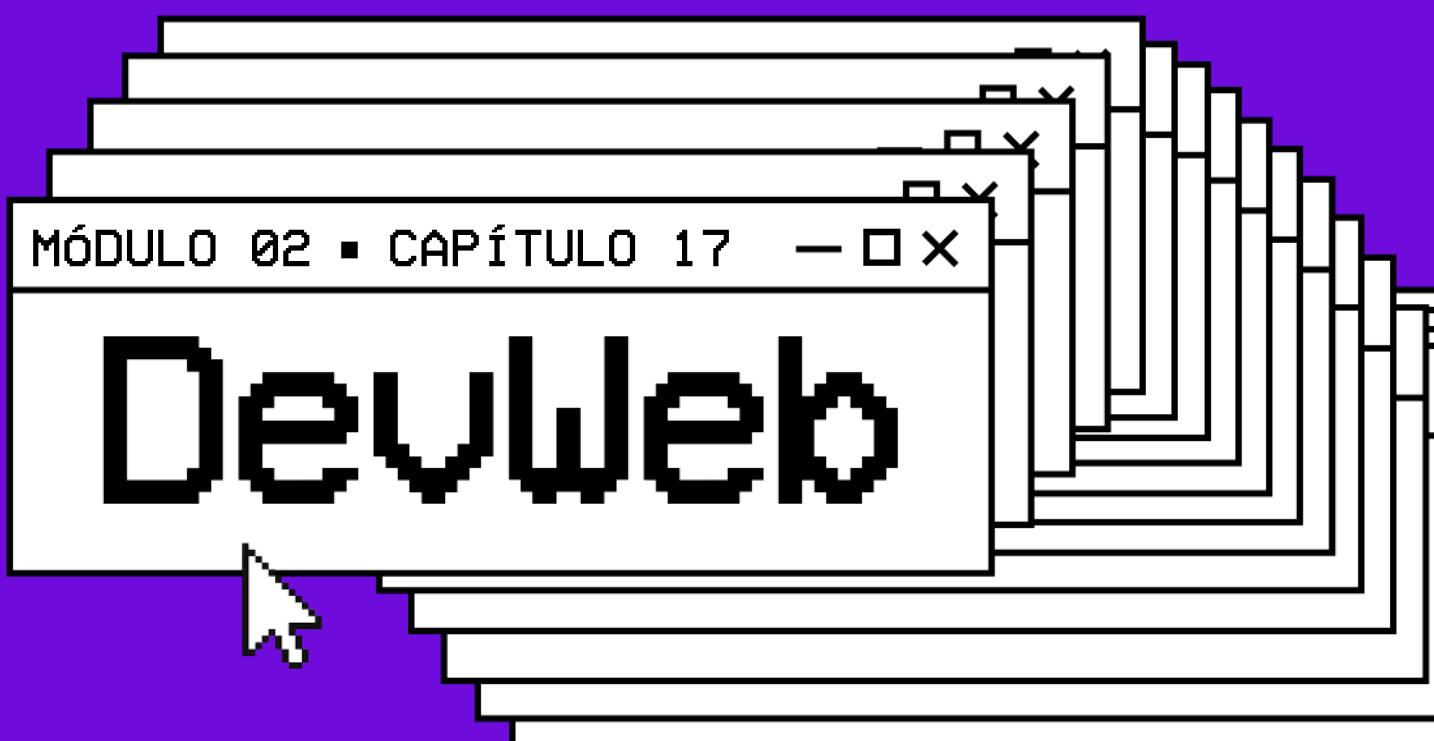
- [A] shadow
- [B] block-shadow
- [C] box-shadow
- [D] shadow-box

10. Para arredondar os vértices de um elemento HTML5 exibido como uma caixa, usamos qual propriedade?

- [A] border-radius
- [B] box-radius
- [C] border-vertex
- [D] vertex-box

Suas anotações

Não guarde conhecimento. Ele é livre. Compartilhe o seu e veja ele se espalhando pelo mundo 



PRIMEIRO MINI-PROJETO



M02C17

PRIMEIRO MINI-PROJETO

Chegou a hora de colocar em prática tudo aquilo que aprendemos até aqui e unificar tudo em um projeto simples. Mas não se engane, você ainda vai aprender coisas novas para poder criar um site básico com uma estrutura interessante. Vamos começar a analisar esse projeto e a aprender conceitos adicionais para torná-lo real.

— □ ×

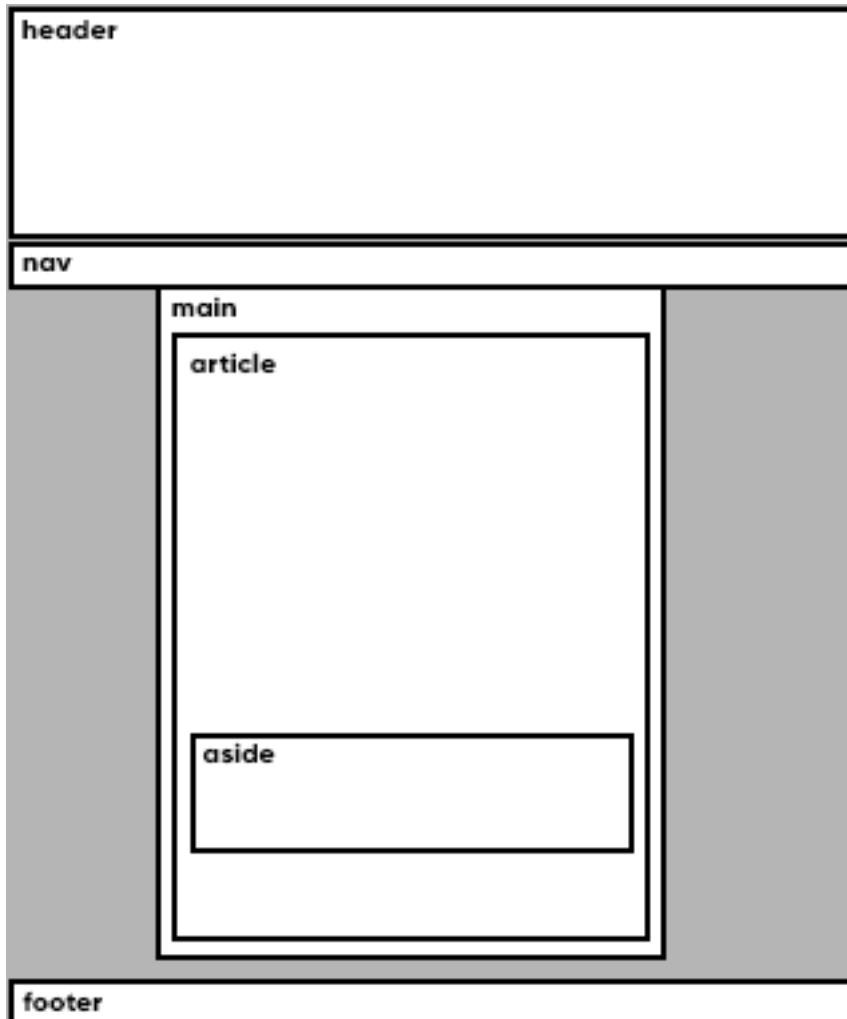


Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos o que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.

A ideia do projeto

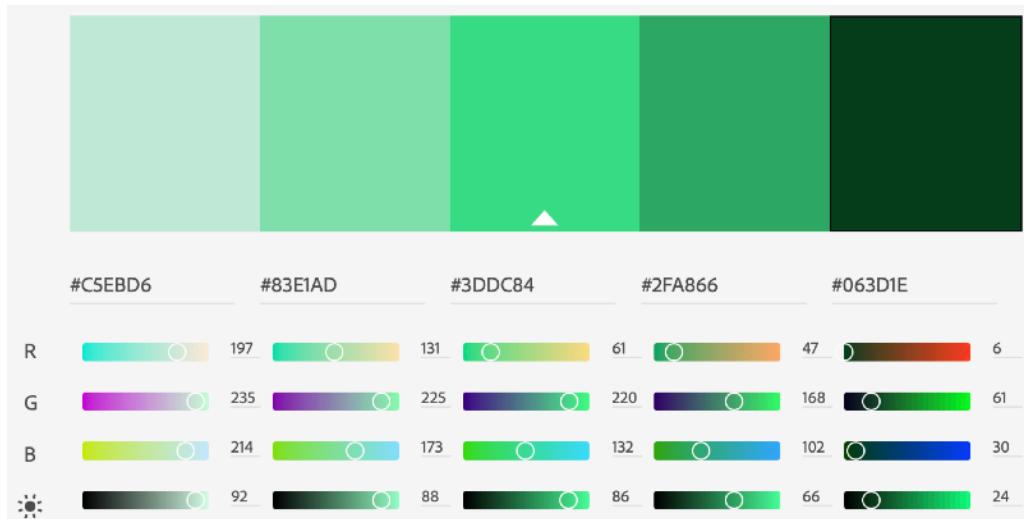
A ideia inicial é criar uma espécie de site de notícias, só que com uma notícia só 😅. O objetivo aqui é apenas ensinar como organizar o conteúdo e apresentá-lo em forma de página Web.

Tudo começa desenhando o que chamamos de *wireframe*, planejando qual vai ser a estrutura e comportamento do site. Ele vai servir de base na hora de planejar as caixas que farão parte da página. Para esse nosso primeiro mini-projeto, planejei o seguinte *layout*:



CRIE SEUS WIREFRAMES: Existe uma ferramenta muito legal para planejar seus *layouts*, tanto para sites quanto para aplicativos desktop e até apps de celular: o **MockFlow**. Como podemos imaginar, existem limitações na versão free, mas dá pra imaginar como o site vai ficar antes de começar a mexer no código.

