**CRIANDO UMA PAGINA WEB**

A primeira coisa que vou fazer é copiar aquele texto do Word e jogar no Sublime Text. 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.

O **“.html”**, diz que este arquivo é uma página web. Salvando este arquivo **index.html**, se eu der dois cliques ele vai abrir direto no navegador.

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.

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.

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.

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.

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”</h>. Assim ele fecha o conteúdo.

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.

Com isso aqui, nós já temos um título e três parágrafos.

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.

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.

Isso é o primeiro passo para modificar e estruturar nosso conteúdo.

Agora, vamos evoluir o conteúdo marcando todas as informações do meio do texto que colocamos em destaque.

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.

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”.

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.

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.

**RESUMINDO:**

Nesta aula, aprendemos:

· Uma introdução ao HTML e às suas ***tags***

· Como definir o título e os parágrafos de um texto

o Utilizando as *tags* **<h1>** e **<p>**, respectivamente

· Como dar destaque para algumas informações do texto, deixando-as em **negrito**, utilizando a *tag* **<strong>**

· Como dar ênfase para algumas informações do texto, deixando-as em *itálico*, utilizando a *tag* **<em>**

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.

// DOCTYPE não é uma tag de conteúdo, passa a informação de qual versão do HTML está sendo usada na página, por isso nao precisamos fechar ela.

[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.

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.

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.

Na estrutura correta do HTML, inserimos as informações que queremos passar para o navegador no head e as *tags* de conteúdo no body.

COMEÇANDO A USAR O CSS – mexendo na apresentação visual 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.

O código ficou da seguinte forma:

<meta charset="UTF-8">

<title>Barbearia Alura</title>

</head>

<body>

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

<p style="text-align: center">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; text-align: center"><em>Nossa missão é: "<strong>Proporcionar auto-estima e qualidade de vida aos clientes</strong>".</em></p>

<p style="text-align: center">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>

Para saber mais: CSS externo

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 contem 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>

**COPIAR CÓDIGO**

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

ORGANIZANDO O ESTILO DA PAGINA

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.

[00:17] Quais são as características dessa forma de criar o CSS? A primeira delas é que não usamos o nome CSS. Usamos uma propriedade chamada style. A segunda é que como este elemento é único e está marcando somente um conteúdo, toda vez que quisermos colocar a mesma configuração em elementos parecidos, vamos precisar repetir esse código. Por exemplo, todos os nossos parágrafos tem essa representação do alinhamento ao centro do texto.

[01:01] Isso não é muito legal. Precisamos toda hora repetir o mesmo código. Aqui temos só três parágrafos. Imagine se tivéssemos vinte, cinquenta, se fosse a descrição de uma receita ou a descrição de como se constrói um prédio, se fosse a descrição de um artigo científico. Teríamos muitos parágrafos de texto, e é desumano e inviável ficar adicionando informações pontualmente a cada parágrafo. Pior ainda seria se a partir de agora quiséssemos mudar qualquer uma dessas informações e alinhar esse texto à esquerda, ao invés de ao centro.

[01:45] Para isso, temos uma forma de fazer esse conteúdo ser repetível nesta página. É a segunda forma de apresentarmos o CSS.

[01:55] Para fazermos isso, temos a opção de dentro do head (recordando: o head é o lugar onde passamos as informações para o navegador, e não onde colocamos as informações que queremos exibir). Dentro do head, adicionamos a tag style. Ela é uma tag que abre e fecha. Dentro da tag, podemos colocar marcações de CSS referentes aos elementos que temos no nosso HTML.

[02:30] O que eu quero fazer agora é que todos os parágrafos sejam alinhados ao centro. Da mesma forma como colocamos no inline, só que agora quero escrever uma vez só.

[02:41] Para fazer isso coloco a tag que quero usar, que é a tag do parágrafo, abro e fecho chaves, e dentro coloco a propriedade do CSS que eu quero, que é o text align.

[02:56] Vou recortar daqui, para sermos obrigados a mexer nesse conteúdo, e agora o meu parágrafo recebe a propriedade text align center.

[03:08] “Mas Pedro, só um dos parágrafos?”. Não, todos os parágrafos desta página.

[03:15] Se salvarmos isso e repararmos no primeiro parágrafo, ele não tem mais essa descrição do text align center. Vamos voltar no navegador e recarregar a página.

[03:30] Reparem que a apresentação não foi alterada. Ou seja, essa propriedade do text align está sendo aplicada a todos os parágrafos. E como temos no primeiro parágrafo esse elemento removido, ela está sendo aplicada uma vez, mas nos parágrafos seguintes estamos aplicando duas vezes, tanto no inline quanto na tag style.

[03:56] Vamos remover essa descrição dos parágrafos.

[04:04] O que esperamos agora é que esse conteúdo não seja alterado. Ou seja, que a apresentação seja a mesma com tudo centralizado.

