Parte superior do formulário

Parte inferior do formulário

**Guia de estilos**

**Montando páginas com componentes**

**Faça esse curso de HTML e CSS e:**

* Entenda o processo de desenvolvimento de páginas web a partir de uma ferramenta de layout, como Figma
* Aprenda a estruturar o projeto com arquitetura de CSS
* Conheça o Emmet e IntelliSense CSS, ferramentas para agilizar a escrita de códigos
* Melhore a manutenção de códigos com variáveis CSS
* Crie componentes a partir de um guia de estilos
* Desenvolva uma página utilizando componentes

**Aulas**

[Figma e guia de estilos](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10693/tasks)

[0 / 723min](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10693/tasks)

* + Introdução
  + Preparando o ambiente
  + Recolhendo recursos
  + Primeiros passos
  + Criando um componente
  + Faça como eu fiz
  + O que aprendemos?

[Arquitetura e ferramentas](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10694/tasks)

[0 / 923min](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10694/tasks)

* + Projeto da aula anterior
  + EMMET e Intellisense
  + Melhorando o desenvolvimento
  + Variáveis CSS
  + Valores complexos
  + Preparando o ambiente
  + Mais ferramentas CSS
  + Faça como eu fiz
  + O que aprendemos?

[Os primeiros componentes](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10695/tasks)

[0 / 929min](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10695/tasks)

* + Projeto da aula anterior
  + O input
  + Para saber mais: Padrão BEM CSS
  + O cartão
  + O título
  + Para saber mais: Componentes
  + Variações de componentes
  + Faça como eu fiz
  + O que aprendemos?

[Outros componentes](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10696/tasks)

[0 / 834min](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10696/tasks)

* + Projeto da aula anterior
  + O item
  + O produto
  + Styleguide - O vídeo
  + O cabeçalho
  + Título do vídeo
  + Faça como eu fiz
  + O que aprendemos?

[Iniciando a página](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10697/tasks)

[0 / 722min](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10697/tasks)

* + Projeto da aula anterior
  + O rodapé
  + Planejando a Home
  + Usando componentes
  + Documentação dos componentes
  + Faça como eu fiz
  + O que aprendemos?

[Encerrando a página](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10698/tasks)

[0 / 926min](https://cursos.alura.com.br/course/guia-estilos-paginas-componentes/section/10698/tasks)

* + Projeto da aula anterior
  + As primeiras seções
  + Seção produto
  + Seção vídeo
  + Documentação dos componentes
  + Faça como eu fiz
  + Projeto do curso
  + O que aprendemos?
  + Conclusão

# 01Introdução

[00:00] Meu nome é Matheus Roberto e eu vou acompanhar você durante essa jornada do curso de Style Guides CSS.

[00:05] Nesse curso vamos ver o que são os Style Guides e qual a importância de se ter um style guide ou guia de estilos. Vamos falar também sobre temas e utilizações que se repetem na página, vamos ver sobre cores, as cores que se repetem, a paleta de cores; tipografia usada na página e até formatos de elementos, como, por exemplo, aqueles botões que vão se repetindo durante toda a página, inputs, botões, cartões e até outros conjuntos de informações.

[00:36] Vamos ver também como vamos fazer para criar componentes que podemos reutilizar durante todo o nosso projeto. E com isso vamos aprender também ferramentas que ajudam no desenvolvimento da página, como EMMET e IntelliSense CSS.

[00:51] Vou falar também sobre a arquiteturas CSS, de como os nossos componentes devem estar dispostos dentro do nosso projeto, então criar pastas e separar esses arquivos conforme uma arquitetura definida, arquivos base de CSS, componentes e aí as nossas seções.

[01:11] Nesse curso vamos fazer, além de inpage aqui, a home page da Casa Verde, a parte mobile. Aqui nós vamos montar cada um dos componentes que serão utilizados nessa página, cabeçalhos, rodapés, cartões, botões, inputs e assim por diante. E nós vamos montar cada uma dessas seções com os componentes que vamos criar durante o curso. Então fica comigo e vamos fazer nosso projeto funcionar.

# 02Preparando o ambiente

Para este curso, você precisará baixar alguns arquivos. O primeiro é o arquivo do projeto no Figma.

O link para download do arquivo [Styleguides.fig](https://caelum-online-public.s3.amazonaws.com/1808-guia-de-estilos/01/Aula1-Styleguides.fig" \t "_blank).

# 03Recolhendo recursos

[00:00] Para começar, vamos importar o projeto dentro do Figma. Vou clicar aqui no canto superior direito do Figma, “Import” e agora vou importar o arquivo “Styleguides.fig”.

[00:12] Caso você não tenha esse arquivo, você pode pausar esse vídeo, voltar no exercício anterior, onde liberamos o arquivo para download. Então basta clicar em Styleguides agora, “Open” e o Figma vai importar o arquivo.

[00:25] Depois que o Figma importar arquivo, vamos clicar em “Done” e abrir agora “Styleguides”. Carregou e agora um passo muito importante no desenvolvimento o projeto é analisar todo o layout que nós temos. Temos aqui duas páginas, desktop e mobile, e dentro temos várias páginas, que são a home, várias imagens, olha só a tipografia, ícones, design de produtos, um carrinho. Agora mobile, bastante cores amarelas, tipografias são só aparentemente duas. Legal.

[01:05] Depois que analisamos toda a página nós vamos recolher todos os recursos necessários para poder montar a página. Nesse curso vamos focar só na versão mobile e nessa primeira página, que é a home, então todas as imagens que nós vamos baixar vão ser só dessa página. Então como vamos fazer isso?

[01:25] Por exemplo, aqui no cabeçalho, eu clico duas vezes na logo, ele vai selecionar aqui no canto e na barra lateral direita eu clico em “Export logo”, é o último dropdown que tem. E basta clicar aqui. E agora onde vou salvar esse arquivo?

[01:42] Primeiro precisamos criar uma pasta desse arquivo para poder salvar as imagens, eu já deixei criada uma pasta aqui chamada “casaverde”, dentro dela não tem nada, é só uma pasta vazia, mas nessa pasta vai ser onde vai conter todos os arquivos do projeto. Então vou criar uma pasta aqui agora, dentro de “casaverde”, chamada de “assets”.

[02:02] Dentro de “assets” vão ficar todas as imagens CSS e também nossos componentes. Vou abrir aqui assets e dentro vou criar outra pasta chamada “img”, que é onde vou guardar as imagens. agora em img, basta clicar aqui e salvar o logo.

[02:19] Próximo vai ser o ícone do carrinho, então eu clico aqui ou até mesmo na barra lateral esquerda, eu clico em “carrinho”, e eu posso clicar aqui, “Export carrinho”, na barra da direita.

[02:32] Só que isso é um ícone agora, então para deixar mais separado eu vou criar outra pasta chamada “ícones” e dentro de “ícones” eu vou salvar o carrinho e tudo o que for relacionado a ícones.

[02:43] Próxima imagem vai ser essa imagem da planta, “Export”, como essa imagem é um pouco mais complexa não vou salvar lá como svg, vou salvar como png. Eu vou mudar até o nome dela, porque esse nome não faz muito sentido, vou colocar “planta-chamada”. Salvei.

[03:04] Próximo ícone é o de carta, “Export mail”, svg, beleza, “ícones”. Seta, carrinho. E agora as imagens das plantas, vamos uma por uma selecionar todas. Vamos lá, seleciona, export.

[03:34] Em alguns casos as imagens vão estar com o nome meio estranho, vamos ter que arrumar para cada uma delas. Nesse caso, apagar esse “1” espaço do lado, deixar “produto-01”, e fazer isso com todos os outros produtos.

[03:58] Agora pegar o ícone da seta, salvar como svg, ele é um ícone, deixar aqui “seta”. O botão de play agora também, export, svg, chamar de “play.svg”.

[04:23] Essas imagens de vídeos nós não vamos pegar, porque vamos usar outra ferramenta depois para poder substituir essas imagens de fundo. Vamos para o rodapé agora. A logo é a mesma, então “Redes”, “facebook”, “twitter”, “instagram”.

[05:00] Agora essas imagens de splash no fundo, então clica duas vezes nesse splash e salvar, vamos lá. Como vai ser a imagem do fundo, então eu vou deixar aqui na “img”, na raiz mesmo, e chamar de “splash-rodape”. E agora a do topo também, export, vou chamar de “splash-chamada”.

[05:35] O que falta agora são só essas linhas e a textura do fundo. Vamos ver, descer tudo, aqui. Perfeito. No canto esquerdo, aqui na barra lateral, elas já estão separadas, a imagem de fundo é só essa aqui, o que podemos fazer é mesclar todas elas, mas não compensa, já que ela já se repete mesmo, não tem por que mesclar. Então clicar aqui, por exemplo, em imagem 1, que é a textura, export como png mesmo, agora colocamos aqui “fundo-textura”.

[06:19] E agora tem as linhas, e são quatro linhas, então é difícil conseguir recortar essa imagem, as quatro linhas de uma vez só, então vamos pegar as quatro aqui e nós vamos mesclar, cadê? “Flatten”, é aqui a opção. Eu selecionei as quatro, botão direito e “Flatten”.

[06:41] Agora ele virou como se fosse um elemento só e agora vamos exportar esse elemento. Clicar aqui em “Vector”, export, pode ser svg também, nesse caso não vai fazer muita diferença, clicar em png, “Export Vector” e vamos chamar de “fundo-linhas”.

[07:10] Já pegamos todas as imagens, mas onde estão as cores? Vamos ter que pegar cada cor, tem esse cinza aqui, que cinza é esse? Eu vou ter que ficar clicando aqui, descobrindo, pegando o valor, 2020. Esse amarelo aqui é outro amarelo, cadê? Qual é o amarelo desse botão? É esse amarelo? Tem algumas cores mais claras aqui. Eu vou ter que ficar olhando elemento por elemento?

[07:38] Quando a equipe de design faz um projeto como esse, eles criam outra página aqui no Figma, por exemplo, chamada guia de estilos. Tudo o que vamos precisar de ferramenta para poder usar na nossa página vai estar separado aqui. Então clicando na “Guia de estilos” aqui na barra lateral esquerda.

[07:55] Nós temos todas as cores que são usadas no projeto, aquele preto, o cinza, o branco e o amarelo. Também como, por exemplo, tipografia, os tamanhos das fontes, qual tipografia é essa, quando tem destaque, botões, como funciona o botão. Também temos espaçamentos, espaçamento também do mobile e assim por diante. É uma ferramenta muito importante.

[08:21] As cores estão aqui mesmo, então toda vez que tivermos dúvida para sabermos que cor vamos usar, nós voltamos aqui e pegamos essa cor.

[08:27] E agora quais são as fontes que temos que usar? Pela tipografia aqui, nós temos só dois tipos de fontes, vamos ver esse, ele é Elsie Swash Caps, e ele está como black, que é o nível 900.

[08:46] E agora esse de baixo, que vai ser Montserrat bold e esse aqui, que é o P, vai ser Montserrat regular. Então nós temos que pegar três fontes, a Montserrat regular, que é 400 de peso, a Montserrat bold, que é 700 de peso, e a Elsie blá blá blá, que tem o peso 900.

[09:19] Onde vamos conseguir essas fontes? Vamos usar uma página que conseguimos pegar fontes gratuitas, que é o Google Fonts. Vou abrir o navegador e digitar “fonts.google”. Agora aqui vamos pegar nossas fontes, deixa eu dar mais um zoom para enxergar melhor.

[09:41] Aqui na barra de busca eu vou escrever aquela elsie não sei o que lá, Elsie Swash Caps. Cliquei. E quais são as tipografias que você me orienta, aqui só tem duas, a regular 400 e a black 900, que é a que queremos. Eu clico em “Select this style”, selecionar esse estilo e apareceu essa barra lateral.

[10:09] Vamos clicar na logo do Google Fonts para voltar para a home e agora a “montserrat”. Está aqui, nós criamos os pesos 400 e 700, então desci aqui, regular, queremos essa. E agora bold 700, está aqui, criamos essa também.

[10:30] Depois que já selecionamos todas as fontes, aqui na barra lateral direita que vai abrir, clicar em “Embed” e ele vai me dar um código html pronto. Só que onde vamos jogar esse html pronto? Primeiro temos que criar nossa página html.

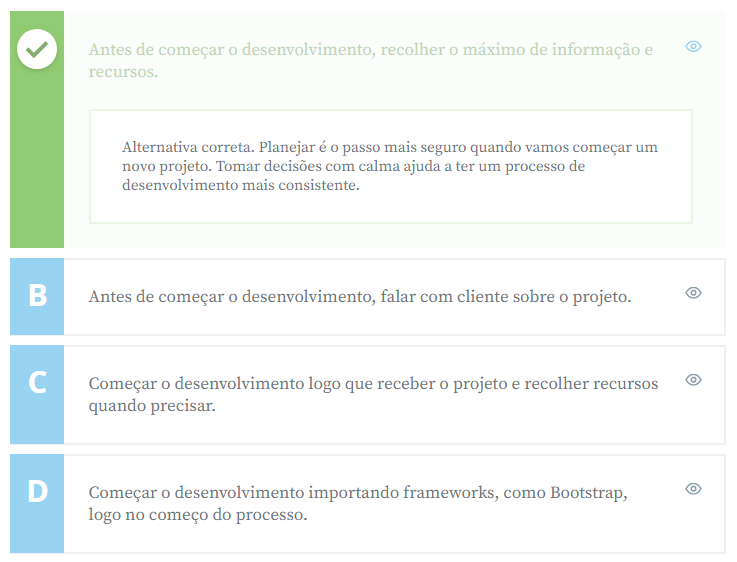
[10:47] Eu vou abrir o Visual Studio Code, que é o editor de texto que eu vou usar durante todo esse curso, e vou abrir aqui, “File”, “Open Folder” e vou selecionar aquela pasta que nós criamos, a “casaverde”. Cliquei aqui, “Select Folder”, e já aparece toda a estrutura dos nossos arquivos. Botão direito na barra lateral, “New File”, eu vou chamar de “index.html”. Fechei a barra lateral e agora vou colocar toda a estrutura básica de um arquivo html.