Em seguida, vamos decidir a paleta de cores que será utilizada. Optei por uma paleta baseada em **monocromia** a partir de tons de verde (já que vamos falar sobre Android), mas você pode usar a técnica que quiser para decidir suas cores básicas que serão usadas. Usei o **Adobe Color** (que aprendemos a usar no capítulo 13) e optei pelas seguintes cores:



Por fim, vamos falar das fontes escolhidas. A primeira, **Bebas Neue**, será a fonte para os destaques principais. Já que vamos fazer uma matéria sobre o mundo Android, optei por uma fonte chamada **iDroid**. Já para o texto padrão, escolhi a **Arial**.

LOREM IPSUM

Bebas Neue, cursive

LOREM IPSUM

iDroid, sans-serif

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Eum, quo temporibus voluptates mollitia eius voluptas qui, officia blanditiis voluptatum ipsum exercitationem incident. Nemo aperiam laboriosam corporis magnam aliquid perspiciatis quae?

Arial, Helvetica, sans-serif

A screenshot of a mobile application interface. The top bar is dark blue with white icons for minimize, maximize, and close. The main content area has a dark purple background with a large white text area. The text reads: "Cursos que vão te levar ao próximo nível". In the bottom right corner, there is a QR code. The bottom left corner features the Estudonauta logo, which includes a stylized figure icon and the text "estudonauta".

Já ouviu falar em responsividade?

Ainda que tenha optado por criar um projeto simples, não vamos abrir mão da versatilidade, pois queremos que nosso site possa ser visualizado em vários dispositivos com tamanhos diferentes de tela. O nome que damos a essa característica é **responsividade**. Um site dito responsivo vai ser capaz adaptar os tamanhos dos seus componentes para manter o site plenamente visível em qualquer tela. É claro que por enquanto, não aprendemos muita coisa para permitir essa adaptação completamente para todo tipo de situação, mas com o que vimos até aqui já dá pra começar.



Largura e altura adaptáveis

No capítulo anterior, vimos as propriedades para largura (`width`) e altura (`height`) de uma caixa. Se não configurarmos a largura, por exemplo, ela vai ocupar 100% do seu contêiner. Porém, para que nosso site se adapte mais facilmente ao tamanho da tela, uma das técnicas básicas que podem ser utilizadas é deixar algumas caixas estratégicamente um pouco mais flexíveis indicando valores mínimos (`min-width` / `min-height`) e máximos (`max-width` / `max-height`). Para o nosso projeto, vamos definir os limites da largura da caixa `<main>` onde estará contido o conteúdo principal do artigo.

```
min-width: 400px;  
max-width: 800px;
```

A configuração acima vai garantir que a caixa em questão não fique com largura menor que 400px (para telas pequenas) e nem seja maior que 800px (para telas muito grandes).

Para medidas entre 400px e 800px, o tamanho será adaptável e a caixa vai ocupar 100% da largura do contêiner. Isso vai facilitar bastante a adaptação do conteúdo a diversos tipos de tela.



DÁ PRA SER MAIS ADAPTÁVEL? É possível aplicar outras técnicas adicionais para dar ainda mais capacidade de adaptação do conteúdo, como as *media queries* com a regra `@media` e as caixas flexíveis com as *flex boxes*. Mais pra frente, abordaremos esses conteúdos no curso, mas por enquanto as medidas adaptáveis já vão quebrar um galho.

IMPORTANTE: variáveis em CSS

Com essa dica, você vai levar suas folhas de estilo a um outro nível! Vamos aprender a utilizar variáveis personalizadas com CSS para cadastrar os esquemas de cores e fontes do nosso site e isso vai facilitar muito na hora de efetuar eventuais mudanças estéticas no nosso site.

Para criarmos variáveis para nossas configurações, devemos definir uma área de definições dentro do seu estilo para uma pseudo-classe chamada :root, que definem as configurações para a raiz de uma árvore, que vai servir para o documento inteiro.

Note que, pelas declarações criadas ao lado, definimos nove variáveis com as configurações de cores e fontes que definimos no início desse capítulo.

```
:root {  
    --cor0: #c5ebd6;  
    --cor1: #83E1AD;  
    --cor2: #3DDC84;  
    --cor3: #2FA866;  
    --cor4: #1A5C37;  
    --cor5: #063d1e;  
    --fonte0: Arial, Helvetica, sans-serif;  
    --fonte1: 'Bebas Neue', cursive;  
    --fonte2: 'Android', sans-serif;  
}
```

A partir de agora, definir cores e fontes em nossos elementos HTML ficará extremamente mais fácil e personalizável, utilizando a função var().

```
body {  
    font-family: var(--fonte0);  
    color: var(--cor0);  
    background-color: var(--cor1);  
}
```



IMPORTANTE: As variáveis personalizadas em CSS devem ter seus nomes iniciando com dois traços obrigatoriamente.

Olhando as declarações feitas para o seletor de body acima, pode parecer inicialmente que elas ficaram um pouco mais confusas. Mas imagine que seu cliente mude as definições de cores e fontes. Quantas alterações teriam que ser feitas para deixar tudo em dia com as novas especificações? Ao usar variáveis, tudo se simplifica muito, pois basta atualizar as variáveis definidas em :root.

The screenshot shows a web browser window with a purple-themed landing page. The main heading is "Soluções digitais para negócios". Below it is the "hostnet" logo, which consists of a stylized white cloud icon followed by the word "hostnet". To the right of the text is a QR code.

Uso do seletor * em CSS

Existe também um seletor especial das CSS que é o asterisco (*), ele tem uma função muito especial, pois basicamente ele aplica uma configuração padrão para **TODOS** os elementos do código HTML ao qual o estilo está sendo aplicado.

No nosso caso, para o nosso mini-projeto, nós vamos utilizar esse seletor global para eliminar as eventuais margens que os navegadores (*user agents*) adicionam a alguns elementos. Isso vai facilitar bastante, pois vai permitir personalizar as medidas que vão aparecer na tela para cada elemento individualmente.

Espaçamento entrelinhas em Textos

Outra configuração muito importante que podemos fazer para textos muito longos é o de espaçamento entre as linhas do nosso texto. Usando a propriedade `line-height`, podemos dizer qual é o tamanho do espaço entre uma linha e a outra do texto. O valor padrão na maioria dos navegadores é algo próximo ao 1.2em, mas veja a seguir a aplicação de outros valores.

Provavelmente você sabe que o sistema operacional Android , mantido pelo Google é um dos mais utilizados para dispositivos móveis em todo o mundo. Mas tavez você não saiba que o seu simpático mascote tem um nome e uma história muito curiosa? Pois acompanhe esse artigo para aprender muita coisa sobre esse robozinho.	Provavelmente você sabe que o sistema operacional Android , mantido pelo Google é um dos mais utilizados para dispositivos móveis em todo o mundo. Mas tavez você não saiba que o seu simpático mascote tem um nome e uma história muito curiosa? Pois acompanhe esse artigo para aprender muita coisa sobre esse robozinho.	Provavelmente você sabe que o sistema operacional Android , mantido pelo Google é um dos mais utilizados para dispositivos móveis em todo o mundo. Mas tavez você não saiba que o seu simpático mascote tem um nome e uma história muito curiosa? Pois acompanhe esse artigo para aprender muita coisa sobre esse robozinho.
<code>line-height: 1.2em;</code>	<code>line-height: 1.5em;</code>	<code>line-height: 2.0em;</code>

Sombras em Textos

No capítulo anterior, falamos sobre as sombras em caixas, utilizando a propriedade `box-shadow`. Pois saiba que também existe uma propriedade específica para criar sombras em textos: o `text-shadow`.

A propriedade `text-shadow` também pode ter quatro parâmetros principais:

1. **Deslocamento horizontal** (*h-offset*): quanto a sombra vai andar para o lado direito (valores negativos causam deslocamento para a esquerda)
2. **Deslocamento vertical** (*v-offset*): quanto a sombra vai andar para baixo (valores negativos causam deslocamento para cima)
3. **Embaçamento** (*blur*): quanto a sombra vai se espalhar pelo fundo
4. **Cor** (*color*): cor da sombra. É possível usar transparência.

`text-shadow: 2px 2px 0px #rgba(0, 0, 0, 0.466);`

CURIOSIDADES DE TECNOLOGIA

Tudo aquilo que você sempre quis saber sobre o mundo Tech, em um único lugar.

Sem text-shadow

CURIOSIDADES DE TECNOLOGIA

Tudo aquilo que você sempre quis saber sobre o mundo Tech, em um único lugar.

Com text-shadow

Como você deve ter percebido olhando as imagens acima, a sombra em textos serve para destacar a letra e seu fundo, criando um contraste entre elas.

Personalizando ainda mais as listas

No **capítulo 9** nós aprendemos a criar listas de vários tipos. Agora, vou te mostrar como criar mais personalizações e a criar um ótimo resultado visual.

No nosso projeto, quero adicionar uma lista com todas as 14 versões principais do sistema Android. Se fizermos usando apenas os elementos comuns sem configurá-los, teremos uma listagem que ocupa um grande espaço vertical. A solução aqui é dividir a lista em duas colunas e modificar o marcador para personalizar ainda mais a exibição do conteúdo.

```
ul {  
    list-style-type: '\2714\0020\0020';  
    columns: 2;  
    list-style-position: inside;  
}
```

A primeira linha de declarações faz com que o marcador seja personalizado com o parâmetro `list-style-type`. O valor \2714 corresponde ao símbolo ✓ que tem o código Unicode U+2714 (confira no site da Emojipedia). O valor \0020 corresponde a um espaço em branco (também pode ser \00A0).

