Texto

Descrição gerada automaticamente

Interface gráfica do usuário

Descrição gerada automaticamente

1. Marcação do Primeiro texto

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Transcrição do Vídeo

[00:00] Vamos começar os nossos desafios, então.

[00:03] Na aula anterior, apresentamos como seria a página desejada, o que vamos construir ao longo deste curso.

[00:11] Nesta primeira aula, nós vamos trabalhar o texto, que é o miolo dessa página.

[00:15] Vamos construir passo a passo esse conteúdo para que você consiga chegar na página desejada.

[00:22] O primeiro passo vai ser trabalhar o parágrafo de texto e os títulos que existem na página.

[00:28] Para isso, existe disponível um arquivo. Tem um [link](https://caelum-online-public.s3.amazonaws.com/1179-html5-css3/01/texto-base.zip) aí para você. É o arquivo “texto base”.

[00:37] Se você abrir esse arquivo, temos o que seria um título e três parágrafos de texto. Usando o próprio Word, nós poderíamos aumentar o tamanho da fonte, deixar o texto em negrito, selecionar alguma coisa de destaque no nosso primeiro parágrafo, como por exemplo o nome “Barbearia Alura” e colocar em negrito. Poderíamos selecionar a missão da barbearia e deixar em itálico. Formatar esse texto de forma a dar destaque para algumas informações específicas.

[01:17] Isso é a forma como trabalhamos essa apresentação do texto no Word.

[01:23] Vamos pensar nisso agora para nossa página, no HTML.

[01:30] A primeira coisa que precisamos fazer usando HTML é que não vamos usar o editor de texto do Word. Vamos usar um editor de código. Eu vou usar o Sublime Text 3. Vocês podem baixar ele na internet. É gratuito, disponível para todos os sistemas operacionais. Podem inclusive usar o Atom, que é outro editor de código, também gratuito e disponível para ser baixado. Ou o Visual Studio Code. Essas são as três recomendações para você usar durante este curso.

[02:18] Eu vou usar o Sublime Text 3. Já tenho ele instalado. Assim que eu abrir o programa, ele me dá uma tela preta, onde vamos escrever todo o nosso código.

[02:31] A primeira coisa que vou fazer é copiar aquele texto do Word e jogar no Sublime Text. Se vocês repararem, toda aquela formatação que nós criamos foi perdida. Este é um editor de código, e o negrito, tamanho de fonte, eles não vêm por padrão. Nós vamos chegar lá.

[03:02] A primeira coisa que precisamos fazer no Sublime Text é salvar este arquivo, pode salvar em uma pasta sua ou na área de trabalho, com o nome index.html. O nome index é padrão para o arquivo principal da sua página. E como nesta página vamos criar somente um arquivo HTML, esse vai ser o nome do nosso arquivo durante todo o projeto. Vamos mexer nesse mesmo arquivo.

[03:32] O .html, diferente do .doc para o Word, diz que este arquivo é uma página web. Salvando este arquivo index.html, se voltarmos na nossa área de trabalho, minimizar tudo, ele agora está com o ícone do Chrome para ser aberto. E se eu der dois cliques ele vai abrir direto no navegador. [04:00] Reparem o que ele fez com o nosso texto. Ele não botou quebra de linha, negrito, itálico, não aumentou a fonte.

[04:14] Se voltarmos para o editor de código, vamos ver que o texto está simples, e não é isso exatamente o que temos, quando vemos uma página web. Uma página web tem o seu conteúdo com HTML. Nós usamos o HTML para fazer uma marcação desse texto. O HTML quer dizer “hyper text markup language”. Uma linguagem de marcação de texto.

[04:40] Para marcarmos esse texto, precisamos usar o que são chamadas de “tags”. O formato de uma tag é um sinal de < um nome qualquer e um sinal de >. Isso é o que fecha uma tag. É o conteúdo que vamos ter para uma tag.

[05:01] E que tags vamos usar neste texto? Por exemplo, na primeira linha temos um título. Vamos usar uma tag chamada “h”, de heading, que é o título do conteúdo.

[05:18] Existem seis níveis de título. Para o primeiro título da página, vamos usar h1. Já na sequência, vamos ter o primeiro parágrafo de texto. O parágrafo de texto é marcado com a tag “p", de paragraph. Vamos fazer isso para o segundo e terceiro parágrafo também.

[05:49] Mas... quando o HTML vai saber onde termina um h1 e começa um parágrafo? Por padrão, ele não vai saber. Nós precisamos informar isso para o nosso navegador. Como precisamos marcar o conteúdo, eu marco o início e o final. Marquei o início do título com h1 e para marcar o final, além do sinal de <, preciso colocar uma barra. Assim ele fecha o conteúdo.

[06:25] Preciso também nos parágrafos botar p. Reparem que não estou digitando o p. Eu coloco a barra, e como estamos usando um editor de código, ele já faz o favor de complementar para mim o que está faltando.

[06:45] Com isso aqui, nós já temos um título e três parágrafos.

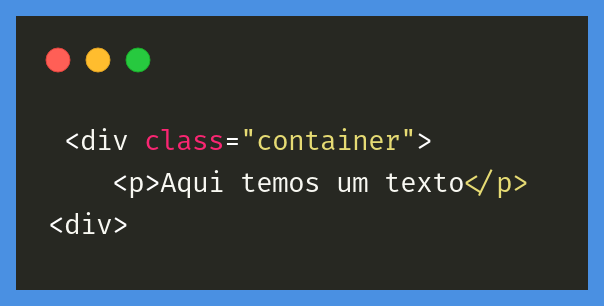
[06:50] Vamos salvar o arquivo. Ou, se você quiser ser mais rápido, ctrl+s. Vamos voltar no navegador e recarregar a página. Ou também com o atalho ctrl+r.

[07:06] Reparem nas modificações que já foram feitas. Temos o conteúdo quebrando a linha, porque o marcamos com os parágrafos. Temos um título com uma fonte maior e em negrito, porque usamos a tag do h1, e já começamos a ter o nosso conteúdo estruturado.

[07:27] Isso é o primeiro passo para modificar e estruturar nosso conteúdo.

[07:35] Agora, vamos evoluir o conteúdo marcando todas as informações do meio do texto que colocamos em destaque.

**Centralizando um elemento com CSS**



**Gabs Ferreira**

10/11/2015

**COMPARTILHE**

Se você desenvolve pra web é mais do que comum querermos centralizar elementos na tela, certo? Vamos imaginar que temos a seguinte *div*:

<div class="container">

<p>Aqui temos um texto</p>

<div>

Se você quiser centralizar o **texto dentro da div**, pode fazer:

.container { text-align: center; }

Você pode ver esse código funcionando aqui **[aqui](http://jsfiddle.net/gabsferreira/jyk4Lzsq/" \t "_blank)**.

Esse código funciona para centralizar qualquer elemento que seja *inline* (você pode ver quais são [inline](https://developer.mozilla.org/en/docs/Web/HTML/Inline_elemente" \t "_blank)) ou *inline-block*.

Ah, mas e se eu não quiser centralizar o conteúdo, mas sim a *div* na tela? Desse jeito que fizemos, não vai funcionar. Nesse caso, temos que fazer basicamente duas coisas:

dar um tamanho para a *div* e setar as margens esquerda e direita para *auto*:

.container {

width: 700px;

margin-left: auto;

margin-right: auto;

}

Você pode ver um demo desse código [aqui](https://jsfiddle.net/gabsferreira/e68mxk2k/).

Esse código funciona para qualquer elemento **[block](https://developer.mozilla.org/en/docs/Web/HTML/Block-level_elements" \t "_blank)**.

# Guia de Unidades no CSS





**Paulo Scalercio**

Atualizado em 30/09/2021

**COMPARTILHE**



Quando começamos a lidar com desenvolvimento web, mais especificamente **HTML e CSS**, é bastante comum que fiquemos presos às ferramentas que já conhecemos e temos familiaridade.

Porém, isso pode se tornar um problema devido ao grande crescimento da Web e, com isso, o surgimento de novos problemas e consequentemente novas soluções. Esse é uma situação corriqueira quando lidamos com unidades de medidas no **CSS**, pela grande variedade, acabamos deixando de lado parte das existentes e não utilizamos, de fato, todo o poder que temos na mão.

**Confira neste artigo:**

* [Medidas Absolutas](https://www.alura.com.br/artigos/guia-de-unidades-no-css#medidas-absolutas)
* [Medidas Relativas](https://www.alura.com.br/artigos/guia-de-unidades-no-css#medidas-relativas)
* [Medidas absolutas no CSS](https://www.alura.com.br/artigos/guia-de-unidades-no-css#medidas-absolutas-no-css)
* [Medidas Relativas no CSS](https://www.alura.com.br/artigos/guia-de-unidades-no-css#medidas-relativas-no-css)
* [Alerta!!](https://www.alura.com.br/artigos/guia-de-unidades-no-css#alerta!!)
* [Conclusão](https://www.alura.com.br/artigos/guia-de-unidades-no-css#conclusao)

Existem algumas unidades que provavelmente você já está acostumado, como o famoso pixel! Nesse post, abordaremos todas as unidades de medidas presentes atualmente e como elas podem nos ajudar durante sua jornada como desenvolvedor web. Antes de prosseguirmos, precisamos entender qual a diferença entre **medida absoluta** e **medida relativa**.

## Medidas Absolutas

Essas são as mais comuns que vemos no dia a dia. São medidas que não estão referenciadas a qualquer outra unidade, ou seja, não dependem de um valor de referência. São unidades de medidas definidas pela física, como o **píxel**, centímetro, metro, etc...

Essas medidas são fixas e não mudam de acordo com as especificações do dispositivo. Esse tipo de medida é indicada para quando conhecemos perfeitamente as características físicas e as configurações das mídias onde serão exibidos nossos projetos.

*Já lançarei um spoiler aqui para você, será que no "mundo real" conheceremos todas as informações de todos os dispositivos? Segura esse pensamento!*

## Medidas Relativas

Essas são as que normalmente não estamos habituados. Essas medidas são calculadas tendo como base uma outra unidade de medida definida, como por exemplo

**em** e o **rem** (veremos mais sobre essas duas medidas no decorrer do post). O uso delas é mais apropriado para que possamos fazer ajustes em diferentes dispositivos garantindo um layout consistente e fluido em diversas mídias.

Devido ao fato de que essas medidas são calculadas pelo browser baseando-se em outra unidade, elas tendem a ser bastante flexíveis. Ou seja, podemos ter resultados diferentes de acordo com o ambiente.

Para começarmos a caminhar pelas medidas existentes, optei por começar pelas medidas absolutas, uma vez que estamos, normalmente, mais familiarizados com essas. Como diria meu amigo Flávio Almeida, bem começado, metade feito!

## Medidas absolutas no CSS

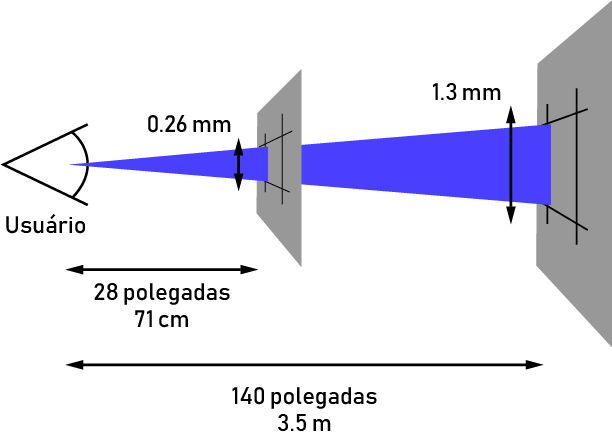
### **Píxels (px)**

Provavelmente você já conhece ou ouviu falar desse rapaz chamado Píxel. Píxel nada mais é do que os pequenos pontinhos luminosos da tela do seu monitor, celular, televisão, etc... Logo, o píxel é o menor elemento em um dispositivo de exibição!

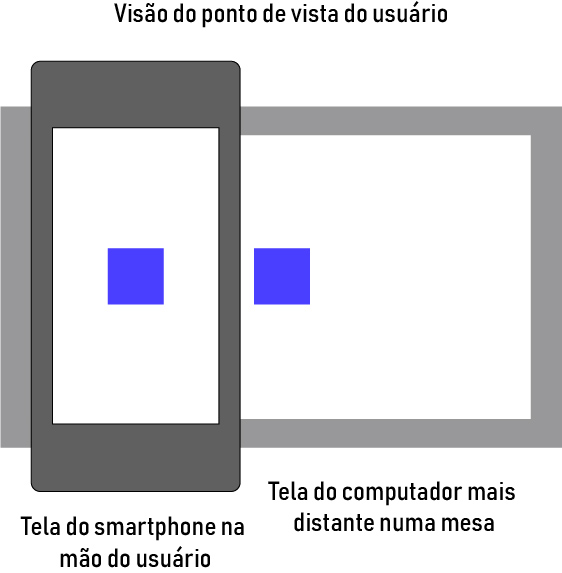
Essa é uma medida bastante famosa para os web designers, grande parte dos desenvolvedores web usam o píxel como unidade principal de seus projetos.

Um detalhe que poucos conhecem é que na verdade, o píxel do CSS **NÃO** é realmente um píxel da tela do dispositivo (hardware), e sim o que chamamos de **[píxel de referência](https://alistapart.com/article/a-pixel-identity-crisis/" \t "_blank)** que geralmente é maior do que o píxel real. O que acaba por torná-lo numa medida abstrata onde é necessário controlar o mapeamento desse píxel de referência para o píxel do hardware (acontece por debaixo dos panos!).

A definição de píxel de referência no **CSS** é o ângulo visual(0.0213deg) de um píxel em um dispositivo com a densidade de 96dpi a uma distancia de um braço do leitor (28 polegadas), veja na imagem abaixo:



O benefício desse píxel de referência é que ele leva a proximidade da tela em consideração, ao usarmos um celular que seguramos próximos de nós, o píxel de referência terá o tamanho semelhante ao de um monitor mais distante de nós, por exemplo.



Portanto, não existe esse papo de que o píxel é "perfeito". Só se estivermos desenvolvendo um site para um mesmo dispositivo, com um mesmo tamanho de tela e que usa o mesmo navegador, mas sabemos que não é assim que funciona no mundo lá fora!

Um ponto interessante de se comentar é que recentemente o **[boostrap 4](https://www.casadocodigo.com.br/products/livro-bootstrap4)** deixou de utilizar **PX** e migrou para **REM** , além disso, o uso do píxel nos dá a sensação de que esse é igual ao píxel do hardware, o que pode gerar confusão para novos desenvolvedores.