[11:19] Vamos lá, “”, “”, dentro de head vamos ter a tag “”. Temos a tag “

# 04Primeiros passos

Quando recebemos um projeto como esse que a equipe de design mandou para nós, podemos colocá-lo em prática de algumas maneiras diferentes. Mas a prática mais segura é a de planejar e avaliar antes de tomar decisões.

Qual alternativa expressa melhor a maneira mais segura de desenvolvimento?



# 05Criando um componente

[00:00] Depois que recolhemos todos os assets, as imagens, a tipografia, agora podemos criar o nosso primeiro componente. Por que vamos criar um componente? O que ajuda a criar um componente? Vamos abrir o Figma.

[00:14] Componentes ajudam bastante na parte de desenvolvimento, nós ganhamos tempo com o desenvolvimento quando temos muitos elementos que se repetem, como, por exemplo, esse botão dessa newsletter, ele aparece várias vezes em outros lugares de outras páginas, aparece aqui na home, aparece no produto mobile, carrinho overlay, aparece na modal aqui, ele vai se repetindo. Essa parte de produto também, tem a foto do produto, tem o nome do produto, o preço e o botão comprar.

[00:43] Esse conjunto de informações vai se repetir várias vezes em várias páginas diferentes, ao invés de criarmos uma estilização para cada um desses elementos e ficar repetindo-os com “Ctrl + C”, por exemplo, em várias páginas, nós criamos uma vez só e toda vez que formos criar esse componente só precisamos lembrar da estrutura dele. Depois que soubermos a estrutura é só aplicar as classes e toda a estilização já aparece por conta, só precisamos focar depois em semântica.

[01:17] Vamos começar, por exemplo, com esse botão aqui em newsletter. Vamos lá. Primeira pergunta é: o que faz esse botão ser um botão? Ele tem o fundo amarelado, ele tem essa cor branca, a tipografia dele é bold e o texto está centralizado.

[01:36] E além disso, eu não sei se dá para ver, mas ele tem um efeito de sombra aqui. Vamos ver no guia de estilos. Esse botão tem um efeito de sombra, que é um drop shadow, 10 pixels de deslocamento horizontal, 10 pixels de deslocamento vertical, 30 pixels de blur e 30% de opacidade.

[02:09] Então já sabemos o que faz esse botão ser um botão, agora podemos componentizar isso. Vou abrir o Visual Studio Code, vou criar agora uma pasta CSS, que é onde vão ficar todos os nossos arquivos CSS de componentes. Dentro de “assets”, “New Folder”, “css”, dentro de “css” agora, abrir aqui pasta “componentes” e agora criar um arquivo chamado “botoes.css”.

[02:37] Fechei a barra lateral e agora vamos de novo, criar a classe “.botao” e dentro de botão vamos criar agora todas as estilizações que fazem esse botão ser um botão, “background-color:”, é aquele amarelado, qual era a cor mesmo? Vou voltar para o Figma, vamos ver, é esse amarelo, o que é esse amarelo? FFCB47. Vamos ver se eu lembro disso, “#FFCB47;”, ok.

[03:09] Agora elemento do texto, “center;”, ele tem agora a fonte Montserrat, então é “font-family: ‘Montserrat’,” caso não tenha Montserrat vai ser uma “sans-serif;”. O peso da fonte também é diferente, ele tem aquele bold, e a cor também é branca, então “color: #FFFFFF;”.

[03:36] Agora colocar os espaçamentos, “100%”, colocar um “padding-top:”, que nem é o padding real, mas só para termos alguma ideia de distância, para o texto não ficar muito colado, “padding-bottom:”, depois vamos arrumar isso. “1rem;”.

[03:54] Agora colocar uma “box-shadow:”, era “10px” de deslocamento horizontal, “10px” de deslocamento vertical, “30px” de blur e agora é a cor amarelada mesmo, então é essa, mais 30%, que dá mais ou menos 3D lá no hexadecimal.

[04:17] E agora, já que vamos colocar isso numa tag bottom, ela tem uma borda já padrão, nós não queremos essa borda, então “border: none;”. Pronto, criamos nosso componente. Nós temos aqui tudo que forma um botão. Agora vamos testar.

[04:35] Eu venho no “index.html” e agora temos que importar esse arquivo botoes.css. Criei aqui “. Dentro de “” agora vamos criar nossa tag “” e agora dentro do button nós vamos colocar “Assinar Newsletter”. Salvei o arquivo css e salvei o arquivo html. Agora eu vou abrir isso no navegador.

[05:23] Venho no navegador, está aqui o botão e vou abrir no modo device mode, então “Ctrl + Shift” no meu navegador, ele já vai abrir no device mode, deixei a largura da tela de 360 pixels. Só dar um zoom para enxergar melhor. Nosso botão está aqui, ele está pronto. Ele está com a sombra, está com a cor de fundo, está com a cor da letra, perfeito.

[05:50] Mas qual é a vantagem disso? Se eu mudar a tag, voltar aqui, criar uma nova tag, por exemplo, “p”, jogar essa “class=’botao’” também nela, “Eu sou um p”, por exemplo, salvei o arquivo html, voltei para o navegador sem recarregar e, olha só, mesmo que seja um p, ele tem um formato de botão.

[06:15] A vantagem de criarmos componentes e usar esses elementos é que não precisamos mais nos preocupar realmente com a estilização em si, só precisamos nos preocupar com a estrutura e com a semântica, e facilitar agiliza muito na hora de fazermos o nosso trabalho.

[06:32] Só que é o seguinte, nós criamos esse componente botão que é um componente relativamente simples, ele tem pouca coisa, foi só uma classe que nós criamos e foi só uma estilização rápida. Só que quantos componentes mais vamos ter na página? Será que eles vão ser mais complexos que o botão? Vamos ter que decorar todas as classes que existem no projeto? Vamos ter que montar os cartões também de produtos, por exemplo, na mão?

[06:58] Para evitar esse tipo de problema, evitar escrever tudo isso na mão, ter essa dificuldade de ficar lembrando, indo e voltando, nós vamos ver algumas ferramentas de otimização. Então fique comigo que vamos ver as ferramentas no próximo vídeo.

# 06Faça como eu fiz

Praticar ajuda bastante no aprendizado de um novo conceito. Assim, é muito importante que você implemente o que foi apresentado nesta aula.

Não deixe de sanar suas dúvidas antes de dar continuidade ao curso. Estaremos te esperando no fórum da Alura caso alguma dúvida surja.

# 07O que aprendemos?

Nesta aula, aprendemos:

* Importar o projeto no Figma
* Recolher recursos para o projeto
* Preparar a estrutura de arquivos
* Criar o componente de botão

# 01Projeto da aula anterior

Caso queira começar daqui, você pode baixar o projeto da aula anterior nesse [link](https://github.com/alura-cursos/styleguide-css/archive/Aula1.zip).

# 02EMMET e Intellisense

[00:00] Existem várias ferramentas que podem facilitar nosso trabalho, só que vamos usar duas ferramentas. Antes de falar quais são essas ferramentas, vamos primeiro pensar quais eram os problemas que nós tínhamos antes.

[00:11] O primeiro problema era exatamente que na hora de escrever as tags nós tínhamos que colocar o símbolo de menor que, escrever “button”, depois “class=”, aspas duplas, “botao”, fechar e assim por diante.

[00:28] Num elemento como o botão, que é mais simples, é só o nome do elemento, depois a classe, e é só uma classe que tem dentro dela, ok, dá para fazer, mas quando temos estruturas muito mais complexas isso já se torna um problema, começamos a perder bastante eficiência do trabalho.

[00:47] A primeira ferramenta que vamos ver para fazer nosso trabalho é o plugin Emmet. O Emmet já vem instalado em alguns editores de texto, como, por exemplo, o Sublime, o Atom e, no meu caso, o Visual Studio Code.

[01:02] Já tenho instalado o plugin, então eu vou usá-lo. Mas como usa esse plugin? Vamos usar o exemplo do botão. Nós temos um “button” e esse button vai ter que ter uma classe. Para poder indicar que eu tenho uma classe dentro desse button, eu simplesmente escrevo “.botao”. Pressiono “Enter”.

[01:25] Olha que legal que o Emmet fez, ele criou a estrutura do button e já adicionou a classe junto e, inclusive, abriu e fechou a tag. O que só falta para fazermos aqui é colocar o “Conteúdo”.

[01:39] Vamos fazer uma coisa mais complexa agora, vamos criar uma seção que vai ter a classe container e dentro dessa seção vai ter um parágrafo com a classe texto e um button com a classe botão. Como vamos montar isso no Emmet?

[02:00] Escrever “section”, que é a nossa seção, ela vai ter a classe container, então eu uso o operador, o ponto, para dizer que ela tem uma classe “container”, e agora como eu faço para dizer que tem alguma tag dentro dessa section?

[02:15] Para isso vamos usar o operador de maior que, “>”. E agora tudo o que vem depois significa que são tags que estão dentro dessa section, vamos ter a tag p com a classe texto e agora nós temos um botão do mesmo nível do p, então ele não está dentro desse p, não é uma tag filha do p, é do mesmo nível. Para indicar isso nós usamos o operador de mais. Então “p.texto+”, vamos ter agora um “button.botao”. Então vamos revisar aqui.

[02:56] Nós temos uma section, uma seção com uma classe container, dentro dessa section vamos ter uma tag p e uma tag button, a tag p com a classe texto e a tag button com a classe botão. Clico aqui no final, pressiono “Enter” e não foi.

[03:18] Nesses casos, quando começamos a clicar fora do comando do Emmet, ele perde essa funcionalidade de pressionar “Enter” e já resolver para nós o problema. Vou voltar aqui, quando for esse caso, eu pressiono “Ctrl + espaço” no final da linha, me aparece o menu de contexto aqui embaixo, aí sim posso pressionar “Enter” que ele já resolve para mim.

[03:40] Nós conseguimos construir uma estrutura mais complexa com menos linhas de código, e conforme vamos treinando o escrever com Emmet, ficamos mais proficientes em escrever códigos mais complexos e muito mais rápido.

[03:57] Um problema foi resolvido, qual era o outro problema que tínhamos? Classes botão, container. Um certo tempo depois do projeto podemos ter muitos componentes e muitas classes para lembrar, então começa a ficar complicado esse processo de ficar desenvolvendo e, por exemplo, aqui na section, vamos supor que eu tivesse várias outras classes que eu pudesse colocar dentro dessa section, qual é a que eu preciso mesmo? Era container, conteúdo? Começa a complicar lembrar o nome das classes.

[04:34] Para isso vamos ver a nossa segunda ferramenta, que é o CSS IntelliSense. No caso do Visual Studio Code, ele tem um nome diferente, mas a ideia é que o plugin faça esse IntelliSense baseado em classes CSS. Cada editor de texto vai ter o seu próprio. O Sublime já tem esse IntelliSense de CSS instalado, mas o Visual Studio Code não, então temos que baixar. E como baixa?

[05:05] No Visual Studio Code, na barra lateral à esquerda, onde tem esses ícones, eu vou clicar no ícone chamado “Extensions”, que é um monte de quadradinho com um quadradinho saindo voando. Cliquei aqui, tem a barra de busca aqui em cima e dentro dessa barra de busca eu vou escrever “HTML CSS support”. Provavelmente deve ser esse primeiro que aparece, que é o do ecmel. Cliquei, agora “Install”.

[05:34] Pode ser que exista uma situação em que o Visual Studio Code peça que você reinicie o programa para poder fazer o plugin funcionar. No meu caso não precisou, mas caso o Visual Studio peça para você, é bom você restartar o Visual Studio Code. E vai ter a opção aqui, inclusive, no botão do lado, se for o caso.

[05:54] Instalado o plugin, como eu uso o plugin? Vou voltar aqui, apagar esse monte de coisa que eu fiz e vou refazer o botão. Vou escrever “button.” e só de ter pressionado o ponto, aqui embaixo apareceu uma sugestão, “botao”.

[06:15] Isso facilita muito no momento em que vamos desenvolver, que não lembramos os nomes das classes, o próprio IntelliSense, aquele plugin, já nos dá sugestões de classes que existem no documento. Quanto mais classes formos criando depois, no processo de desenvolvimento, mais opções vai ter para nós.

[06:32] E, por exemplo, eu quero uma coisa voltada com botão, eu escrevo “bo” e ele já dá sugestões baseado no que eu comecei a escrever. Então, de novo, se formos mexer com várias propriedades de botão, podemos ter várias opções aqui embaixo aparecendo depois.

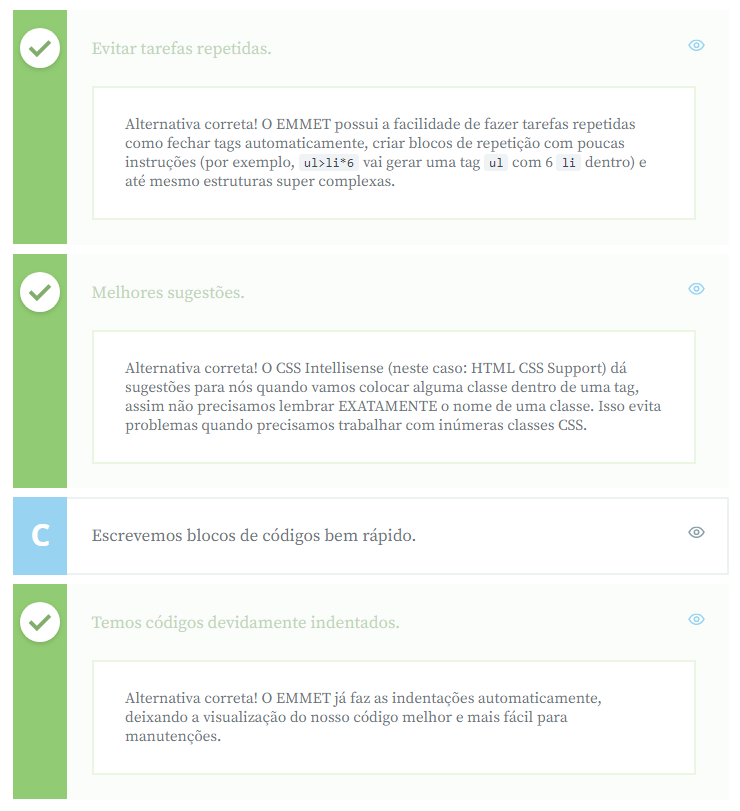
[06:50] Vou dar mais um exemplo, eu vou criar no CSS do botões uma classe só boba, para podermos entender depois, “.botao\_\_texto”. Ele não vai fazer nada por enquanto, é só para termos essa classe existindo. Vou voltar para o html, apagar isso aqui, “button.” e olha só, “botao”, “botao\_\_texto”, já aparece a sugestão automaticamente para nós.