A segunda declaração vai organizar a lista em duas colunas. O total de elementos da lista com `` será dividido em duas partes iguais (ou quase) e o resultado será colunado.

**Cursos grátis de tecnologia
que te preparam para o
mercado de trabalho**

RECODE



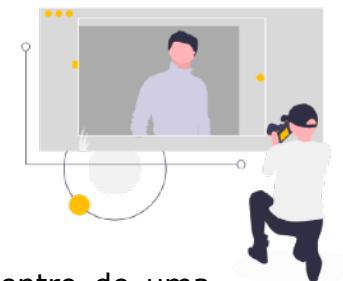
Por fim, a última declaração vai fazer com que os marcadores sejam exibidos na parte interna da caixa que contém a lista. Analise as imagens abaixo e perceba que, por padrão, a caixa de uma lista não inclui os marcadores. Alteramos essa característica usando a declaração `list-style-position` com o valor `inside`, já que a lista vai estar dentro de um `<aside>` no nosso documento HTML.

	
<code>list-style-position: outside;</code>	<code>list-style-position: inside;</code>

Vídeos do YouTube mais flexíveis

Aprendemos no **capítulo 11** como deixar as imagens mais “flexíveis” usando a tag `<picture>`. Também aprendemos a adicionar vídeos usando várias técnicas, inclusive aqueles que estão hospedados em serviços especializados como **Vimeo** e **YouTube**.

O problema é que, quando incorporamos um vídeo do YouTube ou Vimeo, isso é feito através de uma tag `<iframe>` que já vem com as configurações de um tamanho fixo e precisamos alterar isso para que nossa interface possa se tornar mais responsiva e adaptar o tamanho do vídeo dinamicamente, principalmente para telas pequenas.



Para isso, vamos colocar o `<iframe>` de incorporação dentro de uma `<div>` para que o vídeo esteja limitado dentro de um contêiner.

```
<div class="video">
    <!-- AQUI VAI O CÓDIGO DO IFRAME -->
</div>
```

A partir daí, vamos fazer configurações de estilo para os dois elementos:

```
div.video {
    position: relative;
    background-color: var(--cor4);
    height: 0px;
    margin-left: -20px;
    margin-right: -20px;
    margin-bottom: 15px;
    padding-bottom: 59%;
}

div.video > iframe {
    position: absolute;
    top: 5%;
    left: 5%;
    width: 90%;
    height: 90%;
}
```

O que nos importa mais aqui é entender o funcionamento da propriedade `position`, que é a única que não vimos até aqui. O valor padrão para essa propriedade de posicionamento é `static`, que mantém a hierarquia conforme estabelecido no documento HTML.

Na div, nós colocamos o valor como relative para que seja considerado o posicionamento atual do elemento de divisão e que ele se mantenha adaptável para o caso de alteração no tamanho do navegador.

Já dentro do iframe, nós usamos o posicionamento absolute para que a div - que é o seu contêiner - torne-se o ponto de partida para o posicionamento do frame. A partir daí, podemos utilizar propriedades para configurar o deslocamento à esquerda (left) e ao topo (top) e seu tamanho em largura (width) e altura (height), todos em porcentagem de tela.

Tenho um desafio pra você!

Lá no repositório, além do material em PDF e dos códigos dos exercícios 100% disponíveis, também disponibilizamos alguns **desafios** que devem ser resolvidos. Esses desafios não incluem o código original e você deve tentar chegar à resposta sem copiar nenhum código.

Com todo o conteúdo que vimos até essa aula, você já pode resolver o **desafio d010**. Acesse o repositório público, abra a área do curso de HTML+CSS e clique no link de acesso aos desafios. Manda ver! Só não fica pedindo a resposta! Você consegue resolver isso sozinho(a)!

Repositório em: <https://gustavoguanabara.github.io>



Quer acompanhar tudo em vídeo?

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 faz parte da playlist completa onde você encontra os **Módulos 1 e 2** do **Curso de HTML5 e CSS3**, completamente gravado com base nesse material.



Módulo 1 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4dkZ9-atkcmcBaMZdmLHft8n

Módulo 2 do curso: https://www.youtube.com/playlist?list=PLHz_AreHm4d1UpEXkY1AyVLQGcpSgVF8s

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



1. Antes de começar a construir um site, uma das técnicas é desenhar a sua estrutura em forma de grades, mais conhecida como:

- A rascunho
- B wireframe
- C sketch
- D framebars

2. Para que um site seja corretamente exibido em vários tipos de dispositivos com telas de tamanhos diferentes, devemos nos preocupar bastante em torná-lo:

- A recursivo
- B responsivo
- C reusável
- D reciclável

3. Para que a largura de uma caixa torne-se adaptável a vários tamanhos de tela, utilizaremos as propriedades:

- A min-width e max-width
- B min-height e max-height
- C left-width e right-width
- D left-height e right-height

4. Para declarar variáveis em CSS, devemos criar uma pseudo-classe especial dentro do documento CSS atual. Qual é o nome dessa pseudo-classe?

- A :var
- B :declarations
- C :root
- D :variables

5. Toda variável declarada em um documento CSS deve obrigatoriamente começar com:

- A dois traços --
- B um cifrão \$
- C uma cerquilha #
- D uma arroba @

6. Um documento CSS que possui variáveis definidas pode fazer referência a elas utilizando a função:

- A variable()
- B root()
- C env()
- D var()

7. Os navegadores (user agents) aplicam configurações padrão para todo e qualquer elemento HTML5 que é exibido como uma caixa. Para alterar as definições dessas configurações default, usamos um seletor CSS especial que é o:

- A default {}
- B asterisco * {}
- C agent {}
- D global {}

8. Para aplicar sombras às letras que estão dentro de uma caixa, podemos usar a propriedade:

- A text-shadow
- B font-shadow
- C letter-shadow
- D box-shadow

9. Ao usar listas do tipo em HTML5, só conseguimos definir marcadores do tipo circle, square ou disc. Porém, podemos personalizar o símbolo de cada elemento via CSS, usando a propriedade:

- A list-type
- B list-style-type
- C list-symbol
- D list-style-symbol

10. Qual dos valores a seguir é o único que não será aceito para configurar a propriedade position em CSS?

- A static
- B absolute
- C relative
- D fullsize

Suas anotações

Não guarde conhecimento. Ele é livre. Compartilhe o seu e veja ele se espalhando pelo mundo 

DevWeb

Capítulo 18

Aprendendo Git e GitHub

Talvez o conteúdo desse capítulo seja um dos que você mais vai usar durante a sua carreira, que está começando agora. Vamos aprender agora o que são os repositórios, tanto os locais quanto os remotos e qual é a diferença entre eles. E por fim, vamos aprender a colocar nossos projetos no ar, para que possamos acessar os sites em qualquer lugar com conexão à Internet.



Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso

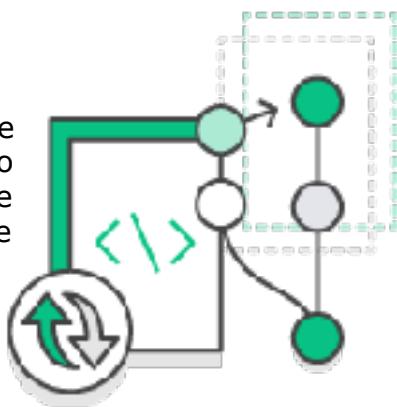
a todo o conteúdo e usá-los com seus alunos. Porém todos o que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof.**

Gustavo Guanabara e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.



Você precisa de um repositório para ser feliz

Talvez nesse exato momento você não faça ideia do que seja um **repositório** e esteja achando que minha afirmação acima foi um pouco exagerada, mas saiba que vou me esforçar para que você consiga entender o uso desse recurso e adote na sua vida de hoje em diante. Repositórios não vão te ajudar apenas no desenvolvimento em **HTML** e **CSS**, eles também serão úteis na criação de qualquer tipo de código em qualquer linguagem de programação que você vá decidir aprender de hoje em diante.



Para explicar a utilidade de cada um dos dois tipos de repositórios que vamos usar (**locais** e **remotos**), criei três pequenas histórias que podem acontecer na sua vida.

Situação 1: como guardar várias versões do seu site?



Quando estamos criando um projeto, começamos aos poucos criando a sua primeira versão dentro de uma pasta local em nosso computador.

Quando atingir uma versão estável, para manter um “*backup* seguro”, sempre usamos ideias brilhantes como gerar uma versão compactada dessa pasta. Isso vai permitir continuar a desenvolver uma versão aprimorada sem perder a versão anterior. Se por acaso algo der errado (e acredite, geralmente dá) basta descompactar nosso arquivo de versão estável, substituir o conteúdo da pasta original e recomeçar os trabalhos da criação da próxima versão. Usando essa prática, com certeza vai acabar se deparando com vários arquivos ZIP com nomes esquisitos.



Esses arquivos compactados ficam se acumulando no seu computador e são úteis caso você precise “voltar no tempo” e desfazer as bobagens que fizemos na madrugada passada quando tentamos desenvolver algo com muito sono.

Situação 2: seu HD foi pro saco, e agora?

Quem nunca perdeu arquivos de trabalho, que atire o primeiro HD queimado! Os dispositivos de hardware sempre param de funcionar no momento em que mais precisamos. E se utilizamos apenas a técnica descrita na situação anteriormente, é

possível que todo o trabalho seja perdido em um estalar de dedos.