Um dos aspectos mais importantes para esses Web Designers é a **escalabilidade e adaptabilidade de um layout**, ou seja, a medida em que as unidades aumentam de uma maneira previsível e razoável, seu layout deve ser capaz de se adequar a essas mudanças.

De um lado, a ideia de se manter o mesmo aspecto (tamanho) em diversos dispositivos pode parecer atrativa (cuidado com o píxel de referência!), mas do outro lado, temos consequências negativas quando estamos lidando com dispositivos de baixa resolução (blurry rendering).



### **Points (pt)**

A **próxima unidade é Point**. Definitivamente essa unidade é mais conhecida pelos designers, principalmente os que estudam tipografia.

Essa medida é geralmente utilizada em propriedades relacionadas a fonte do seu projeto. Sua abreviação se dá com a marcação de **pt** e seu uso não é tão comum, você provavelmente verá essa unidade muito raramente.

Geralmente espera-se que essa medida seja utilizada em folhas de estilo para **impressões**, quando se precisa ter certeza do tamanho da fonte utilizada. Não é recomendada para a estilização em tela!

### **in (inches/polegadas)**

**Polegada ou inch** em inglês é mais uma unidade de medida que conhecemos do mundo das medidas absolutas - geralmente vemos elas quando queremos comprar uma nova TV ou monitor , mas essa unidade também existe no mundo Web.

Apesar de existirem, elas não costumam ser utilizadas em projetos, uma vez que não existem um uso prático para elas (podemos atingir os mesmos resultados utilizando outras unidades)

### **Centímetro e Milímetro (cm / mm)**

Nós brasileiros, que adotamos o sistema métrico, conhecemos bem essas duas medidas, que são bastante utilizadas no dia a dia. Apesar de bastante comuns, tanto centímetro e milímetro são pouco usadas no CSS. Assim como o **pt**, o uso dessas duas é esperado para folhas de estilo para **impressões** (medidas mais precisas), evitando que elas sejam aplicadas para exibições em tela.

### **Paica (pc)**

Também uma unidade pouco usada no mundo web, a **Paica** também vem para o CSS sendo herdada da tipografia. Por não ser uma unidade amplamente conhecida, ela acaba sendo fadada ao esquecimento, mas é sempre importante conhecermos todas as ferramentas que estão à nossa disposição. A relação entre as unidades absolutas é:

*1 in = 2,54cm = 25,4mm = 72pt = 6pc*

## Medidas Relativas no CSS

### **Ems (em)**

Nossa primeira unidade relativa é bastante famosa no mundo **CSS**. Dificilmente você achará algum navegador que não tenha suporte para essa medida, que está presente desde os primórdios. Até para o IE, nós teríamos que usar a versão abaixo da 3.0 para que tivéssemos algum problema.

Esse definitivamente é um dos pontos que fazem o **em** tão popular. O segundo ponto, com certeza se dá a facilidade de criar layouts fluídos e responsivos.

*Mas como funciona esse tal de****em****?Essa unidade muda para os elementos filhos de acordo com o tamanho da fonte (font-size) do elemento pai, então vamos lá. Digamos que temos o seguinte html, me permitindo a licença poética de utilizar a tag style:*

<style>

#div{

font-size: 16px;

}

#filho{

font-size: 2em;

}

</style>

<div id="pai">

div pai

<div id="filho">

div filho

</div>

</div>

Acima, temos uma div pai onde estou definindo um font-size de 16px, dentro dessa div, temos uma única div filha. Como havia mencionado, o tamanho definido para a fonte impactará no **em** dos elementos filhos.

Nesse nosso caso, para a div mais interna (id=filho), **1em será igual a 16px**, seguindo a lógica, **2em será igual a 32px** e assim por diante. Podemos colocar valores como 1.5 também! Nesse nosso caso, **1.5em será igual a 24px** Quando expressamos tamanhos como margin, padding utilizando **em**, isso significa que eles serão relativos ao tamanho da fonte do elemento pai.

Portanto, de acordo com o tamanho da fonte utilizada em determinado elemento, os elementos filhos serão redimensionados de forma a obedecer a referência a esse tamanho de fonte!

Uma técnica bastante utilizada consiste justamente em fazer uso desse poder do **em** componentizando nossos elementos. A ideia é que a alteração do tamanho da fonte do elemento pai faça com que todo o componente se modifique e redimensione baseando-se nesse novo valor. Você pode dar uma olhada em um exemplo nesse fiddle:

Caso não consiga visualizar o jsfiddle, o link é [**esse**](https://jsfiddle.net/r0od5zbe/3/)

Apesar de divertido, o motivo de utilizarmos essa técnica não é para que o usuário tenha um slider e altere o tamanho da fonte rsrs. Mas sim para facilitar a manutenção do componente como um todo, sem ter que sofrer alterando valores de todas as partes do componente. Bem legal né?

O último ponto que devemos nos atentar ao usar o **em** é que quando usamos essa medida, nós temos que considerar o font-size de todos os elementos pai. Por exemplo, se tivéssemos uma terceira div mais interna no nosso exemplo anterior e definirmos o tamanho da fonte para **2em**, nesse caso esses **2em seriam 64px**, uma vez que o font-size do elemento pai foi definido sendo **32px**(2em)! Pegou o pulo do gato?

Isso tende a se complicar quando estamos falando de 5, 6, 7 divs aninhadas, provavelmente não será muito divertido calcular isso! Mas a boa notícia é que temos uma unidade que nos ajuda a resolver esse probleminha.

### Rems (rem, "root em")

O **REM** vem como sucessor do **EM** e ambos compartilham a mesma lógica de funcionamento (font-size), porém a forma de implementação é diferente. Enquanto o **em** está diretamente relacionado ao tamanho da fonte do elemento pai, o **rem** está relacionado com o tamanho da fonte do elemento root (raiz), no caso, a **tag**.

O fato de que o **rem** se relaciona com o **elemento raiz** resolve aquele problema que tínhamos com diversas divs (elementos) aninhados, uma vez que não haverá essa "herança" de tamanhos, lembra?! Ou seja, não precisaremos ter dor de cabeça tendo que realizar cálculos, uma vez que nos baseamos na tag raiz.

Exemplificando, sabemos que a tag html é a tag raiz de todo documento html. Dito isso, se definirmos que o font-size desse elemento será de 18px, então **1rem = 18px**, **2rem = 36px** e assim por diante... Normalmente os browsers especificam o tamanho default da fonte do elemento root (raiz) sendo 16px, então guarde isso no coração! Mesmo essa unidade sendo mais tranquila de se trabalhar, ela não era muito utilizada para design responsivo, o que de primeira pode soar um tanto quanto estranho...

O motivo para isso é o suporte para essa medida. O chrome e o firefox suportavam tranquilamente, assim como o Opera e o Safari, porém, antigamente grande parte dos usuários utilizavam o IE, mais específicamente o IE 8, e esse browser não lidava muito bem com os **rems**, isso fazia com que os desenvolvedores precisassem optar por alguma unidade diferente, em muitos casos, o próprio **em**.

Como disse acima, o valor base é 16px, e isso pode acabar gerando dificuldades para que encontremos alguns tamanhos padrões que costumam ser utilizados. Por exemplo, como faríamos para atingir um tamanho de **10px** utilizando **rem**? Precisamos calcular.

**BASE 16PX**

10px = 0.625rem

12px = 0.75rem

14px = 0.875rem

16px = 1rem

18px = 1.125rem

e assim por diante, realmente não são números muito "amigáveis" ou convenientes Porém, podemos lançar mão de um pequeno truque para nos ajudar (62,5%)

html{

font-size: 62,5%;

}

h1{

font-size: 1.2rem; */\*equivalente a 12px\*/*

}

p{

font-size: 2.4rem; */\*equivalente a 24px\*/*

}

Repare que dessa forma, o valor em píxel será sempre o valor definido em **rem** vezes 10! Fica mais conveniente, concorda?

Apesar de parecer uma boa ideia, devemos ter cuidado com essas abordagens, uma vez que ela forçará que você reescreva todos os font-size do seu site, então tome cuidado!

Existe uma terceira visão sobre isso tudo. Essa solução utiliza px, em e rem de maneira bem definida. A ideia consiste em definir o font-size do elemento root em píxel, módulos utilizando rem e elementos interiores aos módulos utilizando em, facilitando a manipulação do tamanho global que naturalmente escalará o tamanho para os módulos (utilizando rem) e esses por sua vez escalarão os elementos interiores (que utilizam em e referenciam ao elemento pai). Veja um exemplo dessa abordagem. No console será mostrado o tamanho atual da fonte.

Caso não consiga visualizar o jsfiddle, o link é [esse](https://jsfiddle.net/p78usjfe/10/) Perceba como a combinação dessas medidas pode ser benéfica. O limite é só sua imaginação!

### Porcentagem (%)

Apesar de não ser uma unidade de medida, a porcentagem costuma ser bastante utilizada quando falamos de layout responsivo e fluido, por isso, não poderia deixá-la passar.

A porcentagem permite que criemos módulos que sempre vão se readaptar para ocupar a quantidade especificada. Por exemplo, se definirmos um elemento tendo um tamanho de **50%**, independente do dispositivo em questão, esse módulo sempre ocupará metade do espaço que lhe cabe (caso esteja dentro de algum outro elemento).

Veja o exemplo abaixo

Caso não consiga visualizar o jsfiddle, o link é [esse](https://jsfiddle.net/2yxo7puj/10/)

Repare que se alterarmos o tamanho da div container, o elemento interior a ela se redimensionará de forma a sempre ocupar a porcentagem especificada no CSS!

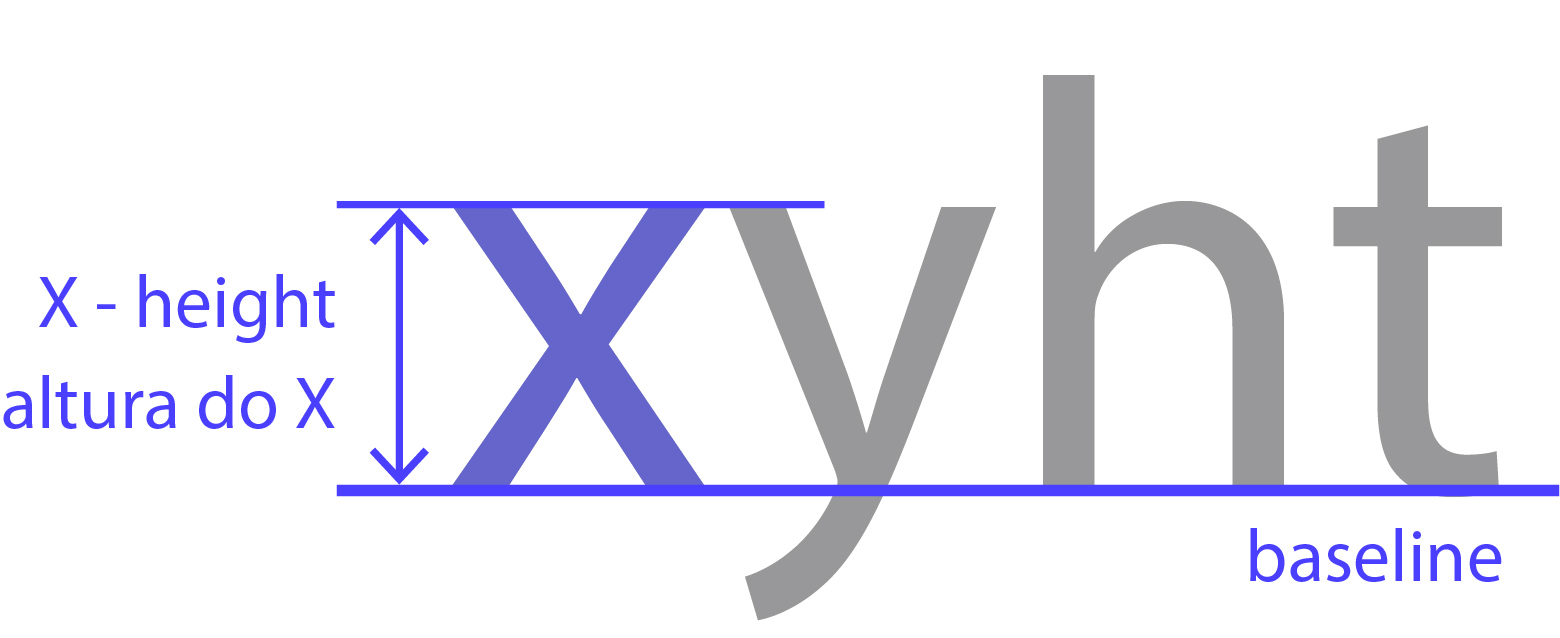
Veja também que o slider está alterando tanto a largura quanto a altura do elemento, mas se alterarmos cada um separadamente, também funcionará! Visto esse exemplo, podemos dizer que a porcentagem tem um comportamento um tanto parecido ao nosso já conhecido **em**, já que ele se relaciona diretamente com o tamanho da propriedade do elemento pai.

Portanto, ao trabalharmos com a porcentagem, temos o mesmo problema que tinhamos com o **em**, quanto mais elementos aninhados, mais complicado será de definirmos exatamente o tamanho, por isso, tenha cuidado quando utilizá-la!

### Ex

Talvez você nunca tenha ouvido falar dessa unidade do CSS, mas ela existe. Diferentemente da forma como a **EM** e a **REM** funcionavam, essa unidade não se relaciona com o tamanho da fonte (font-size), mas com qual fonte está sendo utilizada naquele momento (font-family), mais especificamente ao tamanho do caractere **x** dessa fonte em questão (x-height).

Como o browser sabe esse valor? Esse valor pode vir diretamente com a fonte, o browser pode medir o caractere em caixa baixa (lower case) e se esses dois não funcionarem, o browser estipula um valor de **0.5em** para **1ex**.



Com isso, se quando mudamos o tamanho da fonte (depende do elemento) o **em** e o **rem** mudam, dessa vez, quando alteramos completamente a fonte, o **ex** mudará. O uso dessa unidade está mais presente em ajustes tipográficos, nos dando um controle mais preciso quando o padrão definido para algumas tags não se adéqua corretamente ao nosso layout.

### Ch

Também uma unidade pouco conhecida, o **ch** (character unit) é definida na documentação como sendo a "medida avançada" da largura do caractere zero ("0").

Existe uma discussão antiga onde se debateu bastante sobre essa unidade e o que realmente seria "medida avançada", [você pode acompanhar aqui](http://meyerweb.com/eric/thoughts/2012/05/15/defining-ch/). A ideia é que um elemento com, por exemplo **100ch** de largura poderá comportar uma string de 100 caracteres dessa determinada fonte, caso essa fonte seja monospace (todos os caracteres têm o mesmo tamanho).