[07:23] Agora que nós temos todas as ferramentas disponíveis para nós, para facilitar nosso trabalho, agora podemos começar a desenvolver o restante da página.

# 03 Melhorando o desenvolvimento

Durante o curso nós vamos utilizar dois plugins com funções diferentes.

Quais as vantagens de usar essas ferramentas?



# 04Variáveis CSS

[00:00] Se você copiou as coisas que eu fiz na aula passada, deve ter esse button inacabado, eu vou primeiro limpar essas partes aqui. Também a classe botoes.css, eu vou limpar essa classe também. Salvei os arquivos, beleza. Agora temos os arquivos mais ou menos como estavam antes.

[00:21] Seguinte, começamos a desenvolver aqui o botão e nós colocamos as cores, e vou lembrar, onde estão essas cores mesmo? Esses valores de cores? Estão no Figma, então eu preciso voltar no Figma toda vez que eu quiser saber qual é o nome de uma cor.

[00:38] E outra coisa também, é meio complicado ficar lembrando os hexadecimais, por exemplo, o amarelo é FFCB47, e esse preto aqui não é preto, ele tem uma cor diferente, ele é 202020. Aí tem o cinza, vamos ver o que é esse cinza, 909090. Ter que ficar lembrando os hexadecimais, qual era o preto, qual era cinza? Será que é 9090? 2020? Já começa a ficar muito complicado.

[01:15] Dá para resolver isso indo e voltando aqui para o Figma, só que é meio lento isso, temos que ficar saindo do nosso ambiente de trabalho e depois voltando para o Figma para poder ver a cor, aí volta de novo para o Visual Studio Code, volta de novo para o Figma, e ficar lembrando os hexadecimais é bem chato.

[01:36] Seria interessante se pudéssemos usar valores, por exemplo, amarelo, preto, cinza, branco são nomes mais fáceis de lembrar do que FFCB47 ou, qual era o cinza? Eu nem lembro mais qual era a cor do cinza mais. E como vamos criar esses nomes para os valores de cores que queremos?

[01:57] Para isso vamos usar uma ferramenta de CSS que são as variáveis CSS. Como nós usamos? Primeiro vamos criar um arquivo que vai conter todas essas variáveis que nós vamos criar. Vou abrir a barra lateral. Dentro da pasta “css” nós vamos criar outra pasta que vai ser a pasta de componentes básicos ou de estilizações básicas. Chamei aqui “base” e dentro desse base vou criar um novo arquivo chamado “variaveis.css”. Ok, fechar essa barra lateral.

[02:28] E agora aqui dentro podemos começar a colocar nossas variáveis, só que essas variáveis têm que estar dentro de um seletor, e queremos que o seletor seja universal, que todo mundo tenha acesso às variáveis. Eu poderia usar o seletor de html, mas podemos usar outro, que também é do html, que tem um peso mais forte para o seletor, que é o “:root”.

[02:53] Ele age da mesma maneira que colocar um seletor de html, só que os valores que colocamos dentro desse seletor têm um peso mais forte do que um seletor de tag. Agora aqui dentro podemos começar a colocar nossas variáveis.

[03:07] Por exemplo, aquela cor amarela maluca, podemos colocar “--amarelo”, para dizer que nossa variável é o nome amarelo, dois pontos, agora colocamos o valor, que é aquele “#FFCB47”, é o amarelo.

[03:31] Agora o branco, “--branco: #FFFFFF”;”. O “--cinza: #”, o que era? “909090”. E agora o preto, que não é um preto de verdade, é um preto mais claro, “#202020”. Pronto. Salvei o arquivo variáveis. Lindo, nós temos nossas variáveis aqui. E agora podemos fazer os ajustes no botão.

[04:07] Vou voltar aqui em botões, cadê? Background color, vamos usar agora a cor amarela. Para usar variáveis eu escrevo “var”, abro os parênteses e dentro agora eu coloco “--amarelo”. E agora no color, que é esse FFF, vamos apagar isso, escrever “var(--branco);”.

[04:35] Salvei o CSS e agora nós temos que importar esse variáveis lá no nosso index.html. Se eu não importar vamos ver como fica no navegador. Eu vou recarregar aqui a página, olha só, ele perdeu as cores, porque não existem essas variáveis ainda importadas no nosso html. Vou voltar para o index, agora vamos importar aqui.

[05:00] Uma dica: isso é para boas práticas. Quando formos importar arquivos de CSS é interessante deixarmos numa ordem específica. Eu vou importar aqui antes do botões, já que é um CSS bem básico, que deveria ser importado no começo de toda a página, vou colocar aqui “link”, “href=’assets/css/base/variaveis’”.

[05:28] Como esse CSS de variáveis que criamos deveria ser importado antes, faz sentido ser antes porque ele contém todas as informações básicas para que outros componentes possam usar, é interessante deixá-lo antes, logo no começo do desenvolvimento da página. Salvei aqui o import, vamos voltar para o navegador, recarregar, as coisas voltaram a funcionar.

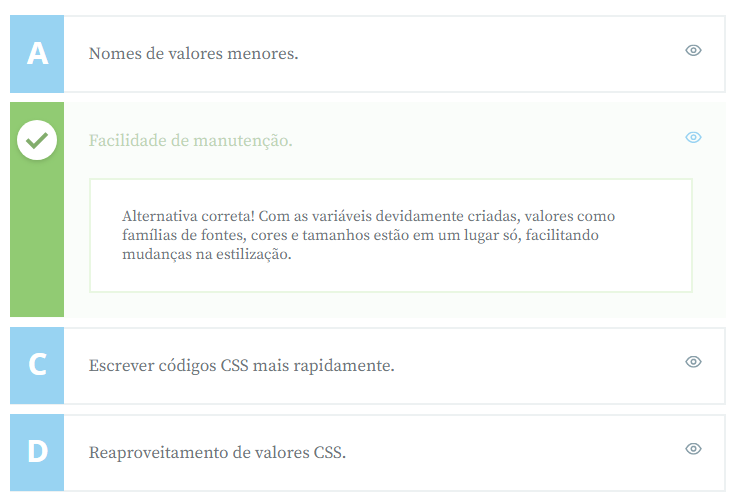
[05:50] “Só que, Matheus, eu preciso importar isso aqui nessa ordem?”, não precisa. Vou mostrar aqui. Dei “Ctrl + X” e importei agora abaixo do botoes.css. Então em teoria ele carrega primeiro o botoes.css e depois ele carrega as variáveis. Vou salvar aqui o html, voltei para o navegador, recarreguei, ele continua funcionando.

[06:13] A ordem do import dessas variáveis não importa muito, só que por questão de boas práticas nós costumamos deixar esses imports como, entre aspas, pré-requisitos, antes de qualquer outra coisa. Então voltar aqui, “Ctrl + Z”, deixar agora a variáveis aqui no topo, o import antes do botões. Então nós temos nossas cores e nós já fizemos nossos ajustes do botões, vamos para o próximo passo.

# 05Valores complexos

Trabalhar com variáveis CSS traz vantagens para nós, pessoas desenvolvedoras, como não precisar decorar valores complexos de cores.

Qual outra vantagem de usar variáveis CSS?



# 06Preparando o ambiente

Para o acompanhar o próximo vídeo, você pode baixar o arquivo reset.css.

O link para download do arquivo [reset.css](https://caelum-online-public.s3.amazonaws.com/1808-guia-de-estilos/02/Aula2-reset.css).

# 07Mais ferramentas CSS

[00:00] Aproveitando o embalo das variáveis, também podemos montar toda a estilização padrão que vai ser usada em todas as páginas, que, por exemplo, é o fundo com aquela textura e as linhas.

[00:09] Vamos criar dentro da pasta “css”, “base”, um novo arquivo chamado de “base.css”. Já peço perdão pelo nome do arquivo horrível, porque também não tenho muita criatividade com isso aqui.

[00:21] Por que existe esse base.css? A ideia é que nesse arquivo fiquem todas as informações que todas as páginas precisam para poder funcionar, a estilização básica de toda página. Então quer dizer que dentro desse index.html nós vamos ter o variaveis.css, o base.css e talvez outros imports que talvez tenhamos que colocar para poder ter a estilização padrão da página.

[00:48] Só que quando colocamos muito import básico no começo da página, fica meio difícil, quem vai mexer com essa página depois, saber quando acaba o base, quando começa o componentes, pode ficar meio confuso às vezes. A ideia é que a pessoa que for desenvolver só precise importar um arquivo e ela já tenha toda a estilização necessária para ter a página. E é por isso que existe esse base.css.

[01:22] Como vamos fazer então para mesclar esse base.css com o variaveis.css? O CSS tem uma função chamada import, então nós escrevemos “@import url()” e agora passamos o caminho dessa variável, esse variaveis.css. Como está no mesmo nível é só digitar “variaveis.css”. Só tomar cuidado que no final da linha temos que colocar o ponto e vírgula, porque o autocomplete não coloca para nós.

[01:50] Aproveitar aqui também e já colocar as imagens de fundo, então “body {“, “background: url(../../img/fundo-textura.png), url(../../img/fundo-linhas.png);”. E agora só centralizar o “background-position: center;”. Salvei esse arquivo, base.css, e agora vou no index.html e vou trocar o import de “variaveis.css” para “base.css”.

[02:24] Salvei o arquivo, vou no navegador, recarreguei a página e, olha só, ele não apagou as cores do botão, não apagou as cores da fonte também e ele também colocou o background, aquela textura de fundo e as linhas.

[02:43] Só uma coisa, antes de continuar eu só quero fazer um comentário, que são boas práticas aqui do projeto. Quando temos um import desse tipo, por exemplo, variaveis.css, esse arquivo não é diretamente importado no documento html, ele passa pelo base.css antes. Nesses casos é interessante que troquemos o nome desse arquivo variaveis.css para \_variaveis.css.

[03:11] Eu vou no editor de texto e já vou trocar o nome “variaveis.css” para “\_variaveis.css”. E vou trocar também aqui no base.css. Troquei o import url, coloquei o underline antes. Salvei o arquivo base, voltei para o navegador para ver se nada foi explodido no meio do caminho, nada explodiu. Agora podemos continuar aqui escrevendo nosso código.

[03:39] A pergunta agora é: quem colocou esse espaçamento entre um botão e outro? Quem colocou esse espaçamento entre o botão e a lateral? Verdade, o navegador. O navegador já tem uma estilização padrão que ele carrega, só que nós não queremos usar essa estilização do navegador, queremos usar a nossa estilização. Para limpar essa estilização do navegador vamos usar uma ferramenta que é o reset.

[04:01] O reset vai limpar toda a estilização padrão do navegador e vai permitir que usemos a nossa. Eu vou usar um CSS reset customizado que vocês podem baixar no exercício anterior, que é o exercício de preparando o ambiente. Eu vou copiar esse arquivo que eu baixei e colocar na nossa pasta base, “Ctrl + V”, e aproveitar, já que ele vai ser importado indiretamente no html, trocar o nome para “\_reset.css”.

[04:31] Vou voltar para o Visual Studio Code, dentro do base agora vamos importar, @import: url(\_reset.css);, lembrando que o autocomplete não coloca o ponto e vírgula no final, nós temos que colocar. Salvei o arquivo, voltei para o navegador, recarreguei, ele já removeu toda a estilização padrão que vem do navegador.

[04:55] Uma coisa que podemos fazer já é colocar o nosso espaçamento. Vou voltar para o editor de texto e vou criar uma classe “.container” dentro de base.css, que vai ter um “padding” lateral, à direita, de “1rem;” e um “padding” à esquerda de “1rem;” também. E agora dentro do html nós vamos usar esse container. Eu vou criar uma nova tag chamada “main” e usando o Emmet eu quero colocar nossa classe container.

[05:27] Olha só que legal, só de apertar o ponto ele já nos mostra a classe que acabamos de criar, que é a classe container, então eu seleciono o container, “Ctrl + espaço”, “Enter”, já criei para nós nossa tag main e a nossa classe container, recortar isso aqui, os nossos três botões e colocar dentro do main, só identar, beleza. Salvei o index.html, voltei para o navegador, recarreguei a página e nosso container está lá e já coloquei o espaçamento nos botões.

[06:05] O que podemos colocar também são as fontes que vão ser utilizadas na página inteira. Lembrando que a Montserrat não vai ser usada só no botão, ela vai ser usada na página inteira, então é um controle que pode estar dentro do body, e caso queiramos trocar de todos de uma vez só, podemos deixar dentro do próprio variáveis. Então vamos criar mais uma variável, que vai ser, por exemplo, “--familia-corpo:”, que vai ser a “’Montserrat’, sans-serif”.

[06:42] E agora podemos também colocar o tamanho dessa fonte, então “--tamanho-corpo:”, que vai ser de “1rem;”. Salvei o variaveis.css e agora aqui no base, olha só que legal, basta pegarmos aqui “font-family: var(--família-corpo);” e embaixo do font-family colocamos o tamanho, que é “font-size: var(--tamanho-corpo);”.

[07:18] E no botão, ao invés de colocar esse font-family Monserrat de novo, podemos falar para o botão herdar a família da fonte do próprio body. Então podemos arrancar isso daqui, o Monteserrat, e simplesmente escrever “inherit;”. Salvei o botão, salvei o base, salvei o variáveis, voltei para o navegador, recarreguei.

[07:41] Nada mudou, isso porque o botão está herdando do próprio body. E se trocarmos aqui, do família do corpo, de “Montserrat” para “Arial”, por exemplo, salvei, voltei para o navegador, o próprio botão, como ele está herdando, já troca automaticamente a família da fonte. Deixa eu voltar para o Montserrat, voltar para o base. Legal, contexto pronto, botão arrumado, base arrumada, nós podemos seguir para o próximo passo.