Uma das saídas utilizadas é salvar cópias de segurança em *pendrives* e em HDs externos, além de manter um *backup* em serviços de armazenamento como **Dropbox**, **Google Drive**, **OneDrive**, **iCloud**, etc.



O problema de manter esses *backups* isolados é que, no momento de ligar nossa “máquina do tempo” e voltar para uma versão anterior, precisamos fazer o *download* do ZIP desejado e substituir manualmente a versão atual para a última estável.

Situação 3: mostrar seu site para o mundo

E o que acontece quando você precisar mostrar o projeto de um site que você criou para um amigo, para seu professor ou até mesmo para um possível cliente? Vai mandar o link de um arquivo ZIP para ele(a) baixar, descompactar e abrir no navegador local? É assim que as pessoas estão acostumadas a acessar sites?

Os serviços de armazenamento que eu citei na situação anterior são muito úteis para fazer *backup*, mas possuem uma grande limitação quando queremos praticidade. Por exemplo, salvar os arquivos de um site no **Dropbox** não vai garantir de forma alguma que ele vai ficar disponível para o acesso externo. Ninguém vai poder “acessar seu site”, apenas digitando uma URL no navegador. Tudo vai ficar armazenado como um grupo de arquivos, não como um site hospedado.



ATÉ DÁ, MAS... Existe um recurso do **Google Drive** para a hospedagem de sites, mas nada muito simples e intuitivo. A melhor maneira é sempre usar serviços especializados em hospedagem de sites.

Os repositórios vão resolver isso

Todas as três situações descritas podem ser resolvidas com repositórios locais/remotos de maneira muito simples. E se por acaso você já tentou aprender sobre isso, achou guias com dezenas de comandos e não conseguiu se adaptar, saiba que tudo evoluiu muito nos últimos meses e vamos ver tudo de uma maneira muito simples, sem decorar nenhum comando! Para começar, vamos aprender que existem dois tipos de repositórios: os **locais** e os **remotos**.

Repositório Local

Um **repositório local** tem esse nome porque vai estar sempre no seu computador. A função dele é facilitar a gestão de diversas versões do seu projeto de forma simples e automática.



Lembra da **situação 1** que descrevemos anteriormente? Pois quando instalamos um **Sistema de Controle de Versões** (do inglês *Version Control System - VCS*), deixamos a responsabilidade por gerenciar as versões dos nossos projetos nas mãos de um *software* especializado, que vai fazer tudo automaticamente e vai permitir que você volte a qualquer ponto no momento em que achar melhor.

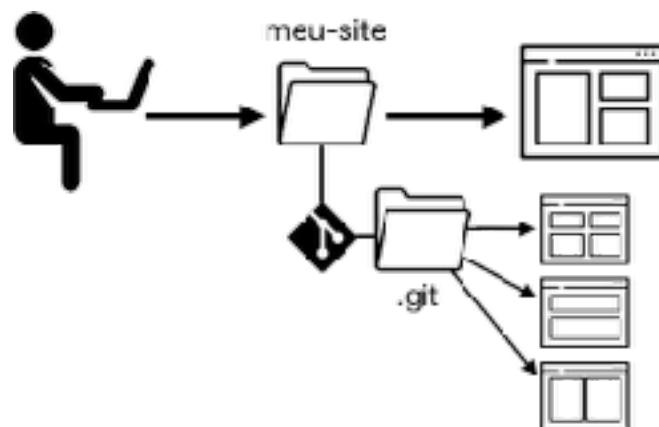
Um dos *softwares* VCS mais famosos é o **Git**, feito por **Linus Torvalds**, o mesmo criador do núcleo **Linux** dos sistemas operacionais. O sistema do Git foi criado no ano de 2005 em poucos dias (10 dias, para ser mais exato), por conta de uma briga entre o *Linus* e o criador de um software chamado **BitKeeper**, que era utilizado para gerenciar as versões em desenvolvimento do *Linux*.



QUER SABER MAIS SOBRE A TRETA? Não vou usar espaço desse material para contar a treta entre **Linus Torvalds** e **Larry McVoy**, mas já contei essa história no YouTube. Se você ficou curioso(a) para saber o que rolou, acesse o link a seguir:

Curso de Git e GitHub: <https://youtu.be/CJtrNuTTs4Q>

O esquema de funcionamento do Git é totalmente focado no nosso computador. Analisando a imagem a seguir, vemos que o software está monitorando uma pasta chamada *meu-site* que tem a versão atual do projeto e uma pasta especial chamada *.git* com várias versões pelas quais o site passou durante sua evolução.

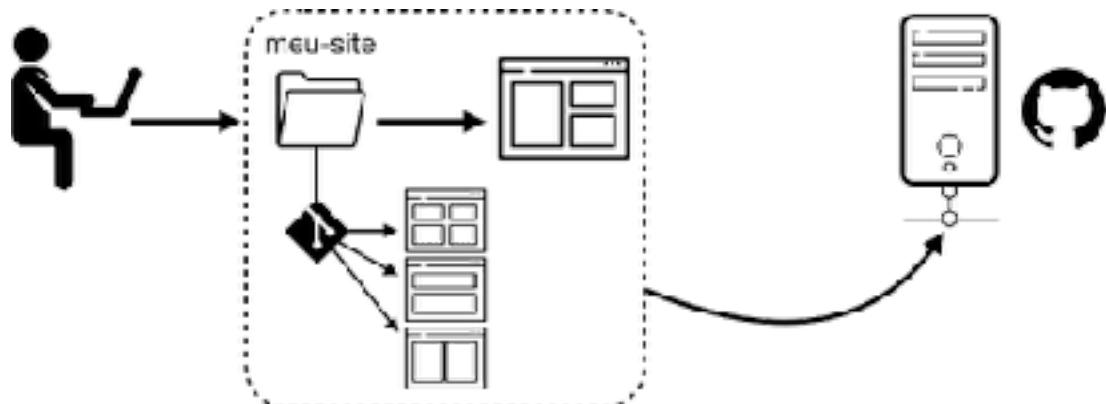


A grande vantagem de usar um **sistema de controle de versões** como o **Git** é poder voltar a qualquer momento para qualquer versão anterior do projeto de forma imediata, tudo 100% transparente para o programador.

Como você já deve ter percebido, os repositórios locais resolvem o problema que apontamos na **situação 1**, mas se o nosso PC quebrar (**situação 2**) ou se quisermos mostrar o projeto para outra pessoa (**situação 3**), ainda não é possível usando esse tipo de sistema, mas não se desespere agora! Continue lendo...

Repositório Remoto

Para solucionar os problemas levantados nas duas últimas situações desse capítulo, vamos precisar de um lugar na nuvem para guardar nossos repositórios locais. E é aí que entra o nosso segundo grande personagem: o **GitHub**.



Criado em 2008 por quatro amigos, o **GitHub** é um serviço que nos permite criar um **repositório remoto** na nuvem para guardar nossos projetos e versionamentos. Ele não é a única opção que existe no mercado (ainda temos o **GitLab**, o **Bitbucket**, e muitos outros) mas provavelmente é a mais popular, tanto que hoje pertence à gigante **Microsoft**, que comprou o serviço em 2018 por 7.5 bilhões de dólares.

Com o tempo, o GitHub começou a ganhar funcionalidades extras, que foram transformando o serviço em uma grande **rede social para programadores**. Além de guardarmos nossos códigos nos servidores, podemos nos comunicar com outros desenvolvedores e até mesmo colaborar com outros projetos que estão disponíveis publicamente para todos.



Ao colocar nossos códigos na nuvem, automaticamente resolvemos o problema da **situação 2**. E um recurso chamado **GitHub Pages** vai permitir a hospedagem gratuita de sites simples, que usem HTML + CSS + JS e disponibilize o acesso através de uma URL. Por exemplo, o mini-projeto que construímos no capítulo anterior está disponibilizado no endereço a seguir:

<https://professorguanabara.github.io/projeto-android/>

O código original também está disponibilizado publicamente no endereço abaixo:

<https://github.com/professorguanabara/projeto-android>

Como instalar e usar isso tudo?

Para poder aproveitar todos esses maravilhosos recursos que explicamos até aqui, precisamos instalar dois softwares no nosso computador: o **Git-SCM** e o **GitHub Desktop**. Também precisamos criar um perfil gratuito no site **GitHub**.

Para fazer o download e instalação do **Git**, acesse o site git-scm.com e clique sobre o botão de *download* da versão mais recente e instale o programa.

Nesse site, encontramos as versões para todos os sistemas: Windows, Linux e MacOS.

Para a instalação no Linux, o site mostra os comandos necessários para realizar a instalação via gerenciador de pacotes.

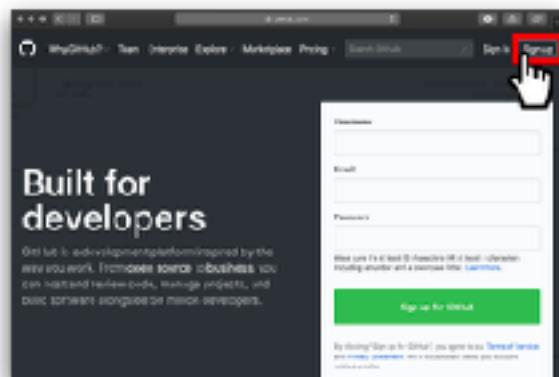


Depois de instalar as duas ferramentas acima, acesse o site github.com e crie o seu perfil clicando no botão **Sign up** indicado na imagem a seguir. Você só vai precisar de um nome de usuário, e-mail de contato e uma senha segura. Um e-mail de verificação será enviado para ativar sua conta.