É comum acharmos definições que dizem que a frase acima se aplica para qualquer fonte, porém isso está errado. Como mencionei acima, a regra de **1ch = 1 caractere** se aplica apenas se a fonte usada for **monospace** (largura fixa). Fontes com a largura variável, qualquer caractere pode ser mais largo ou menos largo que o zero ("0"), como podemos ver na imagem abaixo:



Como podemos analisar, a tipografia Courier (monospace) obedece a regra acima, porém as outras duas não! O que podemos tirar após observações é que normalmente **1ch** é 20% - 30% mais largo, porém isso não é uma verdade absoluta e deve ser observado para cada fonte que você deseja aplicar. Por isso, tome cuidado com o uso!

### Vw (viewport width)

Essa medida faz parte das medidas mais atuais e do futuro do CSS. Viewport units.

Como escrito no título, **vw** significa viewport width, mas o que é viewport?

Viewport nada mais é que a área visível de uma página web para o seu usuário, essa viewport pode variar de acordo com o dispositivo, sendo menor em celulares e maior em desktops.

Antigamente, quando não existiam tablets e celulares capazes de acessar sites, todas as web pages eram pensadas para a tela de um computador, com tamanho fixo e design estático. Com a chegada desses dispositivos móveis, essas páginas eram grandes demais para serem exibidas nesses aparelhos, o que tornava muito difícil a navegação.

A primeira solução partiu dos browsers desses dispositivos, eles adotavam um comportamento de retirar o zoom de forma que o site inteiro coubesse na tela do aparelho,definitivamente não era o ideal, mas uma solução rápida. No HTML5, foi introduzido uma maneira para que os desenvolvedores conseguissem alterar a viewport através da **tag**, corrigindo esse problema de usabilidade relacionado aos dispositivos móveis, mas isso é assunto para outra postagem!

Voltando para o nosso querido **vw**, essa unidade se relaciona diretamente com a largura da viewport, onde **1vw** representa **1%** do tamanho da largura dessa área visível. A diferença entre **vw** e a **%** é bem semelhante a diferença entre **em** e **rem**, onde a **%** é relativa ao contexto local do elemento e o **vw** é relativo ao tamanho total da largura da viewport do usuário.

### Vh (viewport height)

Essa unidade funciona da mesma forma que o **vw**, porém dessa vez, a referência será a altura e não a largura. Existem diversos exemplos práticos e interessantes de uso dessas duas unidades. Você pode ver alguns usos nesse [link](https://css-tricks.com/fun-viewport-units/), provavelmente mais para frente postarei alguns exemplos bacanas. Me cobrem!

### Vmin (viewport minimun)

Essa unidade também se relaciona com as dimensões da viewport, mas com um porém. Anteriormente quando vimos **vh** e **vw** precisávamos escolher se gostaríamos de nos basear na altura (vh) ou na largura (vw) da viewport.

Diferentemente das anteriores, o **vmin** utilizará como base a menor dimensão da viewport (altura x largura), vamos ao exemplo.

Imagine que estamos trabalhando com uma viewport de 1600px de altura e 900px de largura. Nesse caso, **1vmin** terá o valor de **9px** (1% da menor dimensão!), caso tenhamos **100vmin**, esse será igual a **900px**! Interessante né?

No caso acima, a menor dimensão foi a da largura, porém se tivéssemos 300px para altura e 1400px para largura, nosso valor de referência seria o 300px! Sempre a **menor** dimensão!

### Vmax (viewport maximum)

Seguindo a mesma base lógica da unidade anterior, o **vmax** terá como valor de referência a maior dimensão da viewport. Ou seja, utilizando o mesmo exemplo, se tivermos 1600px de altura e 900px de largura, **1vmax** será equivalente a **16px**!

No segundo exemplo ocorrerá a mesma inversão, tendo 300px para altura e 1400px para largura, **1vmax** será equivalente a **14px**. Dessa vez sempre será a **maior** dimensão!

## Alerta!!

Na versão 9.0 do IE, o **vmin** foi implementado com um nome diferenciado (vm) Algumas versões do IE ainda não oferecem suporte para o **vmax**, nas novas versões isso foi corrigido.

Algumas plataformas como o windows tendem a ser inconsistentes na maneira de como eles contam o tamanho da barra de scroll dos browsers. (diminui o viewport ou não) Sempre verifique o suporte dessas medidas, tenha sempre [esse link](https://caniuse.com/#search=vw) no cinto de utilidades.

## Conclusão

Como podemos perceber, existem várias unidades que podemos utilizar no mundo web, mas sempre surgem aquelas perguntas de quando tenho que utilizar? Qual a melhor? etc...

Não existe resposta certa nem errada para essas perguntas, infelizmente não temos uma regra de ouro para todas as situações. O uso dessas unidades depende de diversos fatores como equipe, preferência, familiaridade e assim por diante.

Entretanto, é importante que você como **web developer** tenha todas essas ferramentas no seu cinto de utilidades já que nunca sabemos quando precisaremos utilizar.

Definitivamente a inclusão das unidades da **viewport** foram positivas para a web quando estamos lidando com layout flexível, cabe a você começar a colocá-las em uso durante seu dia a dia!

No mais, espero de coração que esse post tenha agregado algum conhecimento e facilitado sua jornada rumo ao saber.

Aqui na **Alura** você pode se aprofundar mais no universo front-end, dê uma olhada na [**formação front-end!**](https://www.alura.com.br/formacao-front-end)

**Site:** [**Guia de Unidades no CSS | Alura**](https://www.alura.com.br/artigos/guia-de-unidades-no-css)

## Transcrição

[00:00] Vamos começar os nossos desafios, então.

[00:03] Na aula anterior, apresentamos como seria a página desejada, o que vamos construir ao longo deste curso.

[00:11] Nesta primeira aula, nós vamos trabalhar o texto, que é o miolo dessa página.

[00:15] Vamos construir passo a passo esse conteúdo para que você consiga chegar na página desejada.

[00:22] O primeiro passo vai ser trabalhar o parágrafo de texto e os títulos que existem na página.

[00:28] Para isso, existe disponível um arquivo. Tem um [link](https://caelum-online-public.s3.amazonaws.com/1179-html5-css3/01/texto-base.zip) aí para você. É o arquivo “texto base”.

[00:37] Se você abrir esse arquivo, temos o que seria um título e três parágrafos de texto. Usando o próprio Word, nós poderíamos aumentar o tamanho da fonte, deixar o texto em negrito, selecionar alguma coisa de destaque no nosso primeiro parágrafo, como por exemplo o nome “Barbearia Alura” e colocar em negrito. Poderíamos selecionar a missão da barbearia e deixar em itálico. Formatar esse texto de forma a dar destaque para algumas informações específicas.

[01:17] Isso é a forma como trabalhamos essa apresentação do texto no Word.

[01:23] Vamos pensar nisso agora para nossa página, no HTML.

[01:30] A primeira coisa que precisamos fazer usando HTML é que não vamos usar o editor de texto do Word. Vamos usar um editor de código. Eu vou usar o Sublime Text 3. Vocês podem baixar ele na internet. É gratuito, disponível para todos os sistemas operacionais. Podem inclusive usar o Atom, que é outro editor de código, também gratuito e disponível para ser baixado. Ou o Visual Studio Code. Essas são as três recomendações para você usar durante este curso.

[02:18] Eu vou usar o Sublime Text 3. Já tenho ele instalado. Assim que eu abrir o programa, ele me dá uma tela preta, onde vamos escrever todo o nosso código.

[02:31] A primeira coisa que vou fazer é copiar aquele texto do Word e jogar no Sublime Text. Se vocês repararem, toda aquela formatação que nós criamos foi perdida. Este é um editor de código, e o negrito, tamanho de fonte, eles não vêm por padrão. Nós vamos chegar lá.

[03:02] A primeira coisa que precisamos fazer no Sublime Text é salvar este arquivo, pode salvar em uma pasta sua ou na área de trabalho, com o nome index.html. O nome index é padrão para o arquivo principal da sua página. E como nesta página vamos criar somente um arquivo HTML, esse vai ser o nome do nosso arquivo durante todo o projeto. Vamos mexer nesse mesmo arquivo.

[03:32] O .html, diferente do .doc para o Word, diz que este arquivo é uma página web. Salvando este arquivo index.html, se voltarmos na nossa área de trabalho, minimizar tudo, ele agora está com o ícone do Chrome para ser aberto. E se eu der dois cliques ele vai abrir direto no navegador. [04:00] Reparem o que ele fez com o nosso texto. Ele não botou quebra de linha, negrito, itálico, não aumentou a fonte.

[04:14] Se voltarmos para o editor de código, vamos ver que o texto está simples, e não é isso exatamente o que temos, quando vemos uma página web. Uma página web tem o seu conteúdo com HTML. Nós usamos o HTML para fazer uma marcação desse texto. O HTML quer dizer “hyper text markup language”. Uma linguagem de marcação de texto.

[04:40] Para marcarmos esse texto, precisamos usar o que são chamadas de “tags”. O formato de uma tag é um sinal de < um nome qualquer e um sinal de >. Isso é o que fecha uma tag. É o conteúdo que vamos ter para uma tag.

[05:01] E que tags vamos usar neste texto? Por exemplo, na primeira linha temos um título. Vamos usar uma tag chamada “h”, de heading, que é o título do conteúdo.

[05:18] Existem seis níveis de título. Para o primeiro título da página, vamos usar h1. Já na sequência, vamos ter o primeiro parágrafo de texto. O parágrafo de texto é marcado com a tag “p", de paragraph. Vamos fazer isso para o segundo e terceiro parágrafo também.

[05:49] Mas... quando o HTML vai saber onde termina um h1 e começa um parágrafo? Por padrão, ele não vai saber. Nós precisamos informar isso para o nosso navegador. Como precisamos marcar o conteúdo, eu marco o início e o final. Marquei o início do título com h1 e para marcar o final, além do sinal de <, preciso colocar uma barra. Assim ele fecha o conteúdo.

[06:25] Preciso também nos parágrafos botar p. Reparem que não estou digitando o p. Eu coloco a barra, e como estamos usando um editor de código, ele já faz o favor de complementar para mim o que está faltando.

[06:45] Com isso aqui, nós já temos um título e três parágrafos.

[06:50] Vamos salvar o arquivo. Ou, se você quiser ser mais rápido, ctrl+s. Vamos voltar no navegador e recarregar a página. Ou também com o atalho ctrl+r.

[07:06] Reparem nas modificações que já foram feitas. Temos o conteúdo quebrando a linha, porque o marcamos com os parágrafos. Temos um título com uma fonte maior e em negrito, porque usamos a tag do h1, e já começamos a ter o nosso conteúdo estruturado.

[07:27] Isso é o primeiro passo para modificar e estruturar nosso conteúdo.

[07:35] Agora, vamos evoluir o conteúdo marcando todas as informações do meio do texto que colocamos em destaque.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Melhorando o texto**

[00:00] Na aula passada, configuramos basicamente o texto que vai ser o miolo da nossa página. Mas ele ainda não está perfeito. Já marcamos o conteúdo de um título, de parágrafos, já dividimos esses conteúdos, e hoje nossa página está assim. Um título com uma fonte grande e em negrito e parágrafos separados. Temos o título h1 e todos os parágrafos separados.

[00:35] Só que na nossa página queremos dar destaque para algumas informações, por exemplo, o nome “Barbearia Alura”. Também queremos nossa missão em itálico. Vamos fazer esses detalhes passo a passo.

[00:53] O primeiro deles é transformar Barbearia Alura em um texto de destaque. Geralmente, esse texto de destaque se apresenta em negrito, porque dá uma importância visual e semântica para o conteúdo.

[01:11] Já falamos que as tags servem para marcar o início e o final do conteúdo. Logo, Barbearia Alura precisa estar com uma tag no início e uma no final. Essa tag que serve para marcar a importância de um conteúdo leva o nome “strong”. Em uma tradução, seria “forte”.

[01:40] Quando usamos a palavra strong para marcar a tag e salvamos usando ctrl+s, voltamos para a página e recarregamos, imediatamente vemos que o conteúdo foi marcado com negrito. O negrito é só um detalhe visual sobre essa tag. Ela serve, novamente, para dar uma importância para aquela palavra, para aquele conteúdo.

[02:10] Já que isso foi feito, vamos marcar a linha inteira com itálico. Queremos colocar ênfase no texto.

[02:20] Para usar a tag da ênfase usamos em. Essa tag vai começar no início do texto, dentro do nosso parágrafo, e vai terminar no final do texto, também dentro do parágrafo. Essa é uma tag que vai marcar todo o conteúdo e transformar a linha inteira em itálico.

[02:45] Se salvarmos, voltarmos para o navegador e recarregarmos a página, veremos que essa transformação já foi aplicada.

[02:54] Para complementar, vamos marcar nossa missão com negrito, para dar ainda mais ênfase e marcar esse texto como muito importante.

[03:07] Dentro, antes das aspas, vamos botar a tag strong. E depois das aspas, antes do ponto, vamos fechar a tag.

[03:19] Voltando ao navegador essa aplicação já foi feita.

[03:28] Resumindo, marcamos a Barbearia Alura como um conteúdo importante, transformamos a linha inteira para ter ênfase no nosso texto, e dentro da linha, ainda marcamos uma frase como importante. Vamos reparar que no HTML temos a tag parágrafo, dentro dela a tag de ênfase e dentro dela a tag de strong. Uma tag dentro da outra. Usamos o HTML dessa forma. Marcamos um conteúdo e se dentro tem um conteúdo menor que precisa de outra marcação, ele também é marcado, e dentro disso outro conteúdo ainda menor que precisa de mais uma marcação também recebe uma tag.

[04:19] Usar uma tag dentro da outra não é problema. É como as coisas são feitas, e na leitura faz sentido quando você começa a entender todas essas tags.

[04:32] Nosso próximo desafio é como expandir o texto, como ir além desses conteúdos. Mas para isso, vamos parar por aqui e se ver na próxima aula.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Consolidando o seu Conhecimento**

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Crie o arquivo **index.html**.

2) Copie o conteúdo do texto base, que você baixou no início desta aula, para dentro do arquivo **index.html**.

3) Marque o título do texto e os seus parágrafos, usando as ***tags* HTML** próprias para isso:

<h1>Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a Barbearia Alura traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p>Nossa missão é: "Proporcionar auto-estima e qualidade de vida aos clientes".</p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

4) Destaque algumas informações do texto, deixando-as em **negrito**. Da mesma forma, dê ênfase em alguns pontos do texto, deixando-os em itálico. Use as **tags HTML** próprias para isso:

<h1>Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a <strong>Barbearia Alura</strong> traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

1. **Separando o Conteúdo e informações**

## Transcrição

[00:00] Nas aulas passadas, nós começamos a criar o nosso HTML, a modificar o texto para ele ter uma formatação e o significado que nós queremos. Só que precisamos dar um passo atrás para pensar de forma mais ampla nesse HTML.