# 08Faça como eu fiz

Praticar ajuda bastante no aprendizado de um novo conceito. Assim, é muito importante que você implemente o que foi apresentado nesta aula.

Não deixe de sanar suas dúvidas antes de dar continuidade ao curso. Estaremos te esperando no fórum da Alura caso alguma dúvida surja.

# 09O que aprendemos?

Nesta aula, aprendemos:

* Como baixar e instalar plugins no Visual Studio Code
* O uso de variáveis
* Padronização
* Import CSS
* Criar uma folha de estilos única que contém todas as informações de estilos mínimas para uma página

# 01Projeto da aula anterior

Caso queira começar daqui, você pode baixar o projeto da aula anterior nesse [link](https://github.com/alura-cursos/styleguide-css/archive/Aula2.zip).

# 02O input

[00:00] Pensando em botão ainda, voltando para o Figma, na segunda parte nós temos dois botões, um em sequência do outro, um de comprar agora e um de adicionar ao carrinho.

[00:14] Se observarmos bem, os dois são bem parecidos com relação ao tamanho e tipografia, por exemplo, só que um tem borda, um não tem cor de fundo, o fundo é transparente, não tem sombra e a cor é amarelada. Enquanto o outro foi o que nós criamos antes.

[00:32] Para poder fazer esse botão, ao invés de criarmos uma classe e estilizar esse botão do zero, podemos simplesmente aproveitar as propriedades que nós temos aqui do botão de comprar agora, que nós já montamos antes, e só colocar no CSS as variações deste botão, que são a cor do fundo, a borda, a falta de sombra e a cor do texto.

[00:53] Vou para o código, dentro do CSS de botão. Vou criar mais uma classe chamada “.botao--alternativo {“, e dentro eu vou colocar agora as propriedades que são diferentes do botão que nós criamos aqui em cima.

[01:11] A cor do texto vai ser diferente, não é mais branco, vai ser amarelado, então “var(--amarelo);”. A cor de fundo agora é transparente, “background-color: unset;”. Não tem mais sombra, então “box-shadow: none;” e agora temos uma borda, então é “border: 1px solid”, e agora aquela cor amarelada, “var(--amarelo);”.

[01:45] Salvei o botoes.css, voltei para o index.html e agora o lugar desse botão, eu vou trocar esse p que nós colocamos, vou colocar um botão de verdade. Vou escrever “button.botao.botao--alternativo”.

[02:04] Salvei o html, voltei para o navegador, recarreguei a página e, legal, faltou colocar só um texto aqui, “Botão Alternativo”, só para termos uma ideia do que tem ali dentro. Pronto. Agora apareceu o texto.

[02:23] O tamanho é o mesmo, a tipografia é a mesma, só o que muda mesmo foram aquelas quatro mudanças, tudo o que não eram aquelas quatro mudanças manteve do botão que nós colocamos acima. Então tamanho do botão, o espaçamento do texto, tudo isso se manteve.

[02:39] Tanto que se arrancarmos a classe botão desse conjunto, arranquei aqui do html, salvei, voltei para o navegador, ele perdeu o espaçamento, perdeu o tamanho, perdeu a tipografia e só manteve aquelas quatro mudanças que colocamos antes. Precisamos ter a classe botão junto.

[03:01] Outra coisa, o botao--alternativo, essa classe que eu criei, esse nome vem de uma convenção chamada BEM. E eu vou deixar um exercício para você, para saber mais, onde eu vou deixar dois links, um da própria documentação do BEM e outra é um post no Medium falando sobre o BEM, é um post bem rápido, bem simples sobre o BEM.

[03:31] Agora acabamos o botão, acabamos qualquer coisa que tinha a ver com o botão, vamos para o próximo componente que vai ser esse input. Podemos começar montando a estilização básica desse input.

[03:41] Como ele vai ser um novo componente, não podemos esquecer de criar um novo arquivo, então dentro da pasta “componentes”, “input.css”, criei a classe “.input {“, e agora vamos colocar a estilização do input aqui.

[03:56] A cor, por exemplo, podemos herdar do corpo da página, o font-family também e o tamanho da fonte. Todas essas propriedades são interessantes de herdar do corpo da página. Agora colocamos uma largura “100%”, “padding-top: 1.75rem;” e todos os lados vão ter o mesmo espaçamento, então “right”, “bottom” e “left”.

[04:40] E toda vez que colocarmos uma propriedade de tamanho, por exemplo, e depois colocarmos padding, é sempre bom colocar um “box-sizing: border-box;” para poder garantir que esse tamanho vai ser o absoluto e não esse tamanho mais o tamanho dos paddings.

[05:02] Colocar um “margin”, por exemplo, de “1rem;”, o espaçamento para baixo já. Agora temos que arrancar a borda que vem padrão do input. A cor de fundo, por mais que ela já seja branca por padrão, é interessante deixar isso escrito, garantindo que ele vai ser sempre background branco. E para finalizar, só colocar o sombreado aqui, “box-shadow: 10px 10px 30px”, agora o preto, “#000000”, e agora a transparência “0F;”.

[05:43] Salvei o arquivo input.css, importar no html agora , então link “assets/css/componentes/input.css”. E entre um botão e outro eu vou colocar aqui “input.”, adicionar a classe “input” nele. Input está text, ok. Salvei o html. Voltei para o navegador, recarreguei a página, nosso input está aqui funcionando, bonitão. Agora vamos adicionar o ícone.

[06:15] Só voltando para o Figma antes, que outro input será que usa esse ícone também? Não esse ícone em específico, mas tem um ícone. Aqui tem um formulário, aparentemente ícone mesmo vai ser só esse input aqui, mas nada garante que não tenhamos outros inputs com ícones também em outras partes da página, só existe aqui por enquanto.

[06:41] Então podemos criar uma variação desse input só com esse espaçamento para deixar o ícone. Então voltando aqui para o código, input, vamos criar uma nova classe que vai ser essa variação, “.input--icone {“ e colocamos já o espaçamento, “padding-left: 4.25rem;”.

[07:04] E agora, vou voltar para o Figma, esse ícone é só um complemento do texto que está escrito aqui do lado, então podemos deixar esse ícone no CSS com o background image.

[07:18] Então dentro da classe input--icone podemos deixar algumas propriedades de background, como, por exemplo, “background-repeat” já aqui, para não repetir, e o posicionamento do background, que queremos que o ícone sempre fique daquele lado, então “background-position: left”, que o left vai ser exatamente o espaçamento que tínhamos colocado antes, de “1.75rem”, e queremos centralizar verticalmente o ícone, então “center;”.

[07:48] E agora podemos finalmente adicionar o ícone, que vai ser outra classe específica para conter esse ícone de e-mail. Então vai ser um “input--email {“, vai ter um “background-image: url(../../img/icones/mail.svg);”.

[08:10] Salvei o input, voltei para o html e vou adicionar na classe o “input input--icone input--email”. Adicionei essas duas classes em conjunto com a classe input que existia antes, salvei o html, voltei para o navegador, pronto, está aqui o ícone.

[08:29] Agora falta colocar esse texto do meio, que é o que chamamos de placeholder, ele é um atributo da tag input, então vou colocar aqui “placeholder=’Insira seu e-mail’”, salvei o html. Agora nós temos aqui uma dica para a pessoa que for mexer com o site, ela sabe agora o que é esse input, que informação ela tem que digitar aqui dentro.

[09:01] Isso também serve para acessibilidade, o leitor de tela, por exemplo, vai ler o código aqui e vai saber que isso é um campo para poder inserir o e-mail.

[09:11] Só que tem um detalhe, se pegarmos esse input e digitarmos alguma coisa por cima, esse texto alternativo se perde, o placeholder não existe mais, e leitores de tela não conseguem mais retornar aquele valor que estava antes, que era o de insira seu e-mail, então meio que perde a acessibilidade desse input.

[09:35] Para evitar esse tipo de coisa, para que sempre exista uma descrição do para que serve esse input, vamos usar outro atributo chamado aria-label.

[09:45] Então voltando para o código, depois do atributo de placeholder, lançar mais um atributo, “aria-label”, e dentro dele nós vamos colocar o mesmo texto que colocamos dentro do placeholder, que é “Insira seu e-mail”. Salvei o html, voltei no navegador, recarreguei, temos aqui o texto.

[10:06] Fácil, insira seu e-mail, ok, vou colocar meu e-mail aqui, fiz alguma outra coisa no site, o que era isso aqui mesmo? Tem o ícone aqui do lado, então deve ser de e-mail, mas agora com o leitor de tela ele vai ler aqui, insira seu e-mail, a pessoa vai lá e insere o e-mail, passou, tirou, fez alguma coisa, mas o que era esse campo aqui mesmo? O leitor de tela vai falar: “Era o campo de inserir e-mail”, a pessoa fala: “Então beleza, aqui era para ser o e-mail, não era para ser isso que eu escrevi”, então “email@email.com”, por exemplo.

[10:42] Com isso encerramos agora esse componente de input e podemos fazer o próximo componente aqui mais para baixo.

# 03Para saber mais: Padrão BEM CSS

Caso queira saber mais sobre o padrão de CSS **BEM**:

* [Post no Medium](https://medium.com/trainingcenter/bem-em-5min-f5c80fd23439)
* [Documentação oficial do BEM](http://getbem.com/)

# 04O cartão

[00:00] Vamos para o nosso próximo componente, que vai ser a próxima seção. Deixa eu dar um zoom, temos aqui o cartão branco com sombreado, temos um título composto aqui, que é “Como conseguir minha planta”, então ele tem um destaque, outra tipografia, e logo abaixo temos três itens que são os passos para conseguir a planta. Beleza, esse aqui é o componente.

[00:31] Mas antes de bater o martelo com a decisão de que isso aqui tudo é um componente, vamos ver onde mais nós usamos essa composição. Então vou tirar o zoom aqui, vamos ver. Mais para baixo nós temos produtos, vamos ver.

[00:46] Olha nosso título composto aqui, só que ele está fora do cartão aqui embaixo, então beleza, ele nao vai ser usado aqui. Vamos ver aqui. Resumo do pedido, temos o cartão aqui em volta, só que agora não temos o título composto. Vamos ver onde mais.

[01:11] Aqui embaixo, no veja nossos vídeos, é um título composto de novo, só que ele não está dentro do cartão. Então deu para ver que cada um desses conjuntos são componentes separados, então o próprio cartão, só esse fundo branco já é um componente sozinho e o título composto vai ser outro componente e cada um desses itens, “Escolha suas plantas”, “Faca seu pedido”, vai ser outro componente.

[01:44] Toda vez que formos montar um componente, decidir que é um componente, temos que tomar cuidado com isso. Onde mais é utilizado? Onde mais essa estrutura se repete? Do que simplesmente chegar lá e começar a montar e, no final das contas, aquele componente nem poderia ser reutilizado em si, porque a composição inteira não é um componente.

[02:01] Nós íamos criar um componente que só ia ser usado basicamente uma vez ao invés de criar componentes, como, por exemplo, o título, que podem ser reutilizados em vários outros lugares.

[02:13] Vamos criar então o nosso componente de cartão, só o cartão do fundo por enquanto. Vou voltar para o Visual Studio Code, já que é um componente novo. De novo vou no arquivo de CSS, pasta “componentes”, vou chamar de “cartao.css”.

[02:29] Dentro de cartão vamos criar o “.cartao {“, e o que esse cartão tem? Novamente, o que eu sempre recomendo colocarmos quando vamos mexer com espaçamentos, bordas, paddings, é colocar “box-sizing: border-box;”, a primeira coisa. Vamos mexer com padding, mexer com espaçamento, border, já coloca um box-sizing.

[02:48] Porque agora o próximo passo vai ser colocar os paddings, então “padding-top: 2rem;”, e agora o “padding-bottom: 2rem;”. Vou colocar uma cor de fundo, “background-color: var(--branco);”. Colocar um sombreado agora, então “box-shadow: 10px 10px 30px” e agora a cor preta, “#000000”, e agora a transparência.

[03:25] Então fiz o nosso cartão, vou voltar para o html, salvei o cartao.css, vou importar agora no html, link “assets/css/componentes/cartao.css”. Salvei. Vamos pegar, por exemplo, dentro do main, logo abaixo de todo mundo, vamos criar um “article” que vai ser um “cartao”, só isso. É um article com cartão vazio. Abrir agora no navegador, vamos ver o que acontece. Está aqui o cartão.

[04:03] Se colocarmos alguma coisa aqui dentro, por exemplo, um botão, “button.botao”, vou colocar aqui “Um botão”. Salvei o html, olha só, ele está aqui, estrutura, bonitinho, perfeito. Está com espaçamento em cima, embaixo e o efeito de sombra.

[04:33] Só voltando rapidinho para o Visual Studio Code, dentro do cartão usamos essa sombra e usamos essa sombra em mais um lugar, que foi no input. Dentro do input nós usamos o mesmo box-shadow, ou melhor, a mesma cor. Podemos extrair essas cores de sombras também no nosso variaveis.css. Então aquele amarelo transparente e o preto transparente, que foram usados em sombras, podemos deixar aqui.

[05:04] Por exemplo, embaixo de amarelo, no arquivo de variaveis.css, podemos colocar “--amarelo-transparente:”, que vai ser, cadê? Botões, vai ser esse cerquilha aqui, FFCB não sei o que, não dá para decorar esse tipo de coisa. E agora o preto, “--preto-transparente:”, que vai ser – esse dá para decorar porque é bem simples – “#0000000F;”.

[05:45] E agora fazer as mudanças que precisamos fazer nos botões, então ao invés de ser isso aqui vai ser “var(--amarelo-transparente);”. Atualizamos o botões aqui. No base não usamos nada de sombra, o input tem sombra, então aqui no box-shadow vai ser “var(--preto-transparente);”, salvei. E agora no cartão também, aqui vai ser “var(--preto-transparente);”.

[06:24] Salvei tudo, vou voltar para o navegador e nada deve mudar. Vamos ver, recarreguei e nada mudou. Continua com a sombra aqui, dá para ver um pouco aqui da lateral, que a sombra amarelada está aqui ainda, a sombra do input, a sombra do cartão que nós criamos e o botão também está aqui, o sombreado dentro do cartão.

[06:47] Toda vez que a gente for usar agora uma dessas cores transparentes, você precisa colocar, por exemplo, amarelo transparente ou preto transparente. Então agora nós encerramos os componentes, vamos para os próximos.