Se já tiver uma conta criada, clique sobre o botão **Sign in** ao lado.

Para fazer o download e instalação do programa **GitHub Desktop**, acesse o site desktop.github.com e clique sobre o botão de *download* da versão atual e instale.

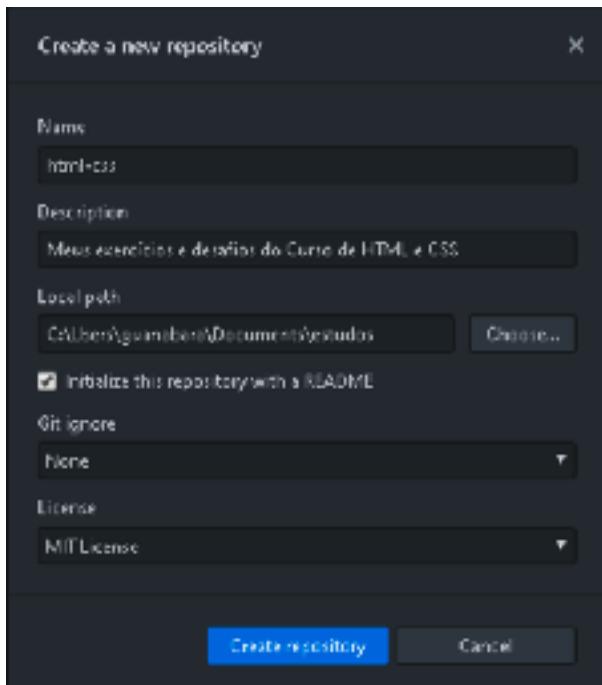
O GitHub Desktop também existe em versões para os sistemas Windows, Linux e MacOS. A versão para Windows só existe para sistemas 64-bits. Em Linux, devemos instalar via repositório.



FAÇA ESCOLHAS SÁBIAS! Tenha muito cuidado na hora de escolher seu nome de usuário. Ele fará parte do seu futuro profissional e pode ser que você precise divulgar seu perfil em uma entrevista de emprego. Fuja daqueles nomes engraçadinhos e agressivos.

Criando o seu primeiro repositório

Abra o **GitHub Desktop** (com conexão ativa à Internet, claro) e faça login na sua conta criada no **GitHub** diretamente na tela do programa. Depois de confirmar a conexão, crie um novo repositório em File > New Repository ou pressione Ctrl+N.



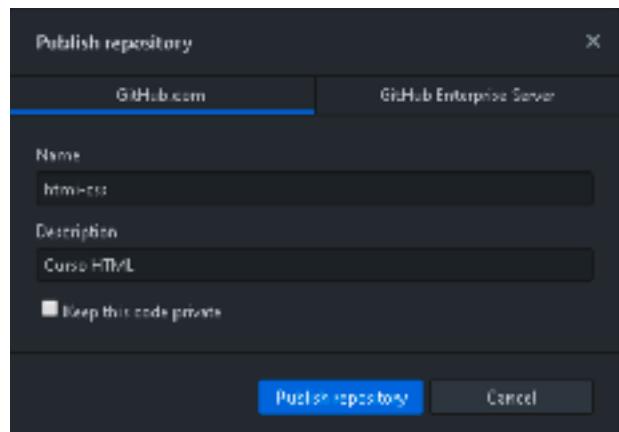
Para começar a versionar os arquivos que criamos durante o curso, vamos criar o repositório de nome `html-css` (mesmo nome da pasta que criamos) dentro da pasta `estudos` que está dentro dos nossos Documentos. Não se esqueça de marcar a opção para criar o arquivo `README.md` e escolher uma licença. No fim, clique em **Create repository**.

Este procedimento vai criar nosso **repositório local** via **Git**.

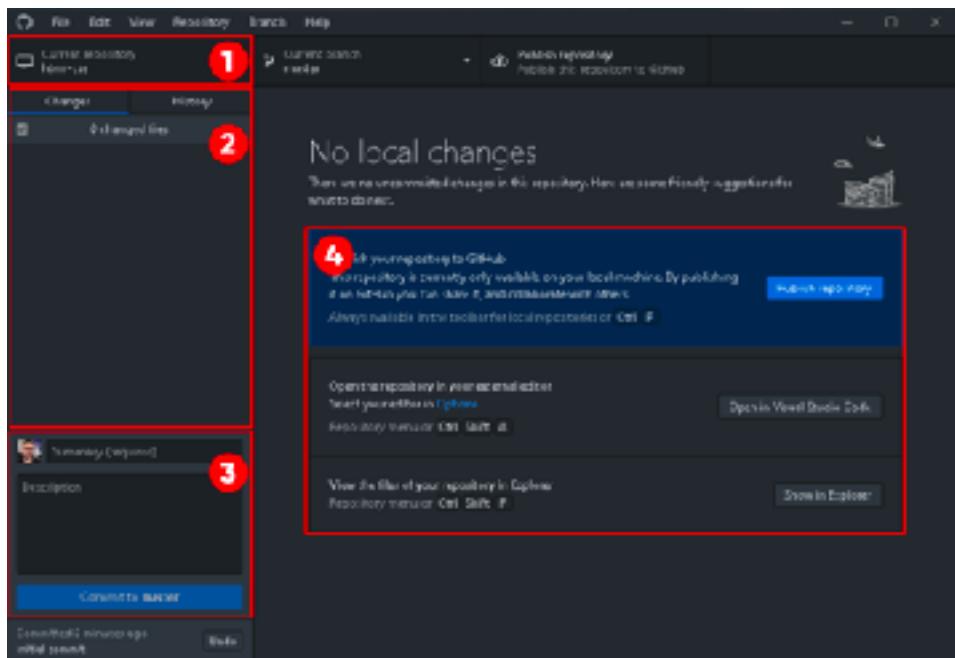
Depois de tudo, clique no botão **Publish Repository** para criar o nosso **repositório remoto** no **GitHub** e todos os arquivos serão enviados para a nuvem.

Assim que solicitar a publicação do repositório, deve escolher se vai deixá-lo **público** ou **privado** (padrão). Desmarque a opção "Keep this Code private", principalmente se deseja utilizar o recurso do **GitHub Pages** e hospedar gratuitamente seu site HTML. Só projetos públicos podem ser hospedados e disponibilizados para outros.

A interface do **GitHub Desktop** é bem simples, mas concentra muitas funcionalidades que unificam recursos do **Git** e do **GitHub** em um só lugar.



MUITO CONFUSO? TEREMOS VÍDEOS! Não se desespere se não estiver conseguindo realizar os procedimentos seguindo apenas esse material escrito. Ele será complementado com uma série de vídeos que estão sendo cuidadosamente criados pra você.



Na **área 1**, escolhemos o repositório ativo no momento. Clicando nessa área, podemos mudar entre todos os repositórios locais que temos no computador.

Já na **área 2**, vão aparecer todas as mudanças detectadas pelo Git nos arquivos que estão na pasta do repositório.

Usaremos a **área 3** sempre que quisermos estabelecer uma versão estável ou submeter uma alteração importante do nosso projeto. Ao colocar uma descrição na caixa **Summary** (resumo) e clicar no botão **Commit** (pode ser entendido como **compromisso** ou como **enviar**). Fazer um *commit* ou “comitar” (jargão técnico em Português) vai criar uma versão **local** do repositório usando o Git.

Depois de comitar um repositório local, podemos realizar uma operação de **Push** (empurrar) que vai transferir todos os códigos para o **GitHub**. O botão *push* vai aparecer na **área 4** logo após um *commit*. Também podemos realizar uma operação dessas clicando no menu Repository > Push ou ainda pressionar Ctrl + P.

Na **área 4**, também vão aparecer botões muito importantes para abrir o projeto usando o Visual Studio Code ou abrir o gerenciador de arquivos do seu sistema com a pasta do projeto.

Habilitando a Hospedagem Grátis

O recurso do **GitHub Pages** não está ativo por padrão em nossos repositórios remotos, mas é muito fácil de habilitar. Isso só pode ser feito diretamente no site do GitHub, individualmente para cada repositório que queremos usar para hospedar projetos HTML.

Acesse o seu perfil no site <https://github.com/username> (substituindo pelo nome do seu usuário, claro) e abra o repositório que quer habilitar a hospedagem. Em seguida, clique sobre a aba **Settings** e role a tela até a área **GitHub Pages**.

A imagem a seguir mostra a tela principal do repositório no site. Observe atentamente as áreas demarcadas, pois elas mostram a posição dos componentes.

The screenshot shows a GitHub repository page for 'professorguanabara/html-css'. The URL 'github.com/professorguanabara/html-css' is highlighted with a red box in the browser's address bar. The 'Settings' tab in the top navigation bar is also highlighted with a red box. The page displays basic repository information: 10 commits, 1 branch, 0 packages, 0 releases, 1 environment, 1 contributor, and the MIT license. Below this, there are sections for branches ('master' is selected), pull requests, and files. A table lists branches: 'desafios' and 'exercícios', both updated 'yesterday'. At the bottom, there are buttons for 'Create new file', 'Upload files', 'Find file', and a green 'Clone or download' button.

Na área de opções, existem duas maneiras de habilitar o GitHub Pages. Na primeira, clique no botão **None** e escolha a opção **master branch**. Com essa maneira, você vai precisar ter um arquivo index.html com o código da página principal que será exibida ao acessar o site. Essa técnica é mais usada quando queremos hospedar um site único e receber uma URL para apresentar o projeto a outra pessoa.