[00:17] O que fizemos para que o nosso navegador soubesse que aquele arquivo abriria no navegador? Que seria uma página da web? Simplesmente colocamos .html no nome do arquivo. Só que não é apenas isso que o navegador precisa de informação para renderizar nossa página de forma correta.

[00:41] Neste curso, estamos abordando a versão 5 do HTML. Como o navegador vai saber isso e não colocar a versão 3 ao invés da 5, por exemplo, na nossa página, fazendo ela parar de funcionar?

[00:56] Precisamos estruturar o nosso conteúdo e passar informações para o navegador, para que ele consiga transformar a nossa página e lê-la da forma correta. Para isso, precisamos falar de uma tag muito importante que vai na primeira linha do arquivo: a tag DOCTYPE.

[01:23] Por padrão, escrevemos essa tag com o nome todo em maiúsculas. As outras tags do HTML não precisam seguir esse padrão, podem estar em minúscula ou maiúscula, mas, por padrão, escrevemos o nome dessa tag em maiúsculo para que o navegador e para que todos os desenvolvedores possam ler este padrão.

[01:46] Essa tag tem alguns detalhes. O primeiro deles é que o primeiro caractere dentro dela tem uma exclamação. Essa exclamação serve para que essa tag seja identificada como DOCTYPE. E outra informação que colocamos é a versão do HTML que estamos usando. No caso, estamos usando a versão 5. Ela trouxe várias modificações e modernidades para o código. Uma delas é que não precisamos mais botar a versão 4.0, 3.0 ou 5.0. Se declararmos DOCTYPE html, estamos dizendo para o navegador que estamos usando a última versão disponível do HTML.

[02:37] Como estamos fazendo uma página extremamente moderna, usando o que há de mais novo, a tag html atende aos nossos requisitos.

[02:48] Se salvarmos esse arquivo e voltarmos ao navegador, recarregarmos a página, vamos reparar que nenhuma mudança aconteceu no visual, mas uma mudança muito brusca aconteceu no entendimento do navegador sobre esta página.

[03:06] O HTML ainda tem mais uma tag estrutural, que é a própria tag HTML. Ela é uma tag de conteúdo que serve para marcarmos tudo que é, dentro desta página, o HTML que vai ser renderizado no navegador.

[03:27] Como essa tag é de conteúdo, ela precisa abrir na primeira linha, e na última linha, a última informação vai ser o fechamento dessa tag /html.

[03:39] A tag do DOCTYPE não precisa ser fechada. Ela é uma tag de informação, não de conteúdo. E a tag HTML, por ser uma tag de conteúdo precisa marcar tudo que está dentro. Só que fica sempre muito difícil de ler o texto se ele estiver todo na linha da esquerda. Para isso, vamos usar a indentação e com o tab vamos jogar todo esse conteúdo para a frente.

[04:07] Agora fica muito fácil lermos o que está dentro do HTML, porque está tudo espaçado.

[04:18] Conseguimos organizar o nosso conteúdo para que ele seja melhor entendido pelo navegador e que esteja estruturado corretamente. É importante destacar que os navegadores modernos, como o Google Chrome, como o Firefox, Microsoft Edge tentam te ajudar se você deixa de acrescentar uma informação, por exemplo.

[04:47] Nossa página funcionou quando não colocamos a tag HTML ou a DOCTYPE, mas por ser um padrão de projeto, é extremamente importante que sempre adicionemos as tags. Mesmo que o navegador hoje não deixe de rodar a página por falta de informação, não sabemos como vai ser o dia de amanhã e como os navegadores vão se comportar. É importante que nossa página esteja correta independente do que o navegador está fazendo.

[05:14] Na próxima aula vamos aprender como melhorar ainda mais essa marcação, como lidar com acentuações, título da página, etc.

Interface gráfica do usuário, Texto, Aplicativo, Email, Teams

Descrição gerada automaticamente

## Transcrição

[00:00] Na aula passada, nós criamos a estrutura do nosso HTML. Usamos a tag DOCTYPE, a tag HTML.

[00:06] Agora, vamos evoluir, passando mais informações para o navegador. O navegador precisa de informações para renderizar nossa página corretamente. Por exemplo, se o sistema operacional do seu telefone, do seu computador ou até mesmo do seu navegador está usando o idioma inglês, você não vai conseguir exibir os acentos da página.

[00:27] No nosso caso, no meu computador, como tudo está em português, os acentos aparecem perfeitamente. Pode ser que isso não aconteça na sua página.

[00:36] Como podemos resolver isso? Ou para que independente da configuração do meu usuário, minha página esteja correta? Precisamos passar a informação corretamente. Para isso, criamos a tag meta.

[00:50] A tag meta é escrita da seguinte forma dentro do nosso HTML: meta charset=”UTF8” entre os sinais < e >.

[01:11] Vamos quebrar essa tag passo a passo para entendermos. O nome da tag é meta, ela passa informações para o navegador. Dentro dela, reparem que diferente do h1, que é uma tag de conteúdo, colocamos uma propriedade dentro dela.

[01:30] Temos o atributo charset, que é o conjunto de caracteres, o characters set, que é o dicionário que estamos escolhendo. E escolhemos o dicionário UTF8. Ele tem as linguagens Unicode, ASCII, mas o mais importante que você deve saber é que ela tem todos os caracteres que são usados na maioria das línguas da Europa, da América Central, América do Norte, América do Sul. A maioria das linguagens com que trabalhamos no dia a dia.

[02:02] Com esses caracteres, vamos resolver o problema do acento. Ele tem ç, crase, acentuação de ~, qualquer uma que usamos na língua portuguesa.

[02:14] Na nossa página, recarregando, não vai fazer diferença nenhuma no meu computador. Pode ser que no seu ele tenha consertado os acentos.

[02:25] Também precisamos dizer para o navegador qual é a linguagem da minha página. Muitas vezes estamos usando o navegador e ele se oferece para traduzir uma página. Por que ele faz isso? Ele entende que uma página está, por exemplo, em inglês, e que seu sistema operacional está em português, então te oferece essa vantagem.

[02:46] Além dele ler o conteúdo, nós costumamos informar qual é a linguagem da página. Fazemos isso dentro da tag HTML. Vamos adicionar a propriedade lang, de language, idioma, e dentro vamos usar PT-BR, que é português do Brasil.

[03:08] Com isso, temos a página sendo entendida em português. Se algum usuário que não entende português estiver lendo nossa página, o navegador vai oferecer a opção de traduzir.

[03:21] Temos tudo que precisamos para que o conteúdo da nossa página seja lido perfeitamente.

[03:27] Além disso, toda vez que estamos navegando, costumamos olhar para a aba do navegador para saber o nome da página, e a nossa hoje exibe o nome “index.html”. Vamos consertar.

[03:42] Para fazer isso, usamos uma tag chamada “title”, que é o título da página. Vamos usar o nome “Barbearia Alura”. E salvamos a página.

[04:02] Quando voltarmos ao navegador e recarregarmos, o título está lá.

[04:07] É importante passar todas essas informações para o navegador para que nossa página seja lida e exibida corretamente.

[04:16] Por hoje é só. Na próxima aula vamos ver como quebrar ainda mais esse conteúdo e fazer nosso código ficar menos complexo. Te vejo lá.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Teams

Descrição gerada automaticamente

## Transcrição

[00:00] Na aula passada, demos ao navegador mais informações sobre o idioma da nossa página e consertamos o título que estava na aba do navegador.

[00:07] Mas se repararmos no nosso conteúdo, a tag meta e title estão perdidas no meio do que seria o que iríamos exibir na nossa página. Reparem que se olharmos no navegador, vamos ver todo o nosso conteúdo, mas as tags não estão exibidas, porque uma coisa é o que estamos exibindo na página dentro da janela branca do navegador e outra coisa são as informações que estamos passando para ele.

[00:37] Dentro do HTML existe essa divisão estrutural. Como ela é feita?

[00:43] O HTML é dividido em duas partes. A primeira delas é o head, e a segunda é o body. Vamos pensar o seguinte, traduzindo os nomes: head é cabeça e body é corpo. Se pensarmos em um documento, a cabeça são as informações que estamos passando para o navegador e o corpo são as informações que queremos exibir na página.

[01:14] A tag meta é uma informação que quero passar para o navegador. Essa informação vai dentro do head. A tag title é uma informação que quero passar para dentro do navegador. Também vai dentro do head. E toda a informação que quero exibir, todo o texto que já criei vai ser jogado dentro do body.

[01:39] Reparem que para ficar correto o código, ele precisa ser bem indentado. É o que eu falei na aula sobre colocar o espaçamento correto na estrutura do navegador. Para isso, vou jogar os parágrafos para a frente.

[01:57] Conseguimos visivelmente entender que os elementos estão dentro do body. E conseguimos entender visivelmente que os outros elementos estão dentro do head. É assim que fazemos para criar a estrutura do HTML e para que ele esteja correto na hora de ser apresentado ao navegador.

[02:18] As informações que estamos passando para o navegador ficam no head e as informações que queremos exibir na nossa página ficam no body. Com isso, concluímos a estrutura da nossa página. Vamos evoluir colocando mais conteúdo, mexendo nesse conteúdo e melhorando a página que temos até o momento.

Interface gráfica do usuário, Texto, Aplicativo, Email, Teams

Descrição gerada automaticamente

**Consolidando o seu conhecimento**

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Defina a estrutura básica do HTML **index.html**, não esquecendo de definir a linguagem da página:

<!DOCTYPE html>

<html lang="pt-br">

<h1>Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a <strong>Barbearia Alura</strong> traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

</html>

2) Passe as informações de renderização da página para o navegador e defina o seu título:

<!DOCTYPE html>

<html lang="pt-br">

<meta charset="UTF-8">

<title>Barbearia Alura</title>

<h1>Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a <strong>Barbearia Alura</strong> traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

</html>

3) Por fim, separe as informações e conteúdo da página:

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<title>Barbearia Alura</title>

</head>

<body>

<h1>Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a <strong>Barbearia Alura</strong> traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

</body>

</html>

Texto

Descrição gerada automaticamente

1. Trabalhando com CSS

## Transcrição

[00:00] Na aula passada nós mexemos na estrutura da página. Nesta, o objetivo é mudar a apresentação visual dos textos que temos. Até agora, só vimos uma forma de fazer isso, que é mudar a tag.

[00:13] A tag do parágrafo tem essa apresentação simples, com esse tamanho de fonte, essa configuração, e vimos que se usarmos uma tag de título, por exemplo, do h1, o tamanho da fonte fica maior. Mas as tags têm um valor, um propósito, um significado, e **para mexer na apresentação visual dos elementos**, não usamos elas. **Usamos o CSS**.

[00:39] O CSS, que em uma tradução seria folha de estilo em cascata, é a forma como mexemos em cada um dos elementos granularmente, em cada um dos pontos desses elementos para podermos apresentar o nosso site do modo como quisermos.

[00:55] Vamos passo a passo mexer nessas fontes para começar a entender o CSS.

[01:03] A primeira coisa que quero fazer é: na linha onde temos a missão, quero aumentar o tamanho da fonte. Vamos lá.

[01:16] No nosso Sublime Text vamos encontrar o parágrafo desejado, onde ainda temos uma tag do strong e uma tag do em, onde fazemos o negrito e a ênfase, e na tag do parágrafo quero aumentar o tamanho da fonte. Por que na tag do parágrafo? Porque quero que seja aplicado em todo o conteúdo que está ali dentro. Por isso que quando falei “folha de estilo em cascata”, é isso. Quando vamos em um elemento anterior, ele reflete para todos os elementos que estão abaixo.

[01:47] Nesse parágrafo vou adicionar a propriedade style, que é o estilo do nosso elemento. Dentro do style e entre aspas vou inserir o que quero alterar. Nesse caso, quero alterar o tamanho da fonte. Para fazer isso, sempre em inglês, tamanho da fonte, **font-size**.

[02:09] Como escrevemos isso?

[02:22] O tamanho da fonte padrão no navegador é medido em pixels. Ocasionalmente o navegador adiciona o tamanho da fonte inicial como **16 pixels**. O que estamos vendo neste momento no navegador é que essas fontes têm 16 pixels. Eu quero mudar a fonte da nossa missão para 20 pixels, por isso 20px ali em cima.

[02:48] Se eu salvar isso e voltar no navegador, a minha fonte já está com outro tamanho na hora da apresentação.

[03:04] Vamos recapitular. Eu adicionei uma propriedade chamada style, coloquei um =, que é uma atribuição para esse conteúdo. Font size, tamanho da fonte, 20 pixels. Reparem que o Sublime Text coloriu de forma completamente diferente. É por isso que é importante, na hora de criarmos algum conteúdo, quando estivermos realmente escrevendo códigos, usarmos um editor de códigos programado para isso. Ele nos ajuda com esses pequenos detalhes, e quando algo está errado ele nos alerta.

[03:44] Por exemplo, se eu der um espaço no font size ele vai perder a marcação visual da apresentação daquela propriedade. Assim sabemos que está errado.

[03:57] Agora que isso está funcionando, quero alinhar todo esse conteúdo ao centro. Só que eu estou fazendo a divisão de quatro tags, tenho a tag do título e tenho três parágrafos. Para alinhar um elemento ao centro, uso uma propriedade chamada text align.

[04:24] Vou começar fazendo isso com o título. Nele também vou adicionar a propriedade style com o conteúdo text align e o valor center.

[04:37] Ou seja, quero alinhar ao centro, centralizar meu conteúdo. Se recarregarmos, meu título está alinhado ao centro.