# 05O título

[00:00] O próximo componente aqui vai ser nosso título composto, vou voltar aqui para o Figma. Nós falamos que esse título, o conjunto de como conseguir minha planta vai ser um componente só, porque ele vai ser usado em vários lugares da página, não só nessa página como também em outras páginas. Vamos ver.

[00:20] Temos aqui o título composto, logo no começo, na seção de chamada. Nós temos um aqui dentro do cartão, logo antes dos produtos, logo antes da seção de vídeos, usamos em vários lugares. E a diferença entre um componente e outro vai ser só o alinhamento de texto praticamente.

[00:43] Como vamos fazer essa estrutura? “Como conseguir minha planta” é um título só, a diferença é que “minha planta” que está com destaque, então podemos montar todo esse texto dentro de um h2, por exemplo, e nós separamos o “minha planta” com um span e damos o destaque no span.

[01:06] Agora que já sabemos como vai ser a estrutura, vamos pensar na estilização. Esse título vai ter um peso diferente, essa parte com destaque vai ter um peso diferente, o título total vai ter um tamanho diferente, inclusive o próprio destaque vai ter um tamanho diferente também.

[01:25] Podemos criar essas variáveis já de tamanho, font-family e pesos, então vamos para o nosso Visual Studio Code. Nós citamos que ia ser um componente novo, então vamos criar um arquivo novo.

[01:43] Tipografia não é uma coisa que muda na página, a ideia é que a tipografia seja a mesma e seja constante durante todo o projeto, o que menos vai sofrer mudanças nesse projeto inteiro vai ser a tipografia. Se alguma coisa mudar, vai mudar de todo o projeto.

[02:02] Como é uma coisa que vai ser importada em todas as páginas, porque a tipografia vai ser igual em todas as páginas, vou voltar aqui no Figma, a tipografia não muda, é a mesma coisa em todas as páginas. Então podemos deixar a tipografia dentro do base. Vou voltar para o Visual Studio Code, dentro de base vamos poder escrever o nosso componente título, por exemplo, “.titulo {“. Vamos fazer então as variáveis.

[02:34] Abri o arquivo de variáveis, vamos começar adicionando a família de fonte nova, então “--familia-titulo-destaque:”, porque, lembre-se, é o destaque que tem a família diferente, porque o restante da página inteira vai usar o Montserrat. É aquela “’Elsie Swash Caps’,”, caso não exista, ela vai ser uma fonte serifada. Próximo vai ser agora colocar o peso do título em destaque, que vai ser de “900;”.

[03:22] Aproveitando que colocamos já o peso do título, vamos colocar também o peso do botão, “--peso-botao: 700;”. E agora colocar os tamanhos. Temos o tamanho do corpo, nós temos o tamanho do título, que vai ser “1.325rem;”, e o tamanho do título com destaque agora, que vai ser de “2rem;”. Perfeito. Desculpa, aqui não é 325, é 375.

[04:00] Agora que fizemos as variáveis, vamos arrumar o botão. Salvei o variáveis, pegar aqui o botão, onde tem o “font-weight: 700;” trocar para “var(--peso-botao);”. Salvei o botão, agora em base podemos fazer nosso título.

[04:16] Nosso título vai ter a cor, que vai ser esse “color: var(--cinza)”. O título vai ter essa cor cinza, só que antes de colocar a cor cinza no título, se vermos bem, todo o corpo da página tem essa fonte cinza. Toda fonte aqui tem essa cor cinza.

[04:48] Na verdade, não é o título que tem a cor cinza, ele tem, mas não é específico do título, o corpo inteiro da página tem. Então ao invés de adicionar isso no título, vamos adicionar dentro do body, logo abaixo de font-size, colocamos “color: var(--cinza);”. E no título vai ficar só o tamanho mesmo, então “font-size: var(--tamanho-titulo);”. Perfeito.

[05:22] Depois do título temos agora o “.titulo--destaque {“ e o que difere o título do título destaque? A font-family, então “font-family: var(--familia-titulo-destaque);”. Próximo vai ser a cor, que vai ser diferente, não vai ser aquele cinza, então “color: var(--preto);”. Próximo vai ser o tamanho da fonte, então “font-size: var(--tamanho-titulo-destaque);”. Agora o peso da fonte, que não vai ser mais 700 nem 400, vai ser 900. “font-weight: var(--peso-titulo-destaque);”.

[06:25] Só que agora temos que fazer esse título pular linha, porque eles não vão ficar tudo numa linha só, eles têm que quebrar a linha, então podemos colocar, por exemplo, um “display: block;” e forçamos a quebra de linha.

[06:37] E coloca o espaçamento entre a parte sem destaque e a parte com destaque, faz uma separação entre eles, e aproveitar também, já fazer o espaçamento para o elemento abaixo, vou voltar aqui para o Figma, que pode ser essa lista de itens ou pode ser a lista de produtos. Então é legal já colocar esse espaçamento para baixo também. “margin-top: .75rem;” e o “margin-bottom: .75rem;”.

[07:12] Agora só finalizar aqui com os alinhamentos, então “.alinhamento--esquerda {“, “text-align: left;” e agora um alinhamento centralizado, então “.alinhamento--meio {“, “text-align: center;”.

[07:38] Já que estamos fazendo as coisas de tipografia, podemos também colocar, por exemplo, uma classe chamada “.texto {“ e essa classe texto vai mexer só com alinhamento, a distância entre uma linha e outra, que é o line height.

[07:53] Já que estamos mexendo com tudo que vai trabalhar com corpo, então corpo de texto, assim como, por exemplo, aqui na chamada – vamos ver onde mais é usado – aqui no rodapé, onde tem as ruas, podemos deixar a distância entre linhas aqui. Voltar pelo Visual Studio Code, “line-height: 26px;”.

[08:22] E agora vamos testar isso tudo que fizemos. Dentro do index.html eu vou apagar esse botão que nós criamos, que não faz sentido, porque ele não existe naquele cartão, e podemos colocar o título, então ele vai ser um “h2.titulo>“ e dentro desse título vamos ter o título destaque, então vai ter um “span”, que vai ser um “.titulo--destaque”.

[08:48] E esse título vai ter o alinhamento centralizado ou na esquerda? No meio, então podemos, junto do título, colocar “alinhamento--meio”. Rodando esse código todo, legal. Antes do span colocar, qual era o texto? Voltar aqui, cadê? “Como conseguir minha planta”. Então “Como conseguir”, e agora dentro do span, deixa eu quebrar a linha só para podermos ver melhor, e agora dentro de span, “minha planta”.

[09:31] Salvei o html, voltei para o navegador, recarreguei e perfeito, “Como conseguir minha planta”. Se inspecionarmos os elementos, o h2 está com o título, com o tamanho do título e o span está aqui também, com a variação de tamanho, 2rem, 900, perfeito. E todo mundo, inclusive o input, está com o texto cinza agora, como deveria ser antes. Legal, vamos para o próximo componente.

# 06Para saber mais: Componentes

Durante o processo de desenvolvimento de uma página e quando vamos componentizar algo, precisamos entender e diferenciar componentes.

Alguns dos passos para diferenciar componentes é isolar partes diferentes do que queremos componentizar e ver se são usadas em outros lugares. Se sim, quer dizer que dentro do componente existem outros possíveis componentes, e o que queremos extrair na verdade é apenas a "casca". Se não, então o conjunto inteiro forma um componente, tanto o conteúdo quanto o que contém essas informações.

# 07Variações de componentes

No curso nós montamos um botão com a seguinte estrutura e estilização:

HTML

<button class="botao">Assinar newsletter</button>COPIAR CÓDIGO

CSS

.botao {

background-color: var(--amarelo);

text-align: center;

font-family: inherit;

font-weight: var(--peso-botao);

color: var(--branco);

width: 100%;

padding-top: 1.75rem;

padding-bottom: 1.75rem;

box-shadow: 10px 10px 30px var(--amarelo-transparente);

border: none;

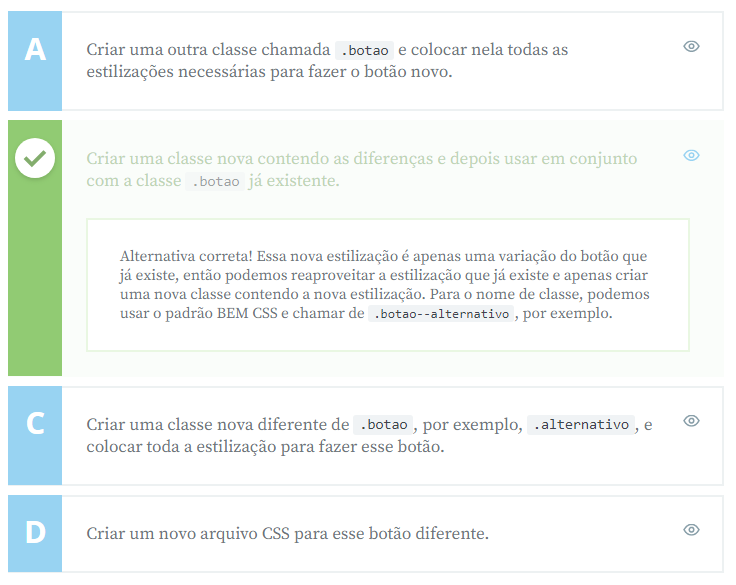
}COPIAR CÓDIGO



Depois tivemos que montar um outro botão com estilização diferente.



Qual seria a melhor maneira de fazer esse botão?



# 08 Faça como eu fiz

Praticar ajuda bastante no aprendizado de um novo conceito. Assim, é muito importante que você implemente o que foi apresentado nesta aula.

Não deixe de sanar suas dúvidas antes de dar continuidade ao curso. Estaremos te esperando no fórum da Alura caso alguma dúvida surja.

# 09 O que aprendemos?

Nesta aula, aprendemos:

* Como baixar e instalar plugins no Visual Studio Code
* O uso de variáveis
* Padronização
* Import CSS
* Criar uma folha de estilos única que contém todas as informações de estilos mínimas para uma página

# 01 Projeto da aula anterior

Caso queira começar daqui, você pode baixar o projeto da aula anterior nesse [link](https://github.com/alura-cursos/styleguide-css/archive/Aula3.zip).

# 02 O item

[00:00] Ainda dentro desse cartão, vamos lá no Figma, nós temos dentro do cartão mais três itens. Vamos pensar na estrutura agora. Nós temos aqui uma lista de itens, dentro de cada item da lista temos um span, que vai ser exatamente o ícone aqui e depois o texto, que pode ser um parágrafo, por exemplo.

[00:26] Pensando na estrutura básica, vai ser uma lista, dentro da lista vão ter vários itens, são três aqui no caso, e dentro de cada item vai ter um span com um p. Já sabemos mais ou menos qual é a estrutura, podemos pensar em trabalhar o CSS.

[00:44] Novamente, é um componente novo, ele vai ser um item, então vamos criar dentro de “componentes” um novo arquivo chamado “item.css”. Agora dentro de item vamos começar pela lista, de fora para dentro.

[00:59] Então “.lista-item {“ e dentro dessa lista vamos arrumar o “display:” dela, vamos colocar o “flex;” para poder deixar alinhados os elementos dentro da lista, para deixá-los centralizados, já que o alinhamento de texto não serve para isso, então para alinhar esses elementos no meio temos que usar o display flex. Então “flex-direction:”, não queremos que ele fique em linha, queremos que ele fique em coluna, e agora vamos alinhar os elementos no meio. Perfeito.

[01:46] Isso aqui já está pronto e agora vamos fazer cada um dos itens. “.item {”, cada item vai ter uma largura de “15rem;” e uma margem para baixo de cada um dos itens da lista de “1rem;”.

[02:25] Agora vamos fazer o círculo amarelo que abraça o ícone, então “.item\_\_icone {”, que esse vai ser o span, o span vai ter que ter um tipo de display, porque ele tem que existir, por mais que ele seja vazio, ele tem que ocupar um espaço, e como o span naturalmente tem um display inline, nós não conseguimos colocar medidas nele, então temos que usar o “display: block;”.

[02:51] Só uma coisa antes, já que temos que usar um display block e colocar uma altura e largura nesse ícone, também temos que deixar o alinhamento de cada um dos itens arrumado. Dentro de “.item {”, voltar rapidinho só, “display: flex;” e agora temos que centralizar esses itens.

[03:11] Vou voltar no Figma só para ter uma visualização melhor. O adjustment está vendo a bola aqui no meio? A bola amarela tem que ficar centralizada com o texto e isso nós vamos conseguir com o display flex. Então “align-items: center;”.

[03:27] Voltando para o ícone, display block. Temos uma margem à direita, se pudermos vamos poder deixar o espaço entre o ícone e o texto, “1rem;”. Colocar agora as dimensões, largura “3.25rem;”, altura também. Vou colocar agora a borda arredondada, “border-radius: 50%;”, vai formar um círculo perfeito.

[04:00] Agora a cor de fundo vai ser “var(--amarelo);”. E aproveitar também, colocar as propriedades de imagem de fundo, que é onde vão ficar os ícones, podemos deixar isso como padrão para todos os itens. Então “background-size:” vai ter sempre “1.375rem;”. Nenhum background vai repetir, “no-repeat;” e a posição dos ícones tem que ficar exatamente no meio, então “background-position: center;”.

[04:41] Por que nós não estilizamos o texto? O texto já tá estilizado por padrão por causa do corpo da página mesmo, então já que montamos toda a estrutura básica, podemos montar a estrutura no Emmet, no html.

[04:53] Voltando para cá, logo abaixo do h2 podemos começar a pensar na estrutura. Nós temos aquela lista e dentro da lista nós temos a classe “.lista-item>”, dentro da lista nós temos itens da lista, “li”, são três, então vezes três, uso o “\*3”, e agora vai ter o “.item>”.

[05:19] E dentro de cada item temos aquele span e o p, então “span.”, o span vai ser um ícone, então tem que ser “item\_\_icone”, e junto desse span tem o “+p”, que como não tem uma estilização única para esse p, é só um p normal. Podemos dar “Enter” e agora só preencher com os textos.