GitHub Pages

[GitHub Pages](#) is designed to host your personal, organization, or project pages from a GitHub repository.

Source

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository.

[Learn more](#).

[None](#) ▾

Theme Chooser

Select a theme to publish your site with a Jekyll theme using the master branch. [Learn more](#).

[Choose a theme](#)

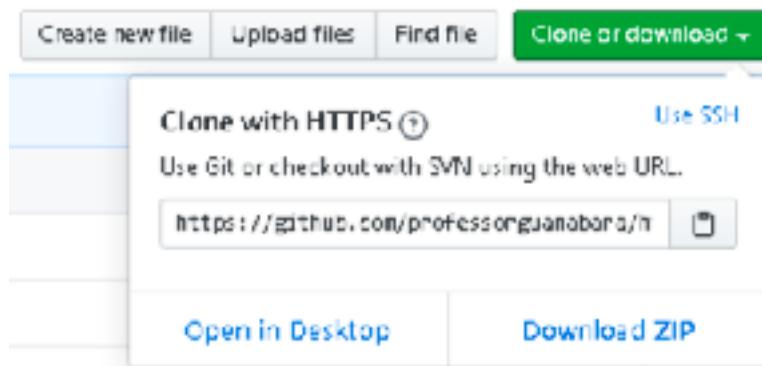
A segunda maneira é clicando sobre o botão **Choose a theme**, que vai permitir escolher um tema para exibir o conteúdo do arquivo README.md como uma página. Essa segunda técnica é mais usada quando vamos hospedar vários projetos em um mesmo repositório ou quando vamos guardar informações sobre um projeto que não é um site, como download de instaladores e coisas do tipo.



IMPORTANTE: O GitHub pode ser usado para guardar códigos em todas as linguagens, incluindo o PHP. Porém, o recurso **GitHub Pages** não vai servir para hospedar o projeto, gerando link para acesso à sua execução. Ele só serve para hospedar HTML + CSS + JavaScript.

Clonando um repositório

Outra operação essencial para quem vai usar o **GitHub** para hospedar repositórios remotos é **clonar** um projeto. Para isso, devemos acessar a URL do repositório no site github.com e procurar o botão **Clone or download** ao lado direito da tela.



Ao clicar sobre o botão verde, escolha a opção **Open in Desktop** e o programa **GitHub Desktop** vai abrir automaticamente e os arquivos serão baixados para o seu computador, criando um repositório local.

Podemos clonar qualquer projeto do **nossa** repositório remoto (público ou privado) ou até mesmo repositórios públicos de **outras pessoas**. Se o repositório for seu, você ainda vai poder atualizá-lo com as alterações que fizer. Já para repositórios de outras pessoas, você não vai poder efetuar alterações e efetuar operações de *push*, a não ser que o proprietário autorize (através das *Pull Requests*).

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_AreHm4dm7ZULPAmadvNhH6vk9oNZA

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

DevWeb

Capítulo 19

Imagens de Fundo

Nos capítulos 6 e 11 do nosso material, aprendemos técnicas para usar imagens como parte do nosso conteúdo. Agora, vamos aprender como usar algumas imagens para complementar visualmente um site, aplicando-as aos fundos dos nossos elementos HTML utilizando estilos. Também vamos ver como manter essas imagens adaptáveis ao tamanho do navegador dos nossos visitantes. Vamos nessa?



Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso

a todo o conteúdo e usá-lo com seus alunos. Porém todos os que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof.**

Gustavo Guanabara e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.



O fundo não precisa ter só cor

Em capítulos anteriores, aprendemos a aplicar cores sólidas ou degradê em qualquer elemento de caixa. Mas você não precisa se limitar a essa possibilidade e pode aplicar imagens ao fundo de qualquer elemento exibido visualmente em HTML. Vamos a um exemplo simples com três `<div>` no corpo de um documento:

```
<body>
  <div class="quadrado" id="q1"></div>
  <div class="quadrado" id="q2"></div>
  <div class="quadrado" id="q3"></div>
</body>
```

Agora vamos criar a configuração de estilo base para toda `<div>` que possui a classe quadrado:

```
<head>
  <style>
    div.quadrado {
      display: inline-block;
      margin: 5px;
      width: 300px;
      height: 300px;
      background-color: #lightgray;
      border: 2px solid black;
      border-radius: 20px;
    }
  </style>
</head>
```

Com esse código aplicado, temos o seguinte resultado na tela:



Viu? Resultado simples, três quadrados exatamente iguais na tela.



NÃO ENTENDEU? Se você criou o código acima, mas não obteve o resultado apresentado, é sinal de que talvez você esteja tentando correr demais com seu aprendizado. Se eu puder te dar um conselho, volte ao capítulo 16 e faça todos os exercícios.

Agora vamos adicionar algumas configurações individualmente a cada quadrado, usando os identificadores diferentes entre eles:

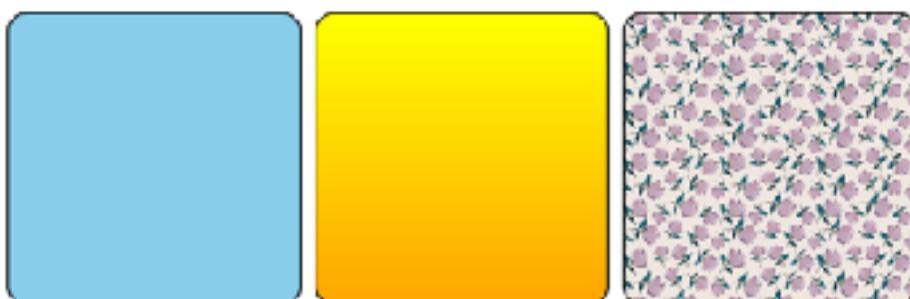
```
div#q1 {  
    background-color: #skyblue;  
}  
  
div#q2 {  
    background-image: linear-gradient(to bottom, #yellow, #orange);  
}  
  
div#q3 {  
    background-image: url('imagens/pattern003.png');  
}
```

Note que na definição da <div> que tem o id com o valor q1, usamos uma cor sólida, já para q2, usamos um preenchimento linear. Já na caixa q3, vamos aplicar uma imagem de fundo através de um endereço informado pela função url().



CADÊ AS IMAGENS? Se você quer as imagens que usaremos nesse capítulo, acesse nosso repositório em <https://github.com/gustavoguanabara/html-css/tree/master/exercicios/ex022>. Lá você vai encontrar a pasta com várias imagens que usaremos.

Ao adicionar as linhas acima ao código, o resultado já muda consideravelmente:



De forma geral, essas são as três maneiras mais simples de preencher uma caixa em HTML: cor sólida, degradê ou imagem de fundo.

E não se esqueça: o **body** é uma grande caixa

Esse recurso também pode ser usado ao <body> do seu site e o papel de parede será aplicado. Vamos considerar um exemplo bem simples que usa uma imagem disponível no nosso repositório.

```
<style>
  body {
    background-image: url('https://gustavoguanabara.github.io/html-css/imagens/mascote.png');
  }
</style>
```

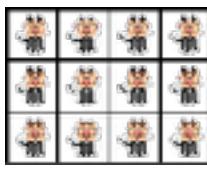
Analisando o código acima, estamos usando uma `url()` externa, já que a imagem estará em outro servidor (no caso, no servidor do **GitHub**). A imagem em questão é razoavelmente pequena, então não vai ser suficiente para cobrir toda a tela. Sendo assim, o resultado será o seguinte:



Personalizando a aplicação do background

Quando o tamanho da caixa é maior que o tamanho da imagem, por padrão, a imagem será **repetida** nos dois eixos (*eixo x* e *eixo y*) quantas vezes for necessário para cobrir a extensão da caixa contêiner.

É possível alterar esse comportamento usando a propriedade `background-repeat`, que aceita os seguintes valores:

background-repeat: repeat;	
background-repeat: no-repeat;	
background-repeat: repeat-x;	
background-repeat: repeat-y;	

Note que ao usar o no-repeat, isso **não obriga** o navegador a aumentar ou diminuir o tamanho da imagem para caber no tamanho da caixa. Para realizar essa adaptação, devemos usar outra propriedade, que veremos mais adiante.

Além de escolher o nível de repetição do *background*, também podemos mudar a **posição de referência** de início das repetições. Por padrão, é considerado o canto esquerdo superior (left top), mas podemos ter várias opções. Use a imagem abaixo como referência sempre que precisar definir a posição do fundo com a propriedade background-position no seu código.

left top	center top	right top
left center	center center	right center
left bottom	center bottom	right bottom

Outra coisa que podemos fazer é **redimensionar** a imagem para forçá-la a caber na caixa. Por padrão, nenhum redimensionamento será aplicado, e a imagem será exibida do seu tamanho natural. Porém, podemos usar a propriedade background-size para alterar esse comportamento.

Os valores aceitos por essa propriedade são:

auto	(padrão) a imagem de fundo será aplicada em seu tamanho original.
[length]px [length]%	Redimensiona a largura da imagem e faz a altura se adaptar automaticamente. Podemos também informar as duas dimensões na sequência ou também usar valores percentuais.
cover	Muda o tamanho da imagem para que ela seja sempre totalmente exibida na tela, sem nenhum corte.
contain	Redimensiona a imagem para que ela cubra o contêiner, mesmo que para isso ocorram alguns eventuais cortes.



SÓ CONSIGO DEMONSTRAR NA PRÁTICA. Essas propriedades de personalização de imagens de fundo precisam de demonstração prática. Nesse exato momento, já gravei vários vídeos ensinando o uso de cada uma delas. Em breve você verá, basta acompanhar o canal **Curso em Vídeo** no **YouTube**

A última propriedade que podemos configurar é o **vínculo** (*attachment*) da imagem de fundo com o resto do documento, principalmente se o conteúdo for maior do que a altura da página e seja necessário vazar uma rolagem vertical.