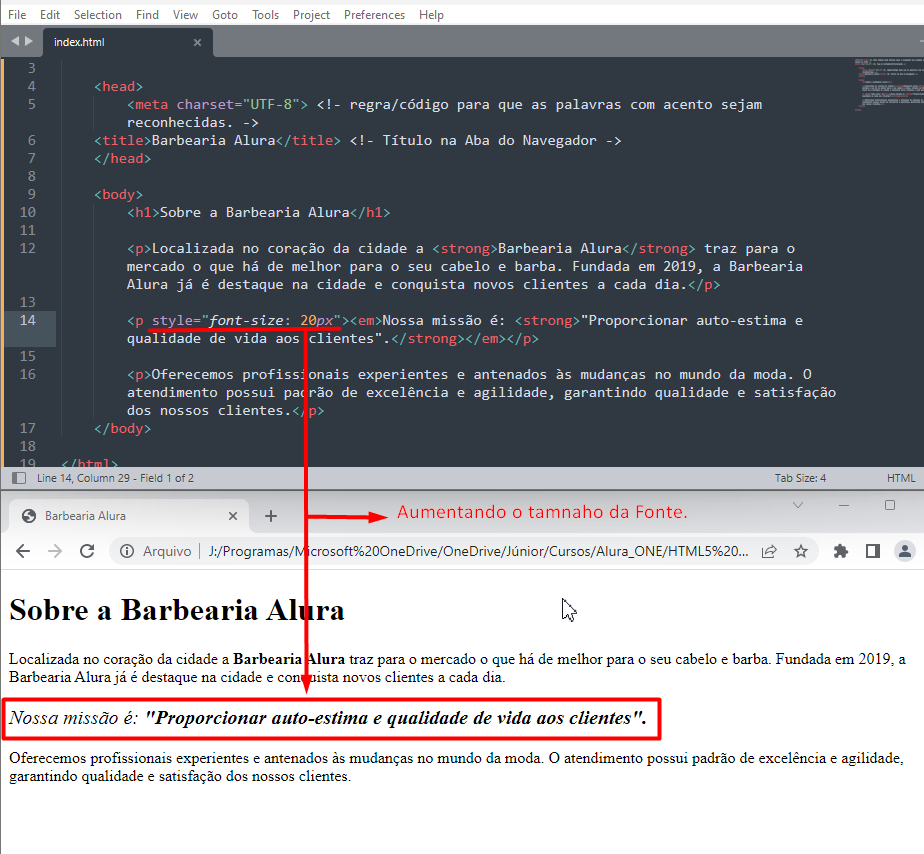
[04:46] Quero fazer isso agora em todos os meus elementos. Vou somente copiar aquilo, colar na linha do parágrafo, colar na linha do último parágrafo, e agora tenho um conflito, porque preciso adicionar duas informações no meu CSS. A primeira é o tamanho da fonte e a segunda é o alinhamento do texto.

[05:19] Para fazer essa divisão entre duas apresentações, dois itens que estamos apresentando, colocamos um ; e assim conseguimos fazer a separação.

[05:37] Reparem que todos os meus elementos têm a propriedade style, o valor text align center, e o meu elemento da missão tem o font size 20 pixels. Vamos salvar e ver como isso ficou no navegador.

[05:57] Esse é o primeiro passo para começarmos a alterar o CSS, a mexer em cada um dos pequenos pontos e das possibilidades que temos.

[06:13] Nesta aula, começamos a mexer na apresentação dos textos, no alinhamento deles e no tamanho da fonte. Na próxima, vamos ver como fazer isso em grandes quantidades de texto e de forma muito mais organizada. Te espero lá.



Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

No **vídeo a seguir**, vamos descobrir que **existem três formas de configurarmos o CSS**. A que fizemos até agora foi **a primeira delas**, a chamada **CSS inline**. Ou seja, na linha onde temos nossa tag, adicionamos a propriedade do CSS. **A segunda** é incluindo a estilização em cada **tag na propriedade style**, no mesmo arquivo html.

**A terceira** forma é através de um **arquivo externo**. Para que a estilização seja aplicada de forma correta, **fique atento**:

### Ao nome do arquivo e sua extensão

Geralmente, chamamos o arquivo que contém os estilos de **style.css**. Observe que este arquivo possui a extensão **.css**.

### Link do CSS no head

**Para utilizar a estilização do arquivo style.css no index.html, precisamos incluir no head o seguinte código:**

<link rel="stylesheet" href="style.css">

Veja o código completo do index.html com o style.css:

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<title>Barbearia Alura</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

</body>

</html>

Observação: É necessário referenciar o arquivo style.css no head do index.html, como mostra o código acima.

Sabendo dessas recomendações, vamos realizar essa etapa passo a passo, junto com o instrutor?

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

## Transcrição

[00:00] Na aula passada, vimos como dividir o CSS e como temos vários níveis para implementar o CSS no nosso HTML. Nessa aula, vamos avançar no CSS, entendendo como podemos colorir nossos elementos no nosso site.

[00:17] A primeira coisa é entender o que queremos mudar. Quero que toda a área ao fundo do texto seja uma área cinza. Quero ter um fundo cinza. E que a cor do texto seja um pouco diferente.

[00:36] A primeira coisa que vamos fazer é mexer em um dos elementos. Eu quero começar mexendo no h1. Para mexer nele, tenho a propriedade style. Quero mexer também na cor de fundo deste elemento. Para mexer na cor de fundo, vou adicionar um ; e vou adicionar a propriedade background. Background é o que define tudo que será o fundo do elemento. Podemos colocar cor, imagem, várias propriedades. Depois vamos falar mais sobre isso. Por enquanto quero mudar só a cor.

[01:16] Para isso, vamos usar a linguagem hexadecimal. Vou fazer um vídeo extra sobre como a cor se comporta no CSS, que tipos de linguagens podemos usar para marcar uma cor. Mas isso fica para depois. Por enquanto, vamos usar a definição da cor cinza, que vai ser “cccccc”. 6 letras c’s.

[01:47] Se salvarmos isso e voltarmos à nossa página, nosso background do título já está cinza. Já atingimos o primeiro objetivo.

[02:00] Mas eu não disse que quero tudo tendo um fundo cinza? Vamos voltar então ao nosso HTML, copiar esse background cccccc e adicionar em todos os parágrafos. Para adicionar esse background em todos os parágrafos, eu poderia colocar linha a linha ou no nosso CSS, onde já temos a descrição do parágrafo. Vou adicionar um ; vou quebrar a linha para ficar mais organizado e vou colocar a mesma definição de background.

[02:34] Quando salvarmos esse arquivo e voltarmos ao navegador, veremos as alterações.

[02:40] Reparem que nesse momento as alterações são só de cor de fundo do elemento. Mas não quero que fique quebrado.

[02:51] Como funciona a visualização nesse momento? Temos a divisão entre os elementos, e essa divisão, esse espaço entre um item e outro, no momento não queremos mexer. Queremos só que o fundo seja cinza.

[03:11] Vamos voltar ao HTML para entender um pouco mais a estrutura. Nela, temos a tag h1 e as tags de parágrafo. São as tags que alterei até agora. Se pensarmos em níveis hierárquicos, todas elas estão dentro da tag body. Ou seja, a tag body é pai de todas essas tags. É quem envolve todas elas.

[03:42] Se eu quero que um fundo seja extra essas tags, ou seja, que ele envolva todas as tags, preciso mudar essa descrição da tag h1 e parágrafo para a tag do body. Podemos fazer isso.

[04:00] A primeira coisa que vou fazer é retirar a descrição. Também vou retirar a descrição do background no parágrafo. E no meu arquivo de CSS vou adicionar a tag body, que é a nova tag em que estou colocando o estilo. E aí vamos colar de volta aquela definição de background e a cor de fundo.

[04:31] Quando salvamos isso e voltamos ao navegador, temos uma alteração. Todo o nosso body passou a ter a cor de fundo. Ou seja, o elemento superior, que envolvia todos, agora tem essa propriedade.

[04:51] Uma coisa importante a se falar sobre o CSS é que idealmente, em um bom padrão de projeto, nosso CSS corresponde à mesma estrutura do HTML. Como o body vem antes do parágrafo, vamos também aqui adicionar antes do parágrafo. Assim nosso código fica mais organizado.

[05:20] Vamos então também, ainda falando sobre cores, mudar a cor de um dos textos.

[05:30] Digamos que eu queira colocar a nossa missão, que hoje é preta, na cor vermelha. Onde está a marcação específica dessa frase? Qual tag está marcando isso? A tag do parágrafo marca tudo, a tag da ênfase marca toda a linha, e a tag strong marca especificamente só aquele conteúdo.

[06:00] Para a tag strong, vamos dar uma definição de cor. Dentro do nosso arquivo de CSS, vamos abrir uma nova definição para a tag strong e vamos colocar a definição de cor.

[06:15] Até agora, só falamos sobre a cor de fundo, que é o background. Para a cor do elemento vamos a propriedade color. Vou usar o nome vermelho, red.

[06:28] Reparem que usei aqui um nome, lá em cima usei um hexadecimal. Não se preocupem, isso tudo funciona, o CSS nos dá essa liberdade para fazer essas modificações. Aos poucos vamos entendendo mais sobre isso.

[06:43] Salvando esse arquivo e voltando para o navegador, a nossa tag está em vermelho.

[06:52] Reparem no que aconteceu. Como estou usando a tag strong para marcar nossa missão, mas também estou usando a tag strong para marcar Barbearia Alura, as duas palavras ficaram em vermelho.

[07:07] Se voltarmos ao nosso HTML, conseguimos ver muito claramente essa similaridade. Ou seja, temos a tag strong usada em dois momentos diferentes.

[07:23] Como faço então para deixar um CSS específico para somente um elemento? Precisamos que: ou o CSS seja aplicado in line ou que mudemos o CSS para atender a uma estrutura.

[07:45] Como assim? Temos um strong dentro de um parágrafo. Na linha de baixo, temos um strong dentro de em, que está dentro de um parágrafo. No nosso CSS, temos como fazer isso. Temos como selecionar especificamente o strong que está dentro do em.

[08:07] Fazemos isso colocando a seguinte estrutura: em strong.

[08:16] Lendo isso como o navegador vai ler, ele vai procurar todos os strongs que estão dentro de todos os ems. No nosso caso, no nosso site, só temos uma possibilidade para essa marcação.

[08:34] Se salvarmos e voltarmos à página, vamos ter marcado especificamente somente este elemento.

[08:42] Aos poucos vamos entender como marcar mais especificamente os elementos e como fazer o código ficar menos complexo. Mas o importante aqui é entendermos como funciona o CSS e essa propriedade do estilo em cascata. Ou seja, marcamos o body e isso refletiu para todos os filhos dele. Marcamos o parágrafo e ele replicou para todos os parágrafos. E na última linha, marcamos o strong, que está dentro do em, ou seja, fazendo essa cascata. Conseguimos procurar este elemento usando a estrutura do HTML.

Interface gráfica do usuário, Texto, Aplicativo, Teams

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

[00:00] Neste bônus, vamos falar sobre cores e como elas funcionam no CSS, no navegador.

[00:06] A primeira coisa sobre a qual precisamos falar são alfabetos. Temos o nosso alfabeto padrão, abcde e por aí vai. Esse é o alfabeto que usamos no português.

[00:26] Os alfabetos para computação são um pouco diferentes. Por exemplo, temos o binário. É o 0 e o 1. Temos o numérico, que vai do 0 ao 9. E aí conseguimos representar todos os números com isso. Ainda temos o dicionário hexadecimal, que é a mesma coisa que o dicionário numérico adicionando as letras abcdef.

[01:03] Com esse dicionário conseguimos representar muito mais coisas do que só com o dicionário numérico. É ele que usamos para marcar cores.

[01:14] Vamos voltar e entender como funcionam as cores para nós.

[01:20] Nós conseguimos enxergar três espectros de cor, e o HTML monta isso. Nós enxergamos o espectro RGB, que quer dizer Red Green Blue. Ou seja, vermelho, verde e azul.

[01:38] Montamos a cor seguindo essa mesma característica, usando o dicionário hexadecimal. Para fazer isso, colocamos # e marcamos seis elementos. Os dois primeiros para marcar o vermelho, os dois segundos para marcar o verde e os dois últimos para marcar o azul.

[02:02] Como é a representação numérica, ou em hexadecimal, para esses números? O zero é a ausência de cor, e o F é o máximo de cor. Então, se eu quiser representar, por exemplo, o preto, que é a ausência de todas as cores, coloco #000000. Ou seja, 00 para vermelho, 00 para verde e 00 para o azul. Se eu quiser representar o branco, coloco #FFFFFF. FF para o vermelho, FF para o verde e FF para o azul.

[02:42] Lembrem que no nosso exercício, se voltarmos ao style do CSS, usamos a representação #CCCCCC. Se olharmos no espectro do hexadecimal, o CC é quase o branco. Não é o preto completo e nem o branco completo.

[03:10] Essa é a representação dos elementos.

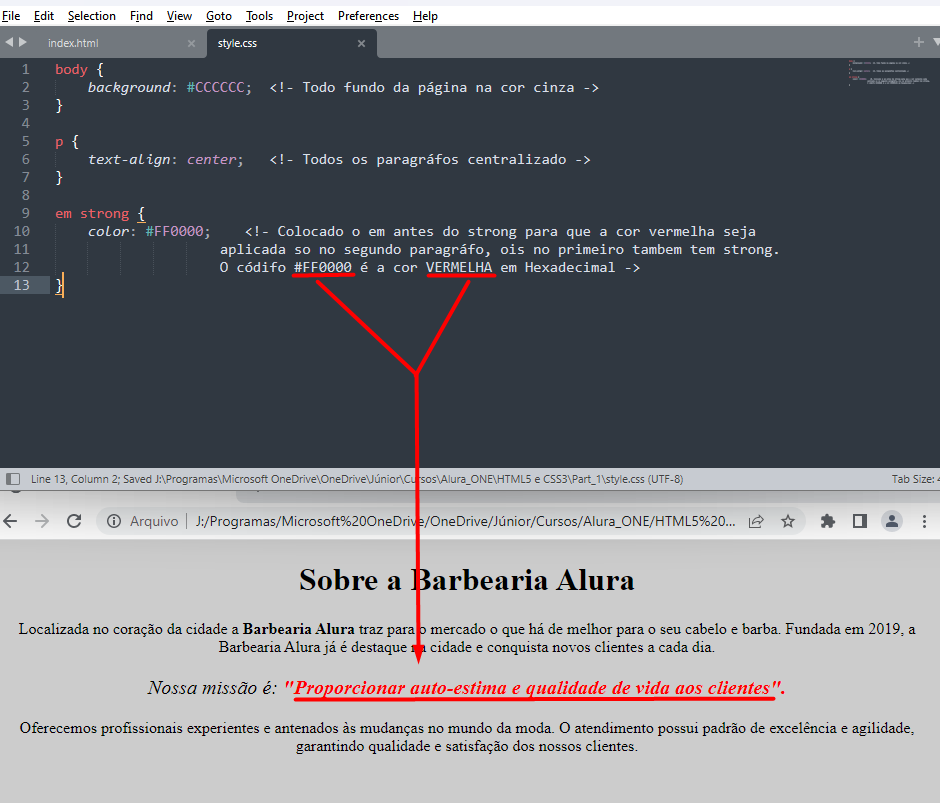
[03:16] Como faríamos, por exemplo, para representar a cor vermelha em hexadecimal? Precisamos colocar # e precisamos que o vermelho seja completo, ou seja, no máximo, e que todas as outras cores não existam: #FF0000.