[05:43] Vamos ver, escolha suas plantas, pedidos e aguarde em casa. Então “Escolha suas plantas”. Aqui é o span, na verdade. “Faça seu pedido”, “Aguarde na sua casa”. Preenchemos com os textos.

[06:14] E agora falta só montar os ícones e cada ícone vai ser individual, um vai ser do cursor, um vai ser do carrinho e um vai ser do caminhão. Vou voltar para o item, logo abaixo, eles vão ser variações dos ícones, então para cada classe de item ícone vai ser um ícone diferente para cada um dos itens, então “.item\_\_icone--cursor {”, ele vai ter um “background-image: url(../../img/ícones/mouse.svg);”.

[06:58] Próximo, “.item\_\_icone”, vai ser cursor agora, vai ser o “--carrinho {“. “background-image: url(../../img/icones/cart.svg);”. Agora o último, que é o caminhão. “background-image: url(../../img/icones/truck.svg);”.

[07:35] Salvei o arquivo de item.css, voltei no html e agora eu vou preencher do lado de cada classe item icone, eu vou colocar “item\_\_icone”, o ícone de cada um dos itens. O primeiro vai ser o de cursor, o segundo vai ser ícone de carrinho e o último vai ser o item de caminhão. Salvei o html.

[08:01] Não vamos nos esquecer de importar o arquivo CSS, porque eu cometo esse erro até hoje, “assets/css/componentes/item.css”. Salvei o html, voltei para o navegador, recarreguei a página e, legal. Praticamente o cartão inteiro com o título e o conteúdo já estão prontos. Agora podemos seguir para o próximo componente.

# 03O produto

[00:00] Vamos para a próxima seção, vamos ver aqui no Figma, a próxima seção agora é de produtos. Tem um cartão aqui dentro que é bem parecido com o cartão que nós acabamos de terminar, vamos ver a diferença entre eles.

[00:13] O cartão de cima, todos os elementos aqui estão organizados na vertical, então vai ser sempre um componente dentro do outro. Esse cartão pode acomodar praticamente qualquer tipo de informação. Vamos ver esse cartão aqui de baixo, de produto.

[00:29] Os três primeiros são divididos na horizontal, ou seja, imagem e conteúdo, e dentro do conteúdo tem o título do produto, o preço e o comprar. Então existe uma grande diferença entre esses dois cartões é que o cartão de cima recebe qualquer tipo de informação, qualquer coisa que quisermos colocar aqui dentro podemos colocar, e esse cartão de baixo é um cartão muito específico para quando vamos usar produto, então são componentes diferentes.

[01:00] Vamos pensar na estrutura desse componente do produto. Temos um cartão que pode ser um article, como nós fizemos em cima. dentro nós temos uma imagem e uma divisão onde vai conter o conteúdo, para poder fazer metade, metade. Então imagem, uma div, dentro da div vamos ter um título, um preço e um botão de compra com ícone dentro. Podemos montar as classes já no css.

[01:34] Dentro da pasta “componentes”, criar um arquivo novo chamado de “produto.css” e agora começar a estilizar o produto, que vai ser o nosso cartão. “margin-bottom: 1rem;”, os espaçamentos, “2rem;”. E como vamos dividir o cartão no meio, o espaçamento entre um e outro tem que ser também 50%, vou colocar aqui “padding-left: 50%;”. E vamos deixar a imagem flutuando dentro desse cartão.

[02:10] E para que a imagem flutue relativa ao cartão do produto, nós colocamos o “position” dentro do cartão de “relative”. E para não deixar a imagem vazar, nós deixamos “overflow: hidden;” e a imagem agora não vaza do cartão, ela fica contida dentro do cartão ainda.

[02:29] “background-color” vai ser branco, vou colocar um sombreado aqui agora, “box-shadow: 10px 10px 30px”, o preto transparente. Toda vez que mexemos com padding é legal deixar um “box-sizing” aqui de “border-box”, e eu vou deixar uma altura definida aqui de “12.5rem;”. Inclusive essa altura vai ser mínima, para garantir que pelo menos o cartão tenha esse tamanho.

[03:14] Dentro do html vamos lá criar um “article.produto”. Salvei o html, vou para o navegador, recarreguei a página e faltou importar. Subi aqui, “link”, “assets/css/componentes/produto.css”. Salvei o html, voltei para o navegador, carreguei, agora aqui está o cartão.

[03:44] Por causa do mean radicals que nós colocamos antes, o cartão vai ter um tamanho pré-definido, além também dos paddings. Agora vamos colocar o conteúdo desse cartão.

[03:57] Próximo conteúdo vai ser a imagem, vamos estilizar a imagem. “.produto\_\_imagem {“. Como essa imagem tem que ficar flutuando, ela tem que ter um “position: absolute;”. E agora colocamos aqui que essa imagem vai ficar no topo, com 50%, ou seja, na metade, “left: -8rem;”.

[04:24] E agora “transform:”, para podermos subir metade dessa imagem, “translateY”, porque se deixarmos só o top 50%, o final da imagem vai ficar na metade do cartão, não queremos isso, queremos que o meio da imagem fique na metade do cartão.

[04:45] E com isso vamos colocar um “z-index:” aqui, que é para garantir que a imagem fique atrás do conteúdo e não tenha risco de a imagem sobrepor o escrito que vai vir à direita. E como cada imagem tem um tamanho diferente, vamos garantir que todas as imagens tenham o mesmo tamanho.

[05:03] E agora com isso podemos construir o conteúdo, aquela divisão. Então “.produto\_\_conteudo”. E dentro do conteúdo vamos só garantir que ele fique acima da imagem, para isso precisamos de um “position” que seja diferente de estático e agora podemos colocar um “z-index: 2;”, vamos garantir que as informações do produto vão ficar acima da imagem.

[05:27] Próximo, que vai ser o título. A estilização do título mesmo por si só já está pronta, só temos que nos preocupar mesmo com os espaçamentos, “.produto\_\_titulo {“, eu coloquei 2, beleza, vamos tirar o “margin-top:”, que o título destaque já tem, nós não queremos que esse margin top já exista, porque tem um padding top já. E agora um “margin-bottom:” também, queremos um margin customizado embaixo, porque o margin de baixo é de 0.75rem.

[06:01] Próximo vai ser o preço, “.produto\_\_preco {”, também ele é um texto comum, não temos que nos preocupar com a estilização porque ele já vai vir do próprio corpo da página, então só vamos mexer com espaçamento. “margin-bottom: 1.5rem;”. E agora o botão comprar.

[06:24] Para isso vai ser “.produto\_\_comprar {“ e a cor vai ser “color: var(--amarelo);”. E como dentro do próprio botão vai ter um ícone do lado, podemos deixar aqui, por exemplo, um “display: flex;”, e vamos alinhar verticalmente a palavra comprar com o ícone de seta, então “align-items: center;”.

[07:04] O próximo agora vai ser o ícone, “.produto\_\_comprar--icone {“. Esse ícone, como vai ser um span, não podemos dar uma dimensão para ele sem antes mudar o tipo de display dele, então podemos colocar, por exemplo, “inline-block;”, que vai garantir que vai ficar uma linha na outra, mas como tem o flex aqui, não teria problema, poderíamos usar outra coisa, mas inline block vai garantir que fique um do lado do outro.

[07:36] Colocamos agora a largura, “0.875rem;”, a altura é a mesma coisa, “0.875rem;”. Uma margem à esquerda e agora realmente a imagem do ícone, então “background-image: url(../../img/icones/seta.svg);”. E agora um “background-size:”, nós queremos que essa imagem não ultrapasse o tamanho que nós definimos, então “contain;”.

[08:15] Agora montamos todas essas classes CSS e queremos colocar no html para testar. Então nós vamos apagar esse article agora e vamos montar um Emmet do zero. Esse Emmet vai ser bem comprido, então vamos lá com calma para podermos entender o que está acontecendo aqui.

[08:40] “article.”, dentro do article nós precisamos da classe “produto>”, porque esse article vai ser nosso cartão, dentro do cartão vamos ter dois elementos, a imagem e a divisão de conteúdo. A imagem tem a própria classe, que é “produto\_\_imagem”, no mesmo nível hierárquico, com “+”, ele tem a “div”, que vai ser o “produto\_\_conteudo”.

[09:10] Dentro de produto conteúdo nós temos o título, que vai ser o “h3”, então ele é o “produto\_\_titulo”. Só que esse produto título só tem o espaçamento, também precisamos do “titulo--destaque”.

[09:31] No mesmo nível do título destaque temos o preço do produto, que é só um “p” comum. Só que esse p comum tem um espaçamento embaixo, então é “produto\_\_preco” e agora no mesmo nível também tem o botão, que é o a, “a.produto\_\_comprar”. E dentro desse link nós temos o ícone do lado da seta, então dentro aqui temos o “span”, que vai ser o “produto\_\_comprar--icone”.

[10:08] Rodando esse Emmet agora, nós só colocamos as imagens aqui, “assets/img/produto-01.png”, o alt vai ser “Foto do produto”, o título vai ser, qual é o nome daquela planta? Vamos ver aqui, “Ajuga reptans”. O preço é 20 reais, “R$ 20,00”. O href nós não temos, vou colocar um cerquilha aqui dentro e agora dentro do a é “Comprar”.

[10:52] Salvei o html, já está importado o produto, recarreguei o navegador e aqui está a planta. Ficou certo e a imagem, como você pode ver, ficou meio estranha, mas não tem muito o que possamos fazer, porque as imagens que estão vindo do Figma estão meio estranhas.

[11:14] Eu tentei fazer aqui de um jeito que importando qual imagem a gente coloca aqui dentro, produto 1, produto 2, ele fique mais ou menos na mesma posição, mais ou menos parecido com o que temos no Figma. Mas quando tivermos o produto final mesmo, as imagens reais, as imagens já vão vir cortadas num tamanho específico para caber nesse cartão. Por hora não se preocupe que a imagem está um pouco diferente do que está no Figma, porque vai ficar estranho mesmo.

[11:42] Agora nós montamos o nosso cartão aqui, o nosso componente de produto. Vamos para o próximo componente.

# 04Styleguide - O vídeo

[00:00] Vamos para a última seção, que vai ser a seção de vídeo. Aqui no Figma nós temos uma imagem que vai ser o banner do vídeo, o botão de play, um título e uma inscrição.

[00:13] Só algumas coisas para se notar aqui, como não temos o vídeo em si, vamos usar uma imagem de fundo, vamos usar uma tag img e depois um pseudo-elemento que vai ser esse botão de play.

[00:25] Só toma cuidado, que imagem não tem um pseudo-elemento, então não podemos usar, por exemplo, img: before para poder colocar essa imagem de play, temos que colocar um container em volta dessa imagem, que esse container vai ter o botão de play, aí sim conseguimos colocar o pseudo-elemento de before em cima da imagem.

[00:46] E agora o título desse vídeo é o mesmo título que tem na unidade, Rio de Janeiro e São Paulo, mas eles vão se repetir, só que o título é muito específico para esses dois componentes, então a melhor coisa que podemos fazer aqui é deixar essa estilização isolada, mas podemos criar as variáveis que esses títulos vão usar.

[01:13] Vamos para o nosso editor de texto, vamos criar um novo componente dentro da pasta de componentes chamado “video.css”. Dentro de vídeo.css vamos primeiro estilizar aquele cartão invisível que fica em volta, que engloba a imagem, o botão, o título e descrição. Vou chamar de “.cartao-de-video {”.

[01:34] Nesse cartão de vídeo, como ele não tem background, não tem nada, ele só tem a distância desse conjunto para o conjunto de baixo, só vamos colocar um margin bottom nele, então “margin-bottom: 2rem;”. E agora vamos estilizar a div, que vai ficar em volta da imagem. “.cartao-de-video\_\_video {“ e aqui dentro vamos deixar o “position: relative;”, que vai deixar a imagem do vídeo como referência para o botão de play ficar no meio.

[02:13] Vou colocar aqui uma largura, “100%”;, uma altura máxima, “17.5rem;”, e qualquer imagem que vaze essa altura vai ficar escondida, então “overflow: hidden”. E agora colocar um “margin-bottom: .5rem;”.

[02:39] E agora vamos fazer o pseudo-elemento, que vai ser o botão de play. Então “.cartão-de-video\_\_video: :before {“. O “content:” tem que existir, independente se vai ter algum conteúdo ou não, posicionamento absoluto, com “left: 50%;”, “top: 50%;” para centralizar o botão de play, e agora vamos colocar um “transform: translate(-50%, -50%);” para voltar metade do botão de play, tanto para cima quanto para o lado.

[03:21] Agora temos que deixar um “display: block;”, senão o tamanho não vai funcionar. Largura vai ser de “4.875rem;” e a altura também vai ser o mesmo tamanho.

[03:42] Próximo passo vai ser deixar o botão de play redondo, então “border-radius: 50%;” e agora vem as propriedades de fundo, a cor e o ícone em si, “background-color”, “backgroud-image:”, que vai ser o play. “no-repeat;” para não repetir a imagem. “background-size: 1.75rem;” e o position, que vai ser “center left 55%;”.

[04:34] Então nós criamos o cartão invisível de vídeo, nós estilizamos o container que vai ficar a imagem de vídeo e o botão de play, que vai ser o pseudo-elemento.

[04:46] Vamos criar o cartão de vídeo, ele vai ficar “article.cartao-de-video>”, dentro vai ter aquela divisão, que vai ficar com o “.cartao-de-video\_\_video>”, e dentro dele vamos ter uma imagem e essa imagem, como não temos realmente o vídeo, eu vou usar um gerador automático de imagem, que é esse link “[https://picsum.photos”](https://picsum.xn--photos-1i0c/) barra a largura, barra a altura da foto. Eu coloquei isso dentro do source da imagem. E o alt vai ser “Banner do vídeo”.

[05:34] Salvei o html, vamos ver como ele fica no navegador. abrir navegador, recarregar a página, descer. Faltou fazer o import desse CSS, “link”, “assets”. Quando eu falo que eu cometo esse erro até hoje é porque eu cometo esse erro até hoje, eu quase sempre esqueço de importar o CSS depois.

[06:02] Agora com o CSS importado, vamos ver, o botão de play está aqui e a imagem do nosso vídeo. Agora falta colocar o título. Como o título vai ser reutilizado no footer, mas não necessariamente vai ter um componente só para ele, vamos ter que estilizar esse título aqui no nosso vídeo.