A propriedade `background-attachment` aceita os valores:

scroll	(padrão) a imagem de fundo vai rolar junto com o conteúdo.
fixed	A imagem de fundo vai ficar fixada enquanto o conteúdo vai sendo rolado.

Simplificando as coisas

Assim como já vimos várias vezes em nosso material, existe também a possibilidade de usar uma *shorthand* para simplificar o uso de propriedades que se apliquem ao fundo de uma caixa. A propriedade abreviada `background` pode ser declarada agrupando as seguintes configurações:

- `background-color`
- `background-image`
- `background-position`
- `background-repeat`
- `background-attachment`

Sendo assim, no lugar de usar:

```
background-color: black;  
background-image: url('imagens/wallpaper002.jpg');  
background-position: center center;  
background-repeat: no-repeat;  
background-attachment: fixed;
```

Podemos reunir tudo em uma única declaração:

```
background: black url('imagens/wallpaper002.jpg') center center no-repeat fixed;
```

Centralização vertical em contêineres

Antes de começar a explicar o assunto sobre o qual vamos falar, preciso desabafar: plural de *contêiner* é muito esquisito, não acha? A propósito, o significado de *contêiner* (versão do Inglês *container*) é simples e direto: "aquele que contém coisas".



Aprendemos no **capítulo 16** que existem elementos que podem conter outros elementos. As `<div>` são um exemplo de elemento *contêiner*. Quando queremos centralizar blocos horizontalmente, aprendemos a usar o `margin:auto;` nas folhas de estilo. Mas como fazer a centralização vertical?

No **capítulo 17**, onde criamos o nosso primeiro mini-projeto, tivemos que arrumar uma maneira de centralizar e redimensionar um vídeo dentro de um *contêiner*. Agora vou te mostrar uma outra técnica.

Vamos começar criando uma hierarquia simples entre dois blocos:

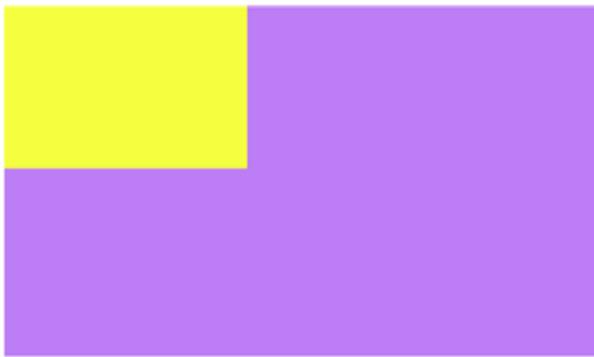
```
<div id="fora">
  <div id="dentro">
    </div>
</div>
```

Agora vamos criar as configurações personalizadas para cada `<div>` dentro da área de `<style>` na área da cabeça `<head>` do código:

```
<style>
  div#fora {
    height: 96vh;
    background-color: #BD7DF5;
  }

  div#dentro {
    height: 200px;
    width: 300px;
    background-color: #F5FF40;
  }
</style>
```

O resultado disso será algo como:



O nosso objetivo aqui é deixar o retângulo interno exatamente no meio do retângulo externo. Vamos começar configurando o posicionamento de cada uma. O retângulo externo terá posicionamento relativo, enquanto o interno terá posicionamento absoluto.

```
div#fora {
  position: relative;
}

div#dentro {
  position: absolute;
}
```

Não crie um novo seletor, apenas adicione as declarações que estou indicando. Qualquer dúvida, assista o último vídeo relativo ao **capítulo 19** do curso.

Quando definimos um bloco com posicionamento absoluto, podemos personalizar a sua posição exata através das propriedades `left` e `top`. Como queremos posicionamento centralizado, vamos configurar os dois na metade do contêiner:

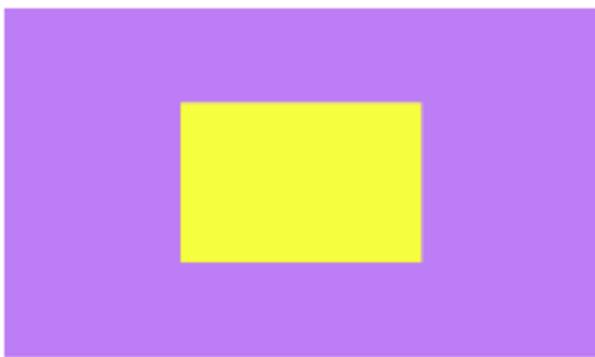
```
div#dentro {  
    position: absolute;  
  
    left: 50%;  
    top: 50%;  
  
    ***  
}
```

Porém, infelizmente, o resultado não será exatamente o que esperamos. Mas não se desespere! Nem tudo está perdido!



Note que, pelas linhas pontilhadas, o retângulo interno foi realmente posicionado a 50% da tela, mas pelo canto superior esquerdo da caixa. Vamos realizar uma transformação e mover o retângulo interno para a esquerda e para cima, para que ele fique efetivamente centralizado.

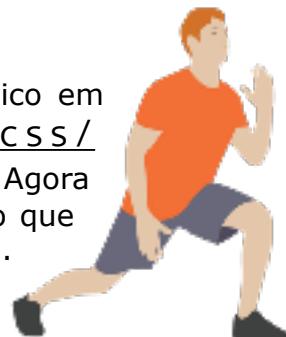
```
div#dentro {  
    ...  
    transform: translate(-50%, -50%);  
    ...  
}
```



E está feito! Essa é apenas uma das técnicas de centralização de conteúdo, mas as outras requerem aprender outros conceitos mais aprofundados das folhas de estilo, como as caixas flexíveis (*Flexbox*).

Hora de exercitar

Chegou a hora de acessar o endereço do nosso repositório público em <https://gustavoguanabara.github.io/html-css/desafios/> e executar os **exercícios 022** 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.**



Tenho desafios pra você!

Lá no repositório, além do material em PDF e dos códigos dos exercícios 100% disponíveis, também disponibilizamos alguns **desafios** que devem ser resolvidos. Esses desafios não incluem o código original e você deve tentar chegar à resposta sem copiar nenhum código.

Com todo o conteúdo que vimos até essa aula, você já pode resolver o **desafio d011**. Acesse o repositório público, abra a área do curso de HTML+CSS e clique no link de acesso aos desafios. Manda ver! Só não fica pedindo a resposta! Você consegue resolver isso sozinho(a)!



Repositório em: <https://gustavoguanabara.github.io>

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_AreHm4d1AnJ_jJtV29RFxnPHDuk9o

Teste seus conhecimentos

Terminou de ler esse capítulo e já acompanhou todos os vídeos e referências externas que indicamos? Pois agora, responda a essas 10 perguntas objetivas e marque em cada uma delas a única opção verdadeira. Aí sim, você vai poder comprovar que realmente entendeu o conteúdo.



Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

Questão...

- A
- B
- C
- D

DevWeb

Capítulo 20

Mini-projeto Cordel

Vamos começar agora mais um mini-projeto onde será possível aplicar conceitos importantes que vimos até o momento. Será um projeto rápido, mas vai incluir o efeito *parallax* às imagens de fundo e uma adaptação dinâmica do tamanho da fonte de acordo com o tamanho da tela. E o conteúdo que vamos dar vida, transformando em um site é um belo cordel.



Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso

a todo o conteúdo e usá-los com seus alunos. Porém todos os que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof.**

Gustavo Guanabara e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.



A cultura da Literatura de Cordel

O Brasil é realmente um país múltiplo quando se fala em cultura, principalmente quando falamos de cultura popular. Uma dessas manifestações populares é a **literatura de cordel**. Ela tem esse nome pois os pequenos folhetos com os poemas são impressos ficam presos em cordas para a exposição.



A ideia dos folhetos não surgiu no Brasil e sim em Portugal, mas foi aqui que ganharam uma forma peculiar ao estampar as capas usando *xilogravuras*, formato muito popular em cidades do nordeste brasileiro.

Outra característica dos folhetos de cordel é ter poesias escritas em forma de versos que podem ser recitados de forma melodiosa e cadenciada, geralmente acompanhadas de uma música tocada com instrumentos tipicamente nordestinos.



UM DOS MAIS FAMOSOS: Quer conhecer um pouco mais sobre os cordéis? Acompanhe aqui um dos mais populares, que versa sobre o a história de Virgulino Ferreira, vulgo cangaceiro Lampião.

Causos de Cordel: <https://youtu.be/56zDjaM1iLg>

E se você está lendo essa introdução e se perguntando “*o que tem a ver cordel com HTML e CSS?*”, tenho um recado sincero pra você: a cultura faz parte da nossa identidade como seres humanos. Nunca desmereça a cultura, amplie seus conhecimentos e valorize sempre os movimentos nacionais. Você não perdeu tempo lendo tudo isso, você ganhou culturalmente. De nada!

Levantei esse assunto aqui, pois escolhi um poema em forma de cordel de **Milton Duarte** chamado “*Cordel Moderno - Tecnologia do agora*” que fala de Tecnologia (olha aí o assunto!) de uma maneira bem simpática e atual. Esse será o conteúdo do site que vamos criar.

Download do pacote básico

Os arquivos básicos que usaremos para criar nosso site já estão disponíveis no nosso repositório oficial do GitHub. Comece acessando o endereço a seguir:

<https://github.com/gustavoguanabara/html-css/tree/master/desafios/d012>

Agora faça o download do arquivo pacote-d012.zip e descompacte os arquivos, colocando todo o conteúdo na sua pasta de desafios, dentro da sub-pasta d012.