[03:36] Podemos voltar ao nosso CSS e alterar para, ao invés de vermelho, #FF0000. Assim, marcamos o vermelho em hexadecimal, e é assim que vamos usar as cores em hexadecimal no nosso projeto.

[03:55] É importante saber também que existe outra forma de representar as cores, além das que já vimos. Temos as cores básicas, a linguagem hexadecimal e a RGB. O mesmo padrão RGB que falamos sobre as cores que conseguimos enxergar, temos um dicionário RGB. Um alfabeto. Ele vai do 0 até 255. Ou seja, ele tem 256 níveis de representação de cada cor.

[04:44] Para isso, o 0 também é a ausência e o 255 é o máximo. A representação no CSS é um pouco diferente. Ao invés de colocar #, eu coloco RGB( , , ), separando as cores por vírgula. Antes da primeira vírgula vem o vermelho. Então, se eu quiser representar o branco, coloco RGB(255, 255, 255). Se eu quiser representar só o azul, RGB(0, 0, 255).

[05:32] Essa é a forma que usamos para representar as cores com esses alfabetos no CSS e isso ser renderizado pelo nosso navegador.



Texto

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, Teams

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Na mesma pasta da página **index.html**, crie o arquivo CSS **style.css**. Neste arquivo, modifique a cor de fundo do body, centralize os parágrafos e altere a cor das *tags* strong que estão dentro de *tags* em:

body {

background: #CCCCCC

}

p {

text-align: center;

}

em strong {

color: #FF0000;

}COPIAR CÓDIGO

2) Na página **index.html**, como último elemento do head, importe o CSS **style.css**:

<link rel="stylesheet" href="style.css">COPIAR CÓDIGO

3) Centralize o texto do h1:

<h1 style="text-align: center">Sobre a Barbearia Alura</h1>COPIAR CÓDIGO

4) E altere o tamanho da fonte do segundo parágrafo:

<p style="font-size: 20px"><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>COPIAR CÓDIGO

5) A página completa ficará da seguinte forma:

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<title>Barbearia Alura</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<h1 style="text-align: center">Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a <strong>Barbearia Alura</strong> traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p style="font-size: 20px"><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

</body>

</html>

Uma imagem contendo Texto

Descrição gerada automaticamente

1. Estilizando imagens

## Transcrição

Você pode baixar um ZIP com a imagem ***banner.jpg*** [*aqui*](https://caelum-online-public.s3.amazonaws.com/1179-html5-css3/04/banner.zip).

[00:00] Vamos começar nossa aula organizando um pouco nosso código, porque ele está bem bagunçado. Tem algumas coisas que não fazem mais sentido. Por exemplo, a tag style, que está vazia. Podemos removê-la.

[00:14] Outra coisa que não está legal é termos CSS no nosso código enquanto temos um arquivo de CSS. O ideal seria movermos todo esse conteúdo para lá.

[00:27] O h1 é muito fácil. É só recortar o style, voltar para o nosso arquivo de CSS, e, como eu disse anteriormente, sempre tentando deixar a estrutura do CSS parecida com a estrutura do HTML. Temos o body, o h1, e aí o parágrafo. Vamos colocar o h1 e depois o style que tínhamos recortado.

[01:02] Vou apenas colar o que estava lá. Reparem no editor de código funcionando. Ele marcou que tem algo muito errado nessa linha, porque colocamos as aspas e o style, já que eu só recortei e colei. Nós não precisamos deles.

[01:14] Se eu salvar, voltar para o navegador, recarregar, o código está perfeito. Não tem nada errado. Funcionando do mesmo jeito. Só movemos o CSS.

[01:28] Mas temos um problema com esse CSS, porque ele é específico para um parágrafo. Para resolver isso, precisamos ter uma marcação para o HTML específica para esse elemento, e no CSS referenciar isso.

[01:49] Usamos para isso o marcador de identificação. É o id. Vamos colocar id = “”. Entre as aspas você coloca um nome qualquer, que é o mesmo que vai ser usado no nosso CSS.

[02:07] Vamos chamar nosso id de “missao". Vamos recortar o código CSS e colar no arquivo CSS. O “missao" é um dos parágrafos, ele vem logo depois do anterior. E como fazemos para marcar? Toda vez que estamos marcando uma tag e queremos referenciar ela no CSS, usamos o nome da tag. Toda vez que temos um identificador no HTML e queremos referenciar ele no CSS, usamos a #. Logo: #missao.

[02:50] Reparem que o editor de códigos já marcou com outra cor, para marcar que está certo.

[02:57] Vamos colar aquele código que tínhamos e salvar. Se recarregarmos o navegador, o código está perfeito. Não tem nada errado.

[03:11] Vamos evoluir esse código então. Vamos adicionar mais conteúdo à página, uma imagem.

[03:21] Essa imagem já está na pasta do projeto de vocês, podem baixar. É uma imagem com cadeiras de barbearia e um balcão atrás. Ela está com o nome “banner.jpg”.

[03:41] Assim como o CSS, a imagem também é um arquivo externo. Nós colocamos a chamada para o CSS no head, porque é uma informação que estamos passando para o navegador. A imagem é uma informação que queremos exibir na nossa página, então vamos colocá-la dentro do body.

[04:01] Na estrutura visual da nossa página, a imagem vai ser a primeira coisa que vai aparecer. Então, em cima de tudo, vou chamá-la. Usamos a tag img para adição da imagem ao conteúdo.

[04:19] A tag img serve para chamarmos uma imagem, e como ela é um arquivo externo, precisamos dizer onde está, dizer qual a fonte dessa imagem. Para isso, usamos a propriedade source. Só que não escrevemos isso por inteiro. Abreviamos para src e colocamos o endereço: img src=”banner.jpg”.

[04:53] Como essa é uma tag de chamada de arquivo externo, não abrimos e fechamos. Ela é só a definição da tag.

[05:02] Se salvarmos o arquivo e abrirmos no navegador, recarregando a página, a nossa imagem já está ali.

[05:12] A imagem é essa tag, com uma fonte e um endereço.

[05:20] Na próxima aula vamos entender como fazer com que a imagem ocupe toda a largura da página, como fazer ela se posicionar melhor no texto, como brincar com espaçamento, etc. Mas isso fica para a próxima. Aqui, nós vimos como reestruturar nosso código, criar um identificador para marcar especificamente um elemento e adicionar uma imagem à nossa página.



Texto

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

## Transcrição

[00:00] Na aula passada, adicionamos uma imagem ao conteúdo da nossa página. Nesta aula, vamos aproveitar para mexer no CSS da nossa imagem.

[00:09] Se repararmos, podemos ver um erro. Nossa imagem não ocupa 100% da largura da página.

[00:18] Vamos aproveitar essa aula também para falar de todas as dimensões de um elemento no HTML.

[00:27] Se olharmos nosso HTML, vamos reparar que a imagem está adicionada antes do h1. Para criarmos um CSS que funcione para ela, vamos adicionar um identificador. Ele vai ter o mesmo nome da imagem, “banner”.

[00:49] Já vimos no nosso CSS que a posição em que ele deve estar é logo acima do h1. Para criarmos um estilo para o identificador, colocamos uma # e no nome dele: #banner.

[01:05] Nele, vamos adicionar todo o estilo que queremos.

[01:09] Voltando ao navegador, é importante falarmos sobre uma ferramenta que todos os navegadores modernos entregam para os desenvolvedores. É chamada de developer tools, ferramentas para o desenvolvedor.

[01:25] Se vocês apertarem F12, irão reparar que a ferramenta do desenvolvedor está à direita. Ela tem, entre outras coisas, um layout mobile para sua página, te deixa mexer no console, saber que tipo de arquivos estão sendo transacionados. Mas isso é assunto para o futuro. O que é importante agora é vermos que no developer tools temos uma divisão muito clara entre o que é o HTML, o que é o CSS e o que está sendo exibido no nosso elemento.

[01:57] Como isso funciona? Digamos, por exemplo, que eu quero ver meu h1. Se eu clicar sobre ele, vou ver todo o estilo dele, todo o estilo que já foi criado, e esse estilo vem do nosso arquivo CSS, e verei também a marcação que ele tem na minha página. Ou seja, a largura, altura, qual o espaçamento que ele tem.

[02:25] Essa ferramenta é fundamental para entendermos e trabalharmos com desenvolvimento. Com certeza você vai usar muito.

[02:32] O que queremos agora é mudar a largura do nosso elemento. Falando sobre dimensões no CSS, temos quatro parâmetros muito importantes. A largura de um elemento, a altura do elemento, e os espaçamentos. Espaçamento interno e espaçamento externo.

[02:54] Vamos item por item.

[02:56] Vou usar nosso h1 como exemplo e vou mexer no developer tools (F12). Reparem que se eu clicar logo depois de text align center ele vai criar uma nova linha para eu poder inserir uma modificação.

[03:10] Só por curiosidade, vamos colocar a cor azul, para vocês verem funcionando.

[03:17] Imediatamente ele vai alterar a exibição do elemento, deixando o título do h1 em azul. Eu não quero em azul, então vou simplesmente apagar.

[03:29] Posso também, voltando com o título azul, recarregar a página, e quando recarrego ele some com aquele estilo criado. É só uma ferramenta para o desenvolvedor, na máquina dele, enquanto ele está fazendo aquilo funcionar.

[03:50] No meu h1, vamos falar algumas coisas sobre a dimensão dos elementos.

[03:55] A dimensão de altura é muito fácil de identificar. Eu insiro com height e coloco em pixels, por exemplo, 100 pixels, ou 100px. Esse elemento passou a ter um espaço muito maior.

[04:17] Colocando o mouse em cima, consigo ver no espaço sombreado de azul a altura do elemento. Vamos alterar para 150px. Vemos que a área aumentou na altura. Só que esse elemento já está ocupando a largura inteira da página. Vamos agora na nossa imagem, onde já temos o identificador do banner, mexer na largura.

[04:43] A largura de um elemento geralmente é adicionada em pixels, ou podemos também colocar a largura em percentual. Se eu quero que ocupe a página inteira, vou colocar 100%. Imediatamente altero a largura da imagem. Ela passa a ocupar 100% da largura da página.

[05:06] Vamos ver o que acontece se eu mexer na altura da imagem e colocar um height com 120px. Reparem que ele respeitou a medida de largura, ou seja, 100%, e ajustou a imagem para que ela ficasse apertada com 120px de largura.

[05:27] A imagem mexe proporcionalmente. Então, se eu aumento a largura, a altura aumenta proporcionalmente. Se eu aumento a largura e a altura, preciso fazer um cálculo muito correto para essa imagem não ficar distorcida. Nós não queremos alterar a altura do elemento.

[05:48] Vamos voltar ao h1. Eu já falei sobre a largura e sobre a altura. Vamos falar sobre o espaçamento.

[05:55] Para fazer isso, vou criar uma borda no elemento, que tem 10px, com formato sólido e cor preta. Não precisa se preocupar agora com essas configurações. Vamos entender sobre bordas e como elas funcionam no futuro.

[06:22] Mexendo na borda, conseguimos entender a largura e a altura do elemento. 100% de largura e 150px de altura. Vamos alterar a altura em pixels para entendermos o que acontece.

[06:39] A borda vai se ajeitando. Ela é a finalização do elemento.

[06:49] O que acontece então se colocamos um espaçamento interno? Ele é chamado de padding, e é configurado de várias formas. Ou ele funciona em todos os lados, ou só para cima, ou só para as laterais, ou só para baixo.

[07:07] Vamos imaginar um espaçamento interno para todos os lados, para ficar fácil de entender. Se eu colocar um espaçamento de 20px internamente, ele aumenta o meu elemento, criando um respiro entre a borda e o conteúdo. Ou seja, ele criou uma margem interna no elemento.

[07:33] Vou clicar do lado para desmarcar o padding, ou seja, ele vai cancelar essa criação, para ficar clara a diferença.

[07:42] Se eu colocar um espaçamento interno, ele vai criar um quadro dentro, dando um respiro para todos os lados. Se eu coloco, por exemplo, um espaçamento só para cima de 20px, ele cria esse respiro só na parte de cima. Esse é o espaçamento interno. É para fazer seu elemento se comportar melhor no espaço que ele tem.

[08:10] O espaçamento interno serve, por exemplo, para eu criar um respiro maior da borda para fora. Então, se eu quiser que a borda não esteja colada na lateral, preciso de um espaçamento externo lateral. Para fazer isso, adiciono a margin, que é o espaçamento externo, ou para a direita ou para a esquerda.

[08:35] Vamos fazer só na esquerda uma margem de 40px. Nossa borda descolou da lateral, criamos um espaço para fora do elemento, entre o elemento e a margem do navegador.

[08:53] Nosso elemento já tem um espaço para cima e para baixo, que foram adicionados pelo navegador inicialmente. Se eu quiser colocar uma margem para todos os lados igual, coloco só o nome dela, só a declaração margin. E na hora que renderizo, ele vai criar uma moldura em volta do meu elemento.

[09:14] Com isso, conseguimos mexer na configuração e na apresentação de todos os itens. Conseguimos mexer na largura, na altura, na borda, no espaçamento interno e no espaçamento externo.

[09:28] Vou recarregar a página, voltando para o que já tínhamos. A única coisa que quero fazer é mudar a largura da imagem, então vamos ao nosso CSS. Agora que já aprendemos, vou mudar a largura com a propriedade width e vou colocar o valor 100%.

[09:46] Salvando e recarregando no navegador temos o site do jeito como queremos.

[09:55] Reparem que quando fechei o developer tools, a imagem se ajustou para sempre o tamanho da página. Isso vai acontecer com qualquer tamanho de tela. Ela vai sempre ocupar 100%.