[06:26] E esse “.cartao-de-video\_\_titulo {“ vai usar alguns valores de propriedades que podem ser repetidos, então podemos colocar dentro das variáveis e o que vai ser repetido vai ser o peso, que é o “--peso-destaque:”. Eu estou colocando o nome genérico porque vai ser usado em mais de um lugar, além do componente de vídeo. Para não colocar, por exemplo, título vídeo aqui e de repente usamos um título vídeo dentro do rodapé, que não faz muito sentido, eu deixei o nome genérico.

[07:03] Salvei, voltei para o vídeo e agora o título. “color: var(--preto);”, “font-size: var(--tamanho-destaque);” e agora “font-weight: var(--peso-destaque);”. E agora aqui só colocar um “margin-bottom: .5rem;”.

[07:52] Salvei o CSS, voltei para o html e agora, logo abaixo da div, fazer um “h3”, por exemplo, que vai ser o “.cartao-de-video\_\_titulo” e junto vai ter um “p”. E agora dentro desse h3 vai ter “Vídeo XYZ”. No texto é “Publicado em março de 2019”.

[08:20] Salvei o html, voltei para o navegador, recarreguei e pronto, a estilização está toda pronta. Deixa eu recarregar, que essa imagem ficou ruim de ver o botão no fundo. Perfeito. Agora temos uma imagem aqui de fundo, que vai ser o banner do vídeo, o botão de play, o título e a descrição do vídeo. Agora podemos finalmente fazer o rodapé.

# 05O cabeçalho

[00:00] Agora acabamos todo o corpo da página, falta agora fazer o cabeçalho e o rodapé do nosso layout. Vamos começar pelo cabeçalho.

[00:10] Nós temos uma logotipo e um carrinho, e ambos estão alinhados aqui, são só esses dois elementos. Vamos criar um novo componente dentro do nosso código, na pasta de componentes, “cabecalho.css”. Vou criar a classe “.cabecalho {“, que vai ter um “display: flex;”, “justify-content: space-between;”, para deixar a logo e o carrinho de compras separados igualmente. “align-items:”, fazê-los serem alinhados pela base, um “box-sizing: border-box;”, largura e agora os espaçamentos. “.75rem;” e “padding-bottom: .75rem;”.

[01:11] Agora fazer a estilização do logo, “.cabecalho\_\_logo {“, deixar só um tamanho definido, porque a logotipo é muito maior do que o espaço que tem disponível no cabeçalho para mobile, então “11.125rem;”. E agora fazer o carrinho, que vai ficar dentro do html. Deixar uma largura também, “1.75rem;”. E agora fazer, dentro do html, a estrutura.

[01:46] Aproveitar, já que eu tenho que subir tudo, logo depois do base eu vou importar o CSS do cabeçalho. “componentes/cabecalho.css”, ok. Antes do main vou criar a tag “header.container.cabecalho”.

[02:11] Dentro do header nós temos uma imagem que vai ser a de logo, “.cabecalho\_\_logo”, e junto nós vamos ter um “a”, que vai ser o carrinho. Dentro do “a” vai ter uma “img.cabecalho\_\_carrinho”.

[02:36] Rodar aqui o Emmet, eu só vou identar para ficar mais fácil de visualizar. Perfeito. O primeiro img vai ser “assets/img/logo.svg”. O alt vai ser “Logo da Casa Verde”, o href nós não temos, vou botar uma cerquilha dentro. Img, o próximo é “assets/img/ícones/carrinho.svg”.

[03:14] E dentro desse a, pensando em acessibilidade, podemos deixar, por exemplo, um “aria-label”, que vai ser “Meu Carrinho”, por exemplo. E agora o alt dessa imagem vai ser “Ícone do Carrinho de compras”, posso até mudar aqui dentro do aria-label.

[03:48] Salvei o html, fui para o navegador, recarreguei e ok, aqui está o nosso cabeçalho. Então agora vamos fazer o footer.

# 06Título do vídeo

Durante o desenvolvimento do componente de vídeo, tivemos uma discussão sobre o título do vídeo e se devemos componentizá-lo ou não. No final, optamos por não componentizar o título.

Escolha as alternativas que melhor representam o porquê.



# 07Faça como eu fiz

Praticar ajuda bastante no aprendizado de um novo conceito. Assim, é muito importante que você implemente o que foi apresentado nesta aula.

Não deixe de sanar suas dúvidas antes de dar continuidade ao curso. Estaremos te esperando no fórum da Alura caso alguma dúvida surja.

# 08O que aprendemos?

Nesta aula, aprendemos:

* Separar componentes
* Lidar com estilizações idênticas com conteúdos diferentes

# 01Projeto da aula anterior

Caso queira começar daqui, você pode baixar o projeto da aula anterior nesse [link](https://github.com/alura-cursos/styleguide-css/archive/Aula4.zip).

# 02O rodapé

[00:00] Depois de finalizado o cabeçalho, nós vamos ver qual é o último componente, que vai ser o rodapé. Vamos pegar o rodapé aqui no Figma e vamos fazer uma análise rápida.

[00:10] O rodapé consiste de uma logotipo, uma lista de redes sociais e depois duas informações sobre a unidade para entrar em contato com a empresa. Vamos focar primeiro nessa primeira parte, o logotipo e a lista de mídias sociais, e também essa imagem de fundo.

[00:27] Vamos lá no nosso código criar um novo componente dentro da pasta de componentes, chamado “rodape.css”. Dentro de rodapé nós vamos cuidar do fundo, “.rodape {“, colocar os espaçamento em cima, “1.5rem;”, o “padding-bottom:” também, para poder deixar o espaçamento embaixo, e agora colocar o fundo em si, então “background-color: var(--branco);”, a imagem, “(../../img/splash-rodape.svg);”, “background-position: top”.

[01:20] Esse posicionamento vai variar dependendo de como vamos pegar essa imagem, então talvez nós tenhamos que fazer alguns ajustes depois nesse posicionamento da imagem. “background-repeat: no-repeat;”. Agora cuidar da lista de mídias sociais.

[01:40] Na verdade, vou fazer a logo primeiro, “.rodape\_\_logo {“, deixar uma largura máxima para a logotipo, “11.125rem;”, “margin-bottom: 1.25rem;”. E agora a lista de mídias sociais, “.rodape\_\_social {“, “display: flex;”. E agora o espaçamento baixo, que vai ser de “2rem;”. E agora nós vamos cuidar de cada mídia social.

[02:18] Nós vamos deixar os ícones como imagens dentro do html, então aqui só vamos estilizar o a por si só. Então “.rodape\_\_social-media {“, “display:” tem que ser “inline-block;”, alguma coisa diferente de inline, para poder colocar altura e largura, “1.625rem;”, altura “1.625rem;” e agora colocar o espaçamento entre uma mídia social e outra, que também vai ser de “1.625rem;”.

[03:02] Nós terminamos essa primeira parte e aqui no html importar o CSS, “assets/css/componentes/rodape.css”.

[03:15] E depois do main criar aqui um “footer.container.rodape>”, dentro do footer temos a imagem, que vai ser “img.rodape\_\_logo+”, abaixo da img temos a lista, que vai ser nossa “.rodape\_\_social>”, dentro nós temos três mídias sociais, então “li\*3” e dentro de cada li temos a tag “a”, que vai ser o “.rodape\_\_social-midia”, e dentro de cada mídia social nós vamos ter os ícones.

[03:54] Rodar esse código do Emmet e agora é só preencher com as informações que faltam. O logotipo nós já sabemos que está em “assets/img/logo.svg”. O alt vai ser “Logo da Casa Verde”. Dentro de cada a vamos deixar só o cerquilha, já que não temos o link e agora os imgs, “src=”assets/img/icones/”, qual era o primeiro? Vou voltar no Figma rapidinho, só para ver, Facebook, Twitter e Instagram. Vou voltar aqui, Facebook. O alt vai ser “Ícone do facebook”.

[04:48] Aproveitando, dentro de cada a, colocar também o atributo “title”, que vai ser para dizer para o leitor de tela que esse link vai para o Facebook. Já que não temos o link propriamente dito, mesmo se tivesse, o leitor de tela não sabe exatamente para onde vai esse link, então temos que deixar a tag title aqui também. Então “Facebook”. A imagem está pronta, ikyrie está pronto, legal, vamos para o próximo.

[05:19] Dentro do próximo a, “title=’Twitter’”, img é “assets/img/icones/twitter.svg”. “Ícone do Twitter”. Próximo a, “title=’Instagram’”, “assets/img/icones/instagram.svg”. Para completar aqui o último alt, “Ícone do Instagram”.

[05:59] Salvei o html, vou para o navegador, recarreguei a página, lá no final está o nosso rodapé, já bonitinho com essa parte pronta. Agora o que falta no nosso Figma são as unidades.

[06:13] Só que essas unidades têm um detalhe muito importante de estruturação, que é onde as informações vão ficar. Esse Rio de Janeiro é um h3? É um h2? Essa Rua Siqueira de Campos, é um p? É um span? O que ele é?

[06:29] Então quando temos informações de contato, o ideal é deixar essas informações dentro de uma tag chamada address, todas as informações. A unidade, o endereço, o bairro e telefone, que fiquem dentro dessa tag address. Vamos fazer então primeiro só Rio de Janeiro. Como vai ficar isso?

[06:50] Logo depois da ul eu vou criar uma nova tag “

”. Dentro de address nós vamos deixar aquele título. Como esse título da unidade tem um destaque, podemos deixar só essa parte separada com alguma coisa, como, por exemplo, um span. E nesse “” vai ser “Rio de Janeiro” e abaixo nós colocamos as outras informações, que é esse endereço, vou copiar e colar.

[07:31] Só que se salvarmos desse jeito, olha só o que acontece, vou voltar para o navegador, todas as informações ficaram na mesma linha. Algumas coisas que vamos ter que fazer aqui, primeiro é fazer o Rio de Janeiro, além de ter o destaque, fazê-lo quebrar linha. Isso nós resolvemos até com um display block, mas e o restante dessas informações? Como eu faço para poder pular essa Rua Siqueira de Campos, da Copacabana e telefone?

[08:03] Nesse caso vamos usar uma tag, que geralmente as pessoas não gostam de usar essa tag, mas ela tem uma função muito importante neste caso, que é a tag “”.

[08:15] A tag br antigamente era usada para poder dar espaçamento entre um elemento e outro, só que agora com margins, paddings, não precisamos mais usar o br. E o br tem a função de quebra de linha, e é isso que queremos, queremos dar o Enter, o br simula exatamente o Enter que nós colocamos aqui.

[08:35] Então colocando os brs nos lugares devidos, aqui não precisa de br, voltei para o navegador, recarreguei a página e, olha só, ele quebrou a linha aqui. Depois do 303 ele pulou a linha, Copacabana, pulou a linha, telefone. Agora podemos cuidar da estilização disso aqui.

[08:54] Voltando para arquivo de rodapé, vamos criar agora os CSSs desses elementos. Vamos começar pela unidade, então “.rodape\_\_unidade {“. Esse unidade é o todo, lembrando que a unidade é o conjunto do nome e a localização. “margin-bottom: 2rem;”. “.rodape\_\_unidade” e agora vamos para o título.

[09:30] Lembrando, a estilização, por mais que seja a mesma do cartão de vídeo, elas são específicas de cada componente. Então aqui var(--preto);”, “font-size:”, nós já criamos todas essas variáveis, são os destaques, “var(--tamanho-destaque);”, “font-weight: var(--peso-destaque);”.

[10:05] Agora o “display: block;”, para poder quebrar aquela linha do título da unidade e um “margin-bottom:”, que dá o espaçamento entre aquele texto e a parte de baixo, “.5rem;”. Salvamos aqui e agora colocar essas classes no nosso html.

[10:28] Então “

” agora vai ter a classe “rodape\_\_unidade” e também aproveitar e colocar o “texto”, que por causa dos espaçamentos entre uma linha e outra, como tudo é um texto só, nós colocamos aquele line-height dentro do base.css, podemos usar aqui usando a classe de texto.

[10:54] Agora dentro de ” e pronto, salvar isso, vou voltar para o navegador, recarreguei a página e pronto, a unidade já está aqui, Rio de Janeiro, com a rua toda certa e agora só falta fazer para São Paulo. Para isso, o que eu vou fazer só é copiar esse address inteiro e trocar as informações.

[11:21] Então “São Paulo”, qual é a rua? Vou voltar para o Figma, “Ctrl + C”. Vou colocar os brs agora, “”. Salvei o html, voltei para o navegador e, pronto, nosso rodapé está finalizado e assim todos os componentes da página já foram montados. Agora o que nos falta é montar todo o projeto.

# 03Planejando a Home

[00:00] Agora que finalizamos todo os componentes, podemos atacar a primeira página aqui. Vamos analisar novamente o que essa página é, do que ela é composta.

[00:11] Começando aqui de cima, temos o cabeçalho, a estrutura do cabeçalho, os seus componentes já estão prontos, então basta importarmos simplesmente o cabeçalho com as classes e já temos a estrutura toda montada.

[00:23] O fundo com essas listras e essa textura de mármore também já está pronto. A seção agora de chamada é composta por um input e um botão, ambos já estão prontos. O componente de título, da tipografia, também já está pronto. E a única diferença desse p para os outros é que ele está com uma largura limitada, mas é a mesma coisa.

[00:50] Agora nós vamos começar a pensar na estilização que vai ser individual para cada seção, porque com os componentes aqui nós já sabemos como monta esse input, é só colocar input e as classes que precisamos. O botão também, o botão e as classes que ele precisa. Mas para montar esse p, esse parágrafo que está cortado, ele é uma coisa específica dessa seção, então vamos usar a estilização da seção.

[01:16] Vamos ver o próximo, o “Como conseguir”. Nós praticamente montamos já, a estrutura pronta desse cartão, mas também, espaçamentos, qualquer coisa que seja diferente dessa seção, precisamos colocar no arquivo específico dessa seção.

[01:31] Produtos, ok também, nós temos todos já esses componentes e agora os de vídeo também, tem outro espaçamento aqui dentro, que é o espaçamento de fora vezes dois, tem um espaçamento dentro dessa seção de novo. E o rodapé também já está pronto, é a mesma coisa que já fizemos antes.