[04:14] Vamos salvar o arquivo, voltar ao navegador e recarregar a página.

[04:22] A primeira forma é única e exclusivamente para esta tag, tanto que o tamanho da fonte só está alterado na linha do meio, no parágrafo da nossa missão. E o parágrafo geral, onde temos o estilo de todos os elementos, está sendo aplicado nos três parágrafos.

[04:53] Já falamos do inline e já falamos da tag style. A terceira forma, que é a mais comum de encontrarmos a criação e apresentação do CSS é com um arquivo externo. Para isso, vamos criar um novo arquivo. Se quiserem um atalho, é ctrl+n.

[05:17] Vamos salvar esse arquivo na mesma pasta com o nome style.css. O nome do arquivo vai ser style, onde vamos guardar todo o nosso estilo, e a extensão do arquivo é CSS.

[05:34] Agora que temos o arquivo criado na mesma pasta, no nosso HTML vamos fazer uma referência para ele. Vamos fazer com que o HTML busque o arquivo externo e passe a ler e aplicá-lo na nossa página.

[05:52] Para fazer isso, vamos usar a tag link, que é justamente a ligação de um arquivo para o outro. Vamos dizer o relacionamento desse link com a propriedade rel e o valor stylesheet, ou seja, folha de estilo, e vamos dizer onde o arquivo está, href, que é o endereço de referência:

[06:28] Agora que temos essa ligação, podemos simplesmente recortar o conteúdo da tag style e jogar no nosso arquivo .css.

[06:43] Na hora de copiar e colar, ele muda um pouco a endentação. Eu vou corrigir, salvar o arquivo CSS, o arquivo HTML e voltar para o meu navegador. O esperado é que quando eu recarregue a página tudo esteja centralizado e o valor da fonte não esteja alterado.

[07:05] Agora que alteramos a posição do nosso código, a visualização não foi modificada.

[07:17] Vamos recapitular rapidamente. Temos a possibilidade de colocar um elemento com uma micro alteração, com a possibilidade de colocar um CSS inline, ou seja, na mesma linha, e o mais importante do inline é destacar que ele só vai ser aplicado no elemento onde foi criado.

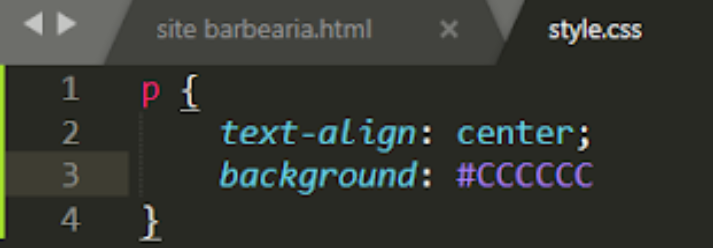
[07:43] Temos a possibilidade de colocar a tag style, onde tudo que fizermos vai ser aplicado na página inteira, economizando muita escrita de código; e temos a possibilidade de adicionar um arquivo externo ao HTML, em que teremos todo o CSS.

[08:05] Essa é a forma mais comum de ser feita, criar um arquivo externo, porque não só o código pode ser aplicado nesta página, como em qualquer outra página em que criarmos um link para esse CSS. Se imaginarmos um site em que temos vinte páginas diferentes, todas elas terão o mesmo estilo. É isso que buscamos quando estamos criando um bom código.

[08:34] Nesta aula, vimos como aplicar o CSS das três formas diferentes. Na próxima aula, vamos evoluir ainda mais essa apresentação, entendendo ainda mais como funciona o CSS e aplicando detalhes diferentes.

EXERCICIO:

Aprendemos que o p no início do arquivo CSS significa que as propriedades serão aplicadas em todos os parágrafos. Após escrever o código a seguir, como ficarão os parágrafos do site?



Resposta: Texto centralizado, cor do fundo em cinza.

O texto está centralizado, como podemos ver pelo center em text-align, e o background está em linguagem hexadecimal na cor cinza.

# **Mesmo CSS em várias páginas**

A maneira correta para declarar um CSS que pode ser usado em várias páginas diferentes é:

Em um arquivo externo

Alternativa correta! Só com um arquivo externo, e fazendo referência a esse arquivo, podemos usar o mesmo CSS em páginas diferentes.

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.

EXERCICIO:

Como funciona o estilo em cascata do CSS?

Tudo que eu adiciono no elemento pai reflete nos elementos filhos.

Como é uma folha de estilos em cascata, a cascata vai descendo, e o que eu faço no elemento pai reflete para o elemento filho.

# **Representando cores no CSS**

# De que maneiras eu posso representar cores no CSS?

Hexadecimal, RGB e com os nomes das cores

Alternativa correta! Podemos representar a cor vermelha com o nome red, o hexadecimal #FF0000 e o RGB rgb(255,0,0).