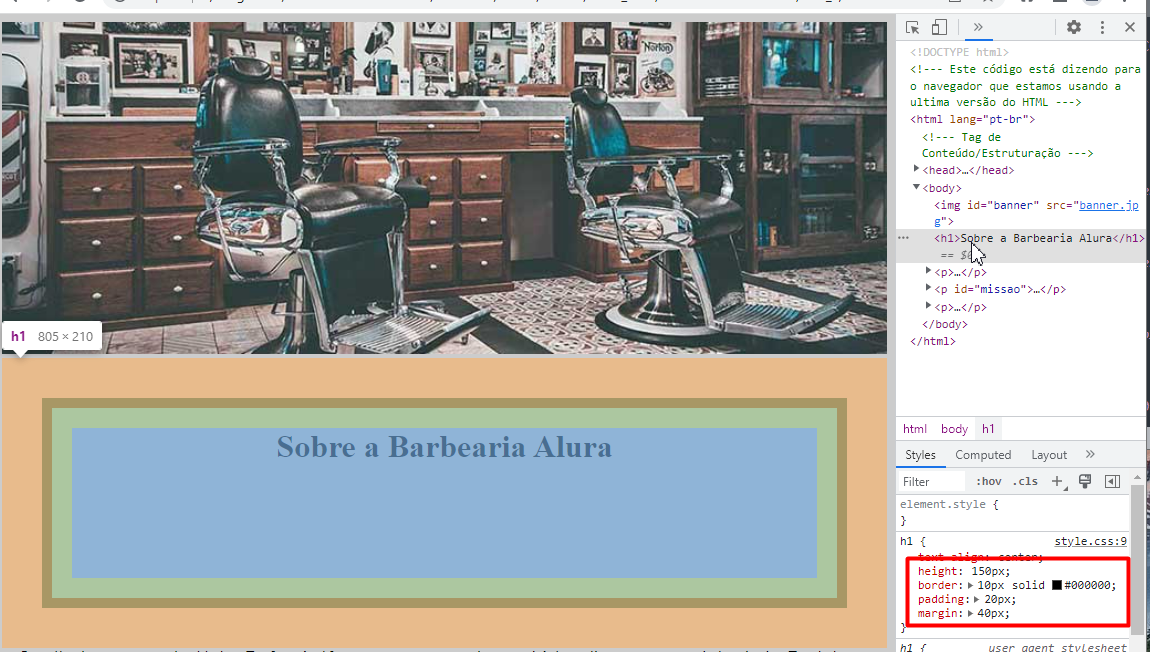
[10:09] Nós ainda vamos praticar muito espaçamento, largura e altura dos elementos. Se isso não ficou extremamente claro, não se preocupe. Aos poucos, com a prática, vai ficando natural.

[10:22] Na próxima aula, vamos melhorar ainda mais esse conteúdo. Te vejo lá.

Interface gráfica do usuário

Descrição gerada automaticamente

* Na imagem abaixo visualizamos em tempo real as mudanças feitas a través da Ferramenta de Desenvolvedor (Developer Tools) tecla F12. Mas essas mudanças não ficam registradas. Ao atualizar a página, o conteúdo volta ao estado original. É mais para visualizar como fica o layout da página e depois aplicar no arquivo .CSS.



Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

## **Transcrição**

[00:00] Vamos fazer aqui uma aula bônus com um vídeo rápido, só para falarmos como se comporta um time de **Front-End** hoje em dia.

[00:07] As interfaces do front-end estão cada vez com mais responsabilidade, são cada vez mais robustas, tem cada vez mais funções. E por isso o time foi crescendo muito.

[00:19] É importante destacar que um time de front-end, ou seja, um time para fazer uma interface não é composto só pelo desenvolvedor front-end. Geralmente, temos alguém responsável pela usabilidade do site.

[00:34] A usabilidade (**UX – User Experience**) é responsável por responder como vai entregar aquelas informações, a forma.

[00:42] Temos ainda o time da interface do usuário. O time do design (**UI – User Interface**), que vai pegar a forma como aquilo vai ser feito e vai dar o visual para aquilo.

[00:55] Por último, temos o codificador, aquele **Desenvolvedor Front-End** que vai pegar o design, a forma como o site foi apresentado com aquele visual e vai transformar em código para web, para um aplicativo ou para algum sistema.

[01:14] Essas responsabilidades (**UX + UI + Desenvolvedor**) muitas vezes estão acumuladas em algumas pessoas, mas é importante pontuar cada uma delas. No nosso site, no nosso projeto, reparem que não paramos para discutir sobre qual cor iríamos escolher, ou qual imagem escolher. No nosso projeto, o designer já fez isso para nós.

[01:36] Geralmente é assim que um desenvolvedor front end vai trabalhar. ***Ele vai receber os inputs e vai extrair daquilo o melhor código possível***.

[01:47] Espero que vocês consigam trabalhar assim e que consigam entregar o melhor código possível.

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

**Consolidando o seu Conhecimento**

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Remova todo o CSS inline da página **index.html**.

2) Como primeiro item do body, adicione a imagem **banner.jpg** (se você ainda não a baixou, faça o seu download [aqui](https://caelum-online-public.s3.amazonaws.com/1179-html5-css3/04/banner.zip)). Adicione também um identificador para a tag img:

<img id="banner" src="banner.jpg">

3) Adicione um identificador para o segundo parágrafo do texto:

<p id="missao"><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

4) No arquivo **style.css**, adicione o CSS que foi removido da página **index.html**, ou seja, alinhe o h1 e os parágrafos no centro da página e aumente a fonte do segundo parágrafo (use o seu identificador para referenciá-lo):

h1 {

text-align: center

}

p {

text-align: center;

}

#missao {

font-size: 20px

}

5) Ainda no arquivo **style.css**, altere a largura da imagem:

#banner {

width: 100%;

}

6) O CSS completo ficará da seguinte forma:

body {

background: #CCCCCC

}

#banner {

width: 100%;

}

h1 {

text-align: center

}

p {

text-align: center;

}

#missao {

font-size: 20px

}

em strong {

color: #FF0000;

}

**O que aprendemos?**

Nesta aula, aprendemos:

* Como reestruturar o nosso código, removendo os **CSS *inline*** e colocando-os no **arquivo CSS externo**
* Como criar um identificador para marcar especificamente um elemento
  + Como fazer referência a esse identificador no CSS
* Como adicionar uma imagem à nossa página
* Como ajustar a altura do elemento, através da propriedade **height**
* Como ajustar a largura do elemento, através da propriedade **width**
* Como ajustar o espaçamento interno do elemento, através da propriedade **padding**
* Como ajustar o espaçamento externo do elemento, através da propriedade **margin**
* Como se comporta um time de front-end hoje em dia

1. **Listas e Divisões de Conteúdo**

## **Transcrição**

[00:00] Já evoluímos nosso site até aqui. Agora, vamos adicionar mais conteúdo. Vamos adicionar uma nova seção no site, que serão os benefícios da nossa barbearia.

[00:10] Para isso, vamos voltar ao nosso HTML. Abaixo do último parágrafo, vamos começar uma nova seção, chamada “Benefícios”.

[00:23] Vimos na primeira aula que para criarmos um título com HTML, usamos a tag h. e usamos h1 para ser o título principal da página. Agora que estamos criando uma nova seção, outra parte do nosso site, vamos colocar outro título. Seguindo a sequência, vamos usar h2.

[00:46] O h2 começa nosso novo conteúdo. Se salvarmos isso e formos no navegador, veremos o “Benefícios” destacado em negrito, seguindo o mesmo padrão que vimos para o título principal da página.

[01:07] Agora, queremos listar os benefícios da nossa barbearia, que são quatro.

[01:13] Vamos ao HTML. A estrutura que eu desejo é: o primeiro item, atendimento aos clientes é o nosso maior benefício. O segundo item, espaço diferenciado. Terceiro item, localização. Quarto e último item, profissionais qualificados.

[01:51] Essa é a intenção que eu tenho. Quero que esses itens estejam listados.

[01:58] Se formos até nossa página e recarregarmos, veremos que estão todos um do lado do outro, sem uma forma. Falta uma tag para marcarmos esses itens.

[02:10] No nosso HTML, a solução mais fácil dadas as ferramentas que temos seria colocar uma tag de parágrafo, mas ela não é correta para essa demanda. Não é a semântica correta para uma lista. As listas têm uma tag própria.

[02:31] Temos as listas não ordenadas, onde não importa qual item vem primeiro ou depois. E temos as listas ordenadas, como por exemplo uma receita de bolo, onde primeiro tenho que ligar o forno e depois colocar a massa.

[02:49] Aqui queremos uma lista não ordenada. Para isso, usamos a tag ul, de unordered list. Uma lista sem ordem.

[03:01] Se quisermos uma lista ordenada, é só trocar para ol, de ordered list.

[03:08] A tag ul é uma tag de conteúdo. Ela inicia e fecha. Precisa fechar ao final da lista. Vamos consertar a endentação dos nossos itens para visualmente sabermos tudo que está acontecendo. Todos os itens da nossa lista vão dentro da ul.

[03:29] Se salvarmos isso e recarregarmos o navegador, veremos que nada aconteceu, só temos agora um pequeno espaço no início da primeira linha. Por quê? A lista, além da tag ul ou ol, no caso usamos ul para lista não ordenada, ainda precisa de marcação para cada item.

[03:50] Para isso, usamos a tag li, de list item. Para cada um dos itens, vou colocar a tag li. Ela inicia em cada uma das linhas e termina ao fim daquele conteúdo, daquele item.

[04:18] Agora que já temos os itens marcados podemos voltar ao navegador. Temos uma lista com os itens não ordenados. Podemos até remover o \*, que serviu só para explicarmos visualmente qual era o objetivo principal.

[04:40] Agora que temos a lista, que temos toda essa estrutura para nosso conteúdo, queremos criar um CSS para ele. Mas até agora só vimos duas formas de marcar o CSS. Primeiro usando as tags e o segundo usando os identificadores.

[05:02] O problema de usarmos as tags é que todas as listas do nosso site terão o mesmo estilo. E o problema de usar identificadores é que eles são únicos. Ou seja, temos uma lista com quatro itens e para marcar cada um deles, teríamos que criar quatro identificadores.

[05:19] Existe uma solução para isso. É o conceito das **classes no CSS**. As classes no CSS servem para marcarmos itens, para posteriormente colocarmos estilo em cada um deles, só que são repetíveis. Ou seja, podemos marcar todos os nossos itens com a mesma classe.

[05:38] Vamos fazer isso.

[05:40] Assim como no identificador, em que adiciono a propriedade id e dou um nome para ela, nas classes faço a mesma coisa. Adiciono a propriedade “class” e entre aspas o nome dela.

[06:14] Podemos colocar o mesmo nome em todos os itens. Essa vai ser a classe que usaremos para marcar todos esses nossos itens.

[06:20] Agora que temos os itens marcados, queremos criar um tamanho de fonte específico para eles e colocar todos em itálico.

[06:30] Vimos nas primeiras aulas como colocar um elemento em itálico visualmente simplesmente usando uma tag, que é ênfase, a tag em.

[06:43] Como fazemos para colocar vários elementos em itálico sem ser uma ênfase, sendo só um detalhe visual? No nosso CSS, vamos criar a referência para classe. Para fazer isso, quando usamos o **identificador**, usamos a **#**. Quando usamos uma **classe**, colocamos um “**.**”.

[07:02] Então .itens, e aí podemos criar todo o CSS para aqueles quatro itens da nossa lista. Quero dizer que o texto está em itálico, e para isso vamos colocar a propriedade font style, com a variável itálico: font-style: italic.

[07:22] Ou seja, estamos transformando todos aqueles itens em itálico.

[07:28] Se recarregarmos nossa página, veremos todos os itens marcados com itálico.

[07:35] Só que no nosso layout visual quero separar esse conteúdo. Quero que comecemos a ter um fundo branco nos itens, ao invés do fundo cinza, justamente para gerar esse destaque.

[07:48] Por isso, na próxima aula nós vamos falar em como criar divisões no nosso conteúdo.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

## **Transcrição**

Você pode baixar um ZIP com a imagem ***beneficios.jpg*** [*aqui*](https://caelum-online-public.s3.amazonaws.com/1179-html5-css3/05/beneficios.zip).

[00:00] Agora, queremos criar mais uma divisão no nosso site, uma separação entre as duas seções.

[00:06] A primeira, onde teremos todo o texto de apresentação do nosso site, e a segunda, onde teremos os benefícios apresentados ao nosso cliente.

[00:14] Para isso, realmente precisamos ter o conceito de uma divisão. Criar dois blocos de conteúdo e separá-los com tags de HTML.

[00:25] Se formos ao nosso HTML, veremos que toda essa primeira parte, a partir do h1, e os quatro parágrafos, respondem a uma primeira seção, que poderíamos chamar de principal; e o h2 e nossa lista pertencem a uma segunda seção, que poderíamos chamar de Benefícios.

[00:52] Para fazer isso, criamos a tag de divisão, a tag div. A primeira vai englobar o h1 e os três parágrafos, e aí vamos indentar isso para ficar correto. Uma dica: se você selecionar tudo com o mouse ou com o teclado e apertar tab uma vez, ele vai indentar tudo como um bloco para você.

[01:25] Vamos fazer a mesma coisa nos benefícios. Criar outra divisão que fecha depois da nossa lista e indentar isso para ficar correto.

[01:42] Temos então no nosso HTML duas divisões. Vamos entender como isso ficou visualmente no nosso navegador. Observando, não notamos diferença alguma.

[01:52] As divisões têm por padrão não interferir na apresentação visual do nosso conteúdo. Elas servem para marcarmos nosso conteúdo, e, a partir do CSS fazermos os efeitos que quisermos. Mas como vamos marcar o conteúdo?

[02:12] Temos duas tags com o mesmo nome. Para isso, precisamos de especificidade, precisamos ou criar um identificador, ou criar uma classe.

[02:25] Já falamos a diferença dos dois. O identificador serve para itens únicos e a classe serve para itens que serão repetidos.

[02:33] Como só temos esse conteúdo, sabemos que eles são únicos, mas se quisermos criar um layout robusto, com todas as técnicas, e quisermos replicar esse conteúdo no futuro, o ideal é criarmos classes. Vamos seguir esse padrão a partir de agora. Sempre que quisermos marcar um conteúdo visualmente, vamos criar uma classe para ele.

[02:54] A classe que queremos criar é a dos benefícios, então vamos criar uma classe com o nome “benefícios”.

[03:02] Se formos no CSS, a marcação de uma classe é um ponto e o nome dela. O que queremos aqui é que ele tenha um fundo branco, e um fundo branco, como vimos, é com a propriedade background.

[03:27] Para fazer a cor branca, precisamos que todos os itens da nossa cor estejam no máximo, ou seja, todas as cores. Então, #FFFFFF.

[03:41] Se recarregarmos, veremos que essa nossa divisão está perfeita no nosso conteúdo. Mas temos um problema. Ela está com fundo branco, mas o fundo da minha página inteira está cinza. Por que isso aconteceu?

[04:00] Quando colocamos o fundo cinza, colocamos na tag do background, e não no nosso conteúdo de texto de apresentação. Portanto, vamos mudar isso a partir de agora.

[04:13] Na nossa tag do conteúdo principal, vamos criar uma classe com o nome “principal”. Para esta tag, quando voltarmos no nosso HTML, lembrando que a estrutura do CSS deve ser correspondente à estrutura do HTML, logo abaixo do banner vamos criar a classe principal. E nela vamos recortar o fundo cinza do background e vamos colar no principal.

[04:54] Agora, temos o site inteiro com o fundo branco e nosso espaço também com fundo branco.

[05:05] Um problema que temos que resolver no nosso site é que temos um espaço branco indesejado entre a imagem e o nosso conteúdo principal, e um espaço branco indesejado na lateral do nosso site.