Dentro desse arquivo compactado, você vai encontrar:

- Duas imagens que aplicaremos ao site, adicionando o efeito parallax a elas
- O texto original, criado por Milton Duarte e disponível no [Recanto das Letras](#)

O projeto pronto

Para ver o projeto desse capítulo funcionando completamente, basta acessar o endereço <https://professorguanabara.github.io/projeto-cordel/> no seu navegador, mas não vale ficar olhando e copiando o código, pois esse é mais um desafio.

Acesse, role por todo o conteúdo, aumente e diminua o tamanho da janela (principalmente na largura) e veja como ele vai se comportar.

Organizando o conteúdo em seções

Agora vamos começar a organizar o conteúdo em seções, que serão formatadas para intercalar áreas brancas com áreas de imagens. Você pode usar qualquer técnica, mas a que eu achei mais coerente semanticamente foi dividir tudo em `<section>` com um ou dois parágrafos e usando quebras de linha para organizar as estrofes.

```
<section>
  <p>
    Estou ficando cansado<br>
    Da tal tecnologia<br>
    Só se fala por e-mail<br>
    Mensagem curta e fria<br>
    Twitter e Facebook<br>
    Antes que eu caduque<br>
    Vou dizer tudo em poesia.
  </p>
</section>
```

Como criar um texto que aumenta ou diminui dinamicamente?

Nesse exercício também usamos um recurso que faz com que o texto aumente ou diminua de acordo com o tamanho (mais especificamente a largura) da janela do navegador.

A dica aqui é usar as medidas relativas das CSS, como vh e vw. Basicamente vh significa **viewport height** e vw significa **viewport width**.

Mas que *diabos* é **viewport**?, você pode perguntar. Bem, de forma resumida a *viewport* é a área visível de uma janela. No início de um documento HTML, dentro da área <head>, nós fizemos a configuração básica do tamanho de uma viewport usando uma tag <meta> desde o nosso primeiro exercício.



```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Na **linha 5** do nosso código base, definimos que a viewport terá a largura máxima baseada na largura do dispositivo e que vai usar uma escala (zoom) inicial da tela para o valor padrão 1.0 (zoom de 100%).

Sendo assim, vamos considerar uma tela de celular popular como o **Samsung Galaxy S9**, que tem uma viewport de 360x740 pixels ou um **iPhone X** com 375x812 pixels. Isso significa que - com o celular na posição vertical - a largura da tela é de 300 e poucos pixels disponíveis para exibir o nosso site.

Para usar uma medida fixa do tamanho de um título, podemos declarar:

```
h1 {
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    font-size: 5em;
}
```

O que vai fazer com que a fonte fique **5 vezes** maior que o seu tamanho base (aproximadamente 16px, dependendo do navegador). E esse tamanho será mantido, tanto para telas pequenas quanto para telas grandes.

Mas e se quisermos adaptar o tamanho da fonte ao tamanho da tela? Aí podemos mudar um pouco a declaração acima e usar:

```
h1 {
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    font-size: 10vw;
}
```

Isso vai significar que o tamanho da fonte será adaptado para que ocupe 10% da largura da viewport, ou seja, em uma tela do Galaxy, com 360px de largura, o tamanho da fonte será 36px. Mas basta deitar o celular para que a largura da tela vá para 740px, fazendo com que a fonte seja mudada para tamanho 74px, o que significa 10% da nova dimensão.

Esse recurso fica ainda mais visível quando estamos em um computador e modificamos o tamanho da janela do navegador. Faça seus testes usando medidas de fontes configuradas com `em` e depois mude para `vw`.

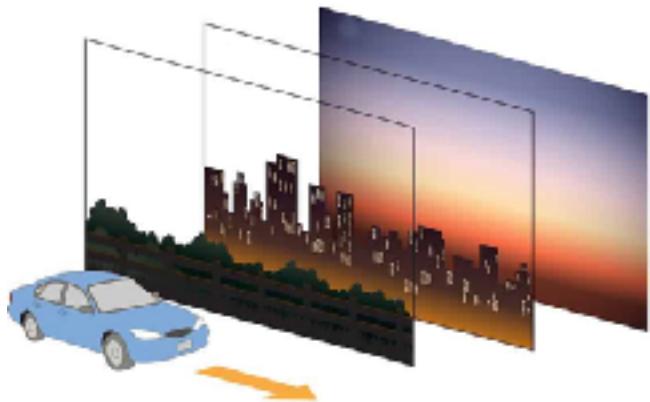


USE COM MODERAÇÃO! Esse recurso de tamanho dinâmico de fontes deve ser usado apenas em pontos muito específicos do nosso site. Não exagere e aplique isso em todos os textos, pois seu site vai acabar difícil de adaptar a telas. Se quiser um recurso melhor, estudaremos as Media Queries mais para frente.

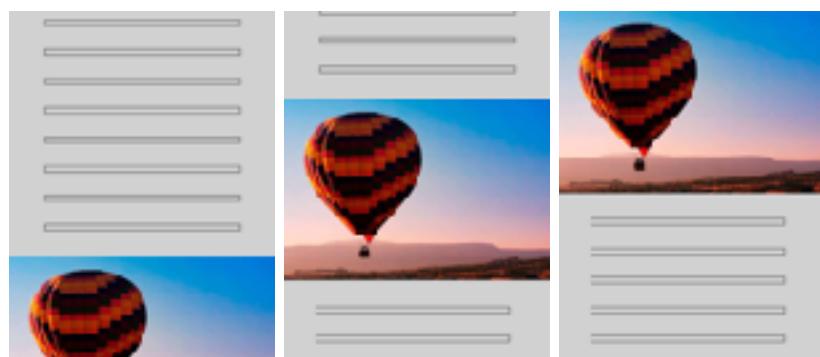
O que é o efeito parallax?

Em vários pontos durante esse capítulo, eu citei esse efeito e provavelmente você percebeu um comportamento diferente nas imagens do site do projeto que te apresentei anteriormente. Elas se movem de um jeito diferente. Vou te explicar esse fundamento com uma situação do dia-a-dia.

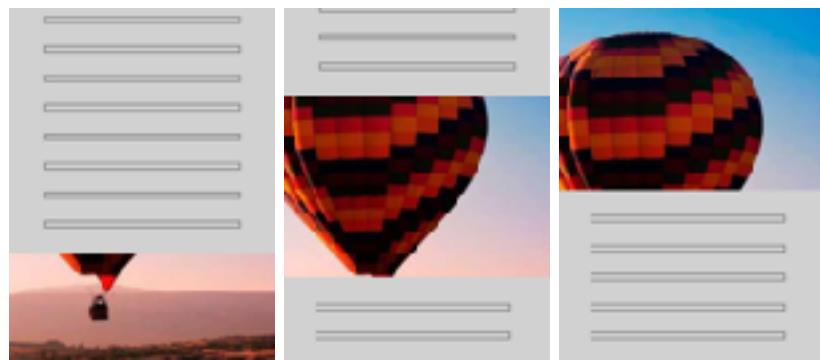
Você já deve ter percebido que quando estamos em movimento (em um carro, ônibus ou trem) e olhamos para a paisagem na janela lateral, aquilo que está mais perto de você parece se mover muito mais rápido e o que está mais longe tem um deslocamento muito mais lento? Pois isso é o que chamamos de **efeito parallax**.



Podemos simular esse mesmo princípio em sites criando diferenças de posicionamentos na rolagem do conteúdo. Em um site comum, imagine que temos um bloco de texto seguido de uma imagem, seguida novamente por um texto. Se não fizermos nenhuma configuração adicional, a rolagem da tela acontecerá da seguinte maneira:



Note que a imagem do balão vai subindo juntamente com os blocos de texto. Agora, quando aplicamos o **efeito parallax**, a rolagem fica um pouco diferente:



Agora o conteúdo vai sendo rolado pela tela, enquanto a imagem vai sendo revelada em pedaços diferentes, dando a impressão de que o bloco com texto está mais perto e a imagem está mais distante.

Talvez o impacto visual gerado por esse efeito não seja 100% percebido enquanto eu tento te explicá-lo em forma de um texto em uma página, mas com certeza você percebeu quando abriu o site do projeto. E se você ainda não abriu, aí vai outra oportunidade. Acesse o link a seguir, ou abra o app de câmera e aponte seu celular para o código QR:

<https://professorguanabara.github.io/projeto-cordel/>



Para obter esse efeito, vamos dar um id à `<section>` que vai conter a imagem e realizar as seguintes configurações:

```
section#img01 {  
    background-image: url('imagens/background001.jpg');  
    background-repeat: no-repeat;  
    background-position: right center;  
    background-size: cover;  
    background-attachment: fixed;  
}
```

Todas as declarações acima foram vistas no capítulo anterior, quando falamos sobre aplicação de imagens em *background*.

Tenho desafios pra você!

Lá no repositório, além do material em PDF e dos códigos dos exercícios 100% disponíveis, também disponibilizamos alguns **desafios** que devem ser resolvidos. Esses desafios não incluem o código original e você deve tentar chegar à resposta sem copiar nenhum código.

Com todo o conteúdo que vimos até essa aula, você já pode resolver o **desafio d012**. Acesse o repositório público, abra a área do curso de HTML+CSS e clique no link de acesso aos desafios. Manda ver! Só não fica pedindo a resposta! Você consegue resolver isso sozinho(a)!



Repositório em: <https://gustavoguanabara.github.io>

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