Nesta aula, aprendemos:

* A mexer na apresentação dos textos
  + No alinhamento deles (text-align)
  + No tamanho da fonte (font-size)
  + Na cor de fundo (background)
  + Na cor do texto (color)
* CSS *inline*
  + Na linha onde temos a nossa *tag*, adicionamos a propriedade do CSS
* A tag **<style>**
  + Dentro da *tag*, podemos colocar marcações de CSS referentes aos elementos que temos no nosso HTML
* A apresentação do CSS com um arquivo externo
* Como funciona o estilo em cascata do CSS
* Como importar um arquivo externo de CSS dentro da nossa página HTML
* Como representar cores no CSS
  + Através do nome da cor
  + Através do seu hexadecimal
  + Através do seu RGB

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.

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.

# **O caminho da imagem**

[**PRÓXIMA ATIVIDADE**](https://cursos.alura.com.br/course/html5-css3-primeiros-passos/task/99674/next)

# **Como inserir imagens de forma correta?**

Para que a imagem apareça de forma correta, fique atento nesses pontos:

* Nome do atributo da tag é src

<img src="banner.jpg">

**COPIAR CÓDIGO**

O atributo **src** caso escrito de outra forma, a imagem não será exibida. Por exemplo:

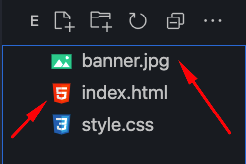
<img crs="banner.jpg">

**COPIAR CÓDIGO**

Neste exemplo de código, a imagem não será exibida, pois não existe nenhum atributo da tag img chamada crc ou crs. **O atributo correto que especifica o caminho da imagem** é src, que significa *source* do inglês ou **fonte** em português. Sendo assim, fique atento com o nome do atributo.

* Imagem no mesmo nível do arquivo

Neste treinamento, recomendamos que as imagens do projeto estejam no mesmo nível do arquivo index.html, ou seja, as imagens do projeto devem estar na **mesma pasta**, como ilustra a imagem abaixo.



**Sintaxe para adicionar img**

**<img src = “endereço.formato”>, ou <img src="endereço">**

**A *tag* img tem a propriedade src, onde adicionamos o endereço**

**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. 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.**

# **Alterando a largura de um elemento**

# **Qual a propriedade que usamos para alterar a largura do elemento?**

O width é a propriedade que devemos manipular para alterar a largura do elemento.

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

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.

ESTRUTURA DE UMA LISTA(NÃO ORDENADA) HTML

No HTML, trabalhamos com listas ordenadas tag ol e listas não ordenadas tag ul, cada uma delas tem um uso específico. Nas listas ordenadas, a ordem dos itens não pode ser alterada, como em uma receita, por exemplo. Já nas listas não ordenadas, a ordem é flexível, e não altera o resultado final. Qual a maneira correta de montar a seguinte lista não ordenada no nosso HTML?

* Item lilás
* Item azul
* Item vermelho

<ul>

<li>Item lilás</li>

<li>Item azul</li>

<li>Item vermelho</li>

</ul>

Alternativa correta! A *tag* para a lista é a ul e cada item da lista precisa ser marcado com um li.

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.

CRIANDO BLOCOS DE CONTEUDOS

Como criar vários blocos de conteúdo no HTML?Blocos de código dentro de vários <div>

Alternativa correta! A *tag* div é a *tag* específica para dividir o conteúdo. Ela não tem outra função além dessa.

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.

Quando usamos a propriedade class no nosso código? Quando queremos repetir o CSS

Alternativa correta! A class serve para replicarmos a aplicação do nosso CSS, sem precisar copiar e colá-lo em vários elementos. Aplicamos a classe para isso.

Nesta aula, aprendemos:

* A trabalhar com listas não-ordenadas e listas ordenadas
  + Para cada um dos itens da lista, utilizamos a *tag* **<li>**
* O conceito das **classes** no CSS
  + Elas servem para marcar itens, que são repetíveis
* Como referenciar uma classe no CSS
* Divisões de conteúdo, utilizando a *tag* **<div>**
* Os comportamentos ***inline*** e ***block***

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.

**Tag do cabeçalho**

**A *tag* header é uma *tag* de conteúdo do HTML, e não uma *tag* de estrutura. Não confunda head com header.**

# **Barbearia Alura**

**O que significa class=titulo.principal em nosso cabeçalho?**

**É a definição de uma classe que será utilizada para estilizar a tag <h1> no CSS**

**Nesta aula, aprendemos:**

* **O conceito de cabeçalho da página e como criá-lo**
* **Que o cabeçalho da página deve ter mais destaque**
* **Que não é recomendado criar estilos usando *tags***
  + **O ideal é usarmos classes para tudo**

**RESUMO:**

**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.**