[05:23] Isso foi o navegador que adicionou. Já vamos aprender a resolver.

[05:29] No momento, temos nosso título de benefícios e nossa lista de itens. Vamos continuar estilizando-os.

[05:38] Seguindo o padrão do bloco de cima, quero que o Benefícios seja centralizado. Para fazer isso, lembrando que no nosso HTML ele está com a tag h2, voltando no nosso CSS, vou criar um marcador para ele a partir da tag h2: text-align: center

[06:06] Voltando na nossa página, já temos um título centralizado.

[06:12] Por último, para completar esse bloco de conteúdo, precisamos adicionar mais uma imagem. Também já sabemos como fazer isso. Através da tag img.

[06:25] Vamos criar a img e vamos colocar o source dela, a fonte, onde essa imagem está. Ela é nossa imagem beneficios.jpg, que está na pasta do projeto.

[06:42] Quando voltarmos ao nosso navegador, a imagem está lá.

[06:48] Reparem que ela está muito grande, ocupando a largura inteira da página. O ideal é que ela fique ocupando 50% da página do lado direito, ao lado dos itens da lista.

[07:06] Na próxima aula, vamos entender como criar essa divisão e posicionar esses elementos do jeito que queremos.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

## **Transcrição**

[00:00] Vamos criar o estilo necessário para fazer este bloco de conteúdo ficar perfeito.

[00:06] A primeira coisa que temos que fazer é ajustar a imagem. Ela está muito grande, ocupando 100% da largura do elemento, e não é isso que queremos. Queremos que ela fique posicionada ao lado da lista de benefícios.

[00:21] Para fazer isso, vamos mexer na nossa imagem, e já sabemos como. Precisamos criar uma classe para esse elemento.

[00:31] Eu vou criar a classe e chamá-la de “imagembeneficios”.

[00:45] No meu CSS, vou criar uma marcação para essa classe: .imagembeneficios

[00:56] E vou dizer que ela vai ter, a partir de agora, 50% da largura: width: 50%

[01:05] Quando voltamos ao navegador, essa imagem já passou a ocupar 50% da largura da página. Só que o que aconteceu para ela não estar ao lado dos itens?

[01:16] Se lembrarmos quando fizemos nossa lista sem a marcação dos itens, da tag li, estavam todos em uma linha. A mesma coisa aconteceu quando fizemos os parágrafos iniciais. Quando colocamos a tag li, ela passou a ocupar 100% da largura da página e quebrar a linha depois dela. Ou seja, o próximo item começava na linha de baixo.

[01:41] Esse comportamento de uma tag ocupar a largura inteira da página é chamado “block”. Ele bloqueia o conteúdo daquela linha. Todos os itens da nossa linha têm o comportamento block. Uma imagem não bloqueia o conteúdo, ela deixa que existam outros na lateral, e esse tipo de conteúdo é considerado inline.

[02:10] A diferença entre eles é que mesmo que eu diminua o tamanho, diminua a largura de um elemento block, ele vai sempre ocupar aquela linha, mesmo preenchendo só metade.

[02:25] Um elemento inline não me deixa alterar, por exemplo, o espaçamento externo e interno dele.

[02:34] Para isso, existe uma terceira característica, quando o elemento possui as duas condições. Ele é inline e block ao mesmo tempo. Ou seja, ele bloqueia uma largura, mas essa largura é fixa. Sou eu que dou o tamanho. E ele me deixa também mexer na largura e nos espaçamentos interno e externo.

[02:58] Vamos aplicar isso para ficar mais claro.

[03:01] A primeira coisa que temos que fazer é que a nossa lista precisa de uma marcação. Ou seja, nosso ul precisa de uma marcação. Eu falei do inline e do block. Isso são características do display. Ele pode ser block, inline ou juntar as duas coisas e ser do tipo inline block.

[03:33] O display inline block ocupa só o tamanho do conteúdo, mas me deixa mexer na largura e espaçamentos. Vamos ver na prática.

[03:43] No nosso navegador, quando recarregamos, vemos que o elemento está ocupando só o tamanho do seu conteúdo, e do lado dele já entrou nossa imagem. Mas aconteceu um erro. Na verdade, é uma características do CSS.

[04:01] Todos os elementos são alinhados pela linha de baixo e queremos que sejam alinhados pela linha de cima. Para isso, no nosso CSS, adicionaremos a propriedade “vertical-align: top”.

[04:18] Quando recarregamos a página, nosso título está centralizado, nossa lista de itens está ocupando só o tamanho dela e nossa imagem está ocupando 50% da página.

[04:33] Vamos agora mexer um pouco na nossa lista de itens. Dar um espaço e aumentar a largura. Como ele é um elemento do tipo inline block, eu posso dizer que ela tem de largura 20% da página e que ela ainda tem um espaçamento externo para a direita de mais 15%.

[05:00] Se recarregarmos a página, veremos os itens se dividirem melhor. Veremos a imagem ocupando metade do conteúdo, alinhada à nossa lista de itens e alinhada também a esse espaçamento que desejamos no meio.

[05:22] Para finalizar esse layout desses dois blocos, agora que estamos revisando largura e espaçamento, vamos mexer também no espaçamento inteiro dos dois blocos.

[05:35] No primeiro, no Benefícios, vamos adicionar um espaçamento interno para todos os lados de 20px, ou seja, padding: 20px

[05:48] Quando voltarmos ao navegador, veremos as coisas se organizando um pouco melhor. Temos uma margem de respiro em baixo e na lateral.

[05:59] No nosso bloco principal, quero adicionar um espaçamento interno de 30px, padding: 30px

[06:13] Agora nosso conteúdo está melhor distribuído e nosso site começando a ter uma cara mais definitiva.

[06:23] Para finalizar todo o nosso visual, precisamos apenas adicionar um cabeçalho à nossa página. Mas isso é matéria da próxima aula.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Consolidando o seu conhecimento**

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Na página **index.html**, envolva o texto em uma div:

<div class="principal">

<h1>Sobre a Barbearia Alura</h1>

<p>Localizada no coração da cidade a <strong>Barbearia Alura</strong> traz para o mercado o que há de melhor para o seu cabelo e barba.

Fundada em 2019, a Barbearia Alura já é destaque na cidade e conquista novos clientes a cada dia.</p>

<p id="missao"><em>Nossa missão é: <strong>"Proporcionar auto-estima e qualidade de vida aos clientes"</strong>.</em></p>

<p>Oferecemos profissionais experientes e antenados às mudanças no mundo da moda.

O atendimento possui padrão de excelência e agilidade, garantindo qualidade e satisfação dos nossos clientes.</p>

</div>

2) E abaixo da div principal, crie a div beneficios, com a lista de benefícios e a sua imagem:

<div class="beneficios">

<h2>Benefícios</h2>

<ul>

<li class="itens">Atendimento aos Clientes</li>

<li class="itens">Espaço diferenciado</li>

<li class="itens">Localização</li>

<li class="itens">Profissionais Qualificados</li>

</ul>

<img src="beneficios.jpg" class="imagembeneficios">

</div>

3) No CSS **style.css**, remova a estilização da cor de fundo do body e passe-a para a div principal. Além disso, dê um espaçamento interno para ela:

body {

}

.principal{

background: #CCCCCC;

padding: 30px;

}

4) Mude a fonte dos itens da lista para itálico, estilize a cor de fundo da div beneficios e alinhe o texto do h2 ao centro:

.itens {

font-style: italic;

}

.beneficios {

background: #FFFFFF;

padding: 20px;

}

h2 {

text-align: center;

}

5) Na lista, modifique o seu display para inline-block e faça ela ser alinhada pela linha de cima. Além disso, altere a sua largura para ser 20% da página e sua margem externa à direita para ser de 15%:

ul {

display: inline-block;

vertical-align: top;

width: 20%;

margin-right: 15%;

}

6) Por fim, altere a largura da imagem dos benefícios:

.imagembeneficios {

width: 50%;

}

Texto

Descrição gerada automaticamente

1. **Finalizando a página**

**Transcrição**

[00:00] O que queremos agora é criar o cabeçalho da nossa página, onde estarão o nome da nossa barbearia e as informações mais importantes do nosso site.

[00:09] Para isso, no nosso HTML, antes da primeira imagem, dentro do body, vamos criar outra divisão. Essa divisão vai ter um conteúdo. Esse conteúdo vai ser o nome Barbearia Alura.

[00:33] Duas coisas sobre isso: essa divisão é a divisão do cabeçalho. Como isso é uma coisa muito comum em todas as páginas web, na última versão do HTML criaram uma tag específica para o cabeçalho. É uma tag mais semântica que diz para o navegador que aquilo ali é a primeira informação a ser apresentada.

[01:01] Ao invés de div, nós vamos usar a tag header. Ao invés de fechar div, vamos fechar o header.

[01:09] É importante destacar que header é o cabeçalho da página, e fica dentro do conteúdo, e o head é a tag padrão do HTML que serve para passarmos a informação para o navegador. Não vamos confundir as duas coisas.

[01:26] A segunda coisa sobre o cabeçalho é que queremos que o nome da nossa barbearia seja a coisa de mais destaque. Vamos entender como isso funciona no navegador.

[01:37] Salvando e recarregando a página, teremos o nome como a primeira coisa a ser lida pelo usuário. Se pensarmos em qualquer matéria de jornal ou texto, a primeira coisa que lemos é o que está no canto superior esquerdo. Ou seja, este título é o conteúdo principal da nossa página.

[02:02] Sempre que temos o conteúdo principal da página, o título principal, usamos a tag h1. Só que no nosso conteúdo usamos a tag h1 para o título Sobre a Barbearia Alura. Vamos trocar.

[02:17] Sobre a Barbearia Alura passa a ter a tag h2, o Benefícios passa a ter a tag h3, e o nosso principal título, nossa informação principal, passa a ter a tag h1.

[02:40] Se recarregarmos e voltarmos ao nosso site, veremos que ficou uma bagunça. Por quê? Usamos a tag h2 para marcar nosso CSS, usamos a tag h1 para marcar o CSS, e no cabeçalho ganhamos aquele estilo que tínhamos criado sem precisar.

[03:04] Essa é a maior dificuldade quando criamos estilos usando tags. Por isso é sempre recomendado usarmos classe para tudo.

[03:14] Vamos então modificar o conteúdo que já temos para que os estilos que criamos fiquem em tags.

[03:22] A primeira coisa no nosso CSS é que o nosso h1 tinha um alinhamento ao centro. Só que o nosso h2 também tinha a mesma informação. Ou seja, criamos dois estilos para a mesma coisa.

[03:39] Se viermos no nosso HTML, podemos transformar em uma classe, chamando essa classe de “titulo-centralizado”, copiar isso e botar essa mesma classe para nosso h3 e copiando o nome, indo no CSS, posso apagar a declaração do h2, e onde tenho a declaração para o h1, trocar ela para .titulo-centralizado

[04:20] Com isso, temos essa mesma característica sendo aplicada nos dois elementos. Se salvarmos nosso HTML CSS e recarregarmos no navegador, nosso título Barbearia Alura já está posicionado à esquerda, nosso título Sobre a Barbearia Alura, que agora é o h2 está posicionado ao centro, e nossos Benefícios também estão centralizados.

[04:48] Por fim, só para termos mais uma perspectiva visual e mexermos um pouco mais nos espaçamentos, vamos dar um espaço interno no nosso título principal para a esquerda. Ou seja, criaremos um espaçamento de 20px.

[05:09] No nosso HTML, vamos dar uma classe para ele, chamando de titulo-principal. Vamos copiar isso, salvar a página, no nosso CSS vamos criar a classe titulo-principal, que vai ter um espaçamento interno para a esquerda de 20px: padding-left: 20px

[05:38] Salvando esse conteúdo, voltando para o navegador e recarregando a página temos esse respiro do nosso título para nossa página.

[05:50] Com isso, marcamos e criamos todo o conteúdo necessário para a primeira página da Barbearia Alura.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

**Consolidando o seu conhecimento**

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Crie o cabeçalho da página **index.html**:

<header>

<h1 class="titulo-principal">Barbearia Alura</h1>

</header>

2) Altere a tag do título **Sobre a Barbearia Alura** para <h2> e a tag do título **Benefícios** para <h3>. Além disso, dê uma mesma classe para os dois:

<h2 class="titulo-centralizado">Sobre a Barbearia Alura</h2>

E:

<h3 class="titulo-centralizado">Benefícios</h3>  
3) No arquivo **style.css**, dê um espaçamento interno à esquerda para o cabeçalho, de 20px:

.titulo-principal {

padding-left: 20px;

}

4) E por fim, remova os estilos das tags h1 e h2 e centralize-os em um único lugar, através da classe titulo-centralizado:

.titulo-centralizado {

text-align: center

}

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Conclusão**

Transcrição

[00:00] Concluímos com sucesso a criação da página da Barbearia Alura.

[00:03] Tínhamos o desejo de criar essa página e passo a passo conseguimos.

[00:08] Obrigado por me acompanhar nessa jornada. Espero que você tenha aprendido bastante.

[00:14] Nossa página Barbearia Alura já tem agora todo o conteúdo que queríamos. Esse conteúdo não só está bem marcado com as tags do HTML, como está perfeitamente estilizado com o CSS.

[00:25] Vamos rapidamente revisar o que fizemos.

[00:28] No nosso HTML, falamos desde a estrutura do HTML, com as tags do DOCTYPE, HTML, head, body, como falamos também das informações que passamos para o navegador, como por exemplo o link do CSS.

[00:44] Falamos ainda de todas as tags que montam o conteúdo, desde a separação do header e das divisões, falamos de imagem, de títulos, dos textos e listas.

[00:56] Também falamos sobre como marcar esses elementos usando identificadores e classes. Pegamos essa marcação inteira e fomos para o CSS.

[01:06] No CSS, vimos como marcar com uma tag, como marcar com identificador, como marcar com uma classe. E abordamos os detalhes principais do CSS, todas as questões das dimensões, como a largura, altura, padding, a margem. Abordamos a criação de um fundo colorido ou de uma cor no elemento. Mexemos ainda no alinhamento dos elementos, no estilo da fonte.

[01:34] Passamos por isso tudo para chegar na página da Barbearia Alura.

[01:41] Espero que você tenha aprendido muito, mas tenho certeza que você está com várias dúvidas ainda. Por exemplo, como criamos outras páginas e fazemos um link entre elas? Como colocamos um formulário de contato? Um preço? Como vendemos algo da Barbearia Alura? Como podemos criar um rodapé e outros estilos básicos que vemos em todas as páginas da web? Por isso, quero que você me acompanhe na parte 2 desse curso. Espero que você aprenda ainda mais.

[02:14] Te espero lá.