[01:54] Agora que já sabemos mais ou menos como vai ser o nosso projeto, que são só algumas seções que nós vamos ter que estilizar para uma coisa específica, nós vamos criar um novo arquivo e a ideia é que esse arquivo se chame index.html, só que já existe um index.html, que é esse arquivo onde nós fizemos todos os testes dos componentes.

[02:14] Por enquanto eu vou pegar esse arquivo index.html e vou trocar o nome dele para, por exemplo, “componentes.html”, e agora sim eu vou criar um novo arquivo, chamar de “index.html”.

[02:29] E agora dentro desse arquivo index.html nós vamos colocar a estrutura básica de um documento html. Vou usar o Emmet aqui, “html:5”, fazer as mudanças do “html lang” para “pt-br” e o restante, o “Document” vai ser “Home | Casa Verde”. E agora já podemos importar nosso arquivo base, “link”, “href=’assets/css/base/base.css’”.

[03:04] E só com esse arquivo nós já temos a tipografia, nós já temos o reset e nós já temos as variáveis, todas dentro de um arquivo só, dentro de um import só.

[03:14] Também é importante colocarmos a fonte, então vou pegar como referência, no nosso arquivo antigo, tem já as fontes aqui, só vou copiar e colar no nosso novo index. Perfeito. Agora nós temos a estrutura básica de um documento html, já com as fontes importadas e com as estilizações básicas importadas.

[03:35] Agora o processo vai ser simplesmente, dentro do body, adicionando os elementos que precisamos, e conforme precisamos de algum componente novo, adicionamos no css e montamos a estrutura aqui dentro do body.

[03:50] O que fazemos com essa página agora? Uma recomendação que eu tenho é não apagar essa página, porque essa página é muito importante para nós, porque ela vai servir como documentação para saber como vamos montar cada estrutura.

[04:03] Então como monta um header? O header está aqui. Como eu monto um cartão? Está aqui um cartão. Tem toda a estrutura, todas as classes que precisamos usar para poder montar cada um dos componentes.

[04:18] Eu recomendo muito que você deixe essa página viva ou, se for o caso, até poder comentar depois aqui, por exemplo, “Estrutura do Cabeçalho”, alguma coisa do tipo, para você ter isso como documentação, como vamos fazer para montar cada componente. Por enquanto é isso, na próxima aula vamos fazer cada uma das seções.

# 04Usando componentes

[00:00] Agora que fizemos a estrutura básica do nosso documento html, podemos começar realmente a montar a nossa página. Voltando para o Figma, vamos começar pelos componentes cabeçalho e rodapé.

[00:12] Existem algumas maneiras que podemos colocar esses componentes dentro da nossa página, eu vou mostrar duas dessas maneiras. Voltando para o código, no cabeçalho vou mostrar a primeira maneira, que é usando o Emmet.

[00:25] Podemos começar com a estrutura “header.cabecalho” e agora o espaçamento lateral, que vai ser o “.container”. E dentro do header nós temos a imagem, que é a logotipo, e a imagem que é o carrinho. Vou colocar aqui a primeira imagem, que vai ser “.cabecalho\_\_logo” e no mesmo nível nós temos o “+a>” e dentro desse a tem a imagem do carrinho, “img.cabecalho\_\_carrinho”.

[00:58] Rodando esse código do Emmet, fazendo os ajustes, deixar eu dar o espaçamento aqui pra podermos visualizar melhor e vou fazer umas mudanças, então “img src=’assets/img/logo.svg’”, alt vamos colocar “Logo da Casa Verde”, o href nós não temos, então podemos colocar cerquilha e agora colocar um “aria-label” também, e colocamos “Carrinho de compras”. Dentro de img agora “assets/img”, é um ícone, então agora temos aqui “carrinho.svg”. O alt vai ser “Ícone do Carrinho de compras”.

[01:53] Nós salvamos agora esse index.html, vamos para o navegador, ainda está aqui o index antigo, recarregar a página e eu fiz alguma caca. A caca que eu fiz foi simplesmente não ter importado o componente. Novamente, é um erro que eu faço até hoje. “link”, “assets/css/componentes/cabecalho.css”.

[02:21] Salvei o arquivo, voltei para o html, recarreguei a página e está aqui, logo Casa Verde e carrinho. Agora vamos fazer o rodapé. E como é o segundo jeito de inserir um componente aqui?

[02:34] Vamos voltar para aquele nosso outro arquivo, que vai nos servir como documentação, vamos ver onde tem o componente de rodapé. Está aqui, é esse footer. Vamos pegar toda a tag, toda a estrutura aqui, “Ctrl + C”, voltar para o index, “Enter” depois da tag header, “Ctrl + V”. Salvei o arquivo e agora vamos importar todos os componentes que esse footer exige. Precisamos do rodapé, do que mais precisamos? Texto já vem do base, então é só rodapé.

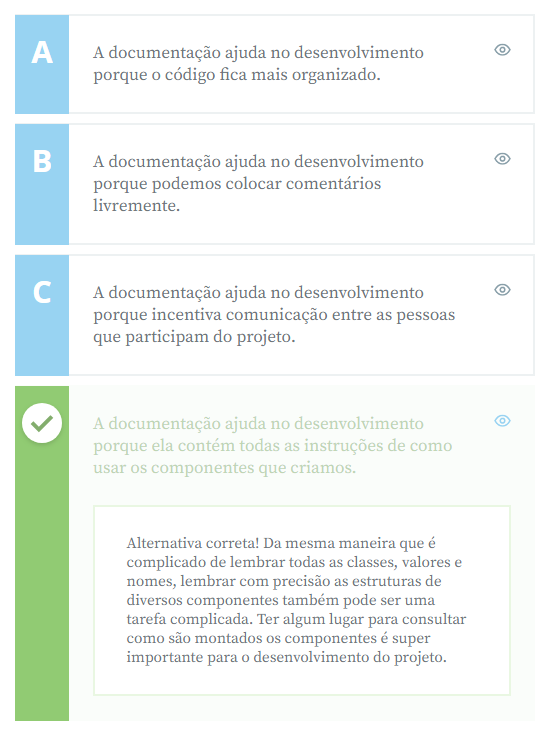
[03:16] Então logo depois de cabeçalho, “link”, “assets/css/componentes/rodape.css”. Salvei o documento, voltei para o navegador, recarreguei a página e agora eu não sei o que está acontecendo. Beleza, funcionou. E nós temos agora o nosso rodapé todo montado, então temos o cabeçalho e o rodapé já prontos aqui.

[03:50] Essas são as duas maneiras que podemos fazer para colocar alguns componentes dentro da nossa página, tanto vindo na nossa documentação informal, copiando e colando e fazendo modificações, caso seja necessário, ou fazer como fizemos com o cabeçalho, que foi montando a estrutura do Emmet. Então vai variar bastante o que podemos fazer, o que seria mais rápido e eficiente. Vamos agora para a nossa primeira seção, que vai ser a seção de chamada.

# 05Documentação dos componentes

Depois que terminamos a fase de criar cada um dos componentes que serão utilizados na página Home, nós renomeamos o arquivo que contém os componentes e criamos uma nova página index.html, fazendo uma espécie de documentação.

Por que a criação dessa documentação é importante para o processo de desenvolvimento do projeto? Escolha a alternativa que melhor representa a justificativa.



# 06Faça como eu fiz

Praticar ajuda bastante no aprendizado de um novo conceito. Assim, é muito importante que você implemente o que foi apresentado nesta aula.

Não deixe de sanar suas dúvidas antes de dar continuidade ao curso. Estaremos te esperando no fórum da Alura caso alguma dúvida surja.

# 07O que aprendemos?

Nesta aula, aprendemos:

* Como planejar o desenvolvimento de uma página com as ferramentas que criamos
* Maneiras de implementar componentes
* Importância de uma documentação

# 01Projeto da aula anterior

Caso queira começar daqui, você pode baixar o projeto da aula anterior nesse [link](https://github.com/alura-cursos/styleguide-css/archive/Aula5.zip).

# 02As primeiras seções

[00:00] Agora que finalizamos o cabeçalho e o rodapé, nós podemos partir para a próxima seção, a primeira seção da página, que vai ser a seção de chamada. Aqui no Figma temos a seção e vamos lembrá-la então.

[00:11] Logo no começo nós temos um título composto, logo abaixo um parágrafo com o tamanho limitado, uma foto de uma planta à direita, um input com “Insira seu e-mail” e um botão. Grande parte dos componentes, com exceção só da imagem, nós já temos todos os componentes prontos, então já podemos começar a desenvolver essa seção já com os componentes que temos.

[00:33] Voltando para o código, logo abaixo de header, eu vou criar uma nova tag chamada “main” e dentro dessa main vão ficar todas as seções que nós temos, todo o corpo da página. E agora criar uma “section.container” que vai ser a nossa primeira seção chamada, e agora dentro nós temos nosso título composto, “h2.titulo>span.titulo--destaque”. Já podemos colocar aqui então nosso título, que vai ser sua casa com as melhores plantas, “Sua casa com as” e aqui dentro do span “melhores plantas”.

[01:15] E logo abaixo nós temos finalmente o parágrafo, a imagem, o input e o botão. O parágrafo é um parágrafo comum, então ele é só um “p.texto”, no mesmo nível nós temos a “+img”. Como nós não temos nenhuma classe específica para ela ainda, é um componente que não criamos ainda, a gente se preocupa com atributos e classes.

[01:39] Agora junto, no mesmo nível, vamos ter o “+form”, que vai ser o formulário, porque o input e o botão têm que ficar dentro de um formulário, e dentro desse form nós colocamos agora nosso “>input” com a classe “.input”, com a classe “.input--icone”, porque esse input vai ter um ícone, e esse ícone vai ser um ícone de e-mail, e no mesmo nível nós temos um “+button” que vai ser a classe “.botao”.

[02:06] Rodando o código do Emmet, agora vamos completar com os textos, aqui. A imagem vai ser “assets/img/planta-chamada.png”, o alt vai ser “Foto do produto”, o action nós não temos ainda, vou colocar aqui um cerquilha, aproveitar já e identar os elementos aqui dentro, para podermos ver melhor.

[02:35] Agora dentro do input faltar o deixarmos acessível, falta colocar o “placeholder”, que vai ser “Insira seu e-mail” e o “aria-label”, que vai ser o mesmo texto, “Insira seu e-mail”. O botão vai ser “Assinar newsletter”. Salvei o documento, agora falta importar as componentes que nós usamos agora na seção.

[03:04] Logo depois do cabeçalho eu vou criar um novo link, “assets/css/componentes/input.css”, e “link”, “assets/css/componentes/botoes.css”. Salvei o documento html, voltei para o navegador, vou recarregar a página e aqui estão os nossos elementos componentes.

[03:28] Aproveitar aqui e só dar um ajuste no botão que nós não arrumamos ainda, os espaçamentos dele. Eu vou abrir aqui botoes.css e trocar esse “padding-top: 1rem” para “1.75rem”, e o “padding-bottom” para “1.75rem”, porque faltou só arrumar esse espaçamento do botão para deixá-lo do mesmo tamanho do input. Carreguei a página, beleza, tudo pronto.

[03:50] Agora podemos fazer as estilizações que são exclusivas dessa seção, como, por exemplo, o espaçamento entre a seção e outra, também a planta, que vai ter que ficar à direita e aquela imagem splash de fundo. Então vamos criar um novo arquivo que vai ser específico da seção.

[04:09] Vamos na raiz da pasta “css”, vamos criar um novo arquivo chamado “chamada.css”, e dentro agora vamos fazer a estilização. Começando com “.chamada {“, que vai ter um “padding-top: 2.5rem;”, um “margin-bottom: 2.5rem;” e agora aquela imagem de fundo, “background-image: url(../../assets/img/splash-chamada.svg);”.

[04:44] E agora o posicionamento dela, vai ser por “background-position”, “top -4rem right;”. E não queremos que esse background se repita, então “background-repeat: no-repeat;”. E agora só deixá-lo com posicionamento relativo.

[05:03] Eu já vou explicar o que é esse posicionamento relativo, quando chegarmos na imagem. O próximo passo agora é deixar o texto limitado. Vamos aqui, “.chamada\_\_texto {“, deixar a largura dele para “12.rem” e agora um espaçamento abaixo, “margin-bottom: 1.55rem;”. Agora vamos fazer a imagem, “.chamada\_\_imagem {“.

[05:32] Essa imagem nós temos que deixá-la à direita do nosso texto, só que temos que deixá-la flutuando para poder ficar atrás do nosso formulário, então vamos ter que mudar o posicionamento dessa imagem para alguma coisa como “absolute”. E por causa disso temos que ter referência do nosso chamada, por isso que ele está com o posicionamento relativo. Agora podemos colocar aqui um “top: 3.25rem;” e um “right: 0;”, para ele ficar totalmente à direita.

[06:02] E para garantir que essa imagem vai ficar abaixo do nosso formulário, além de também flutuar na nossa página, precisamos deixar que o formulário fique acima da imagem. Só arrumar também um problema que eu fiz aqui, “.chamada\_\_imagem”, coloco aqui “.chamada\_\_formulario {“. E dentro desse formulário vamos deixar “position: relative;” e um “z-index” acima de um. Agora sim vamos garantir que o formulário vai ficar assim.

[06:36] Salvei o chamada.css, vou no meu documento html importar esse arquivo que nós criamos, então “link”, “assets/css/”, não é mais componentes, é só “chamada.css”. Salvei o arquivo e agora vamos importar as nossas classes. Então container vai ser “chamada”, junto de “container”, o título está certo, não vamos mexer em nada, o texto agora vai ter um “chamada\_\_texto”.

[07:11] A imagem nós criamos agora uma classe para ela, então vai ter “chamada\_\_imagem” e o formulário também vai ter a sua própria classe, classe “chamada\_\_formulario”. Salvei o arquivo html, vamos ver no navegador, recarreguei a página, perfeito. Eu já tenho aqui a nossa seção chamada pronta. Podemos ir para a próxima seção. Vou voltar para o Figma e a próxima seção é exatamente o mesmo cartão que nós montamos lá dentro do nosso exemplo.

[07:49] Então aqui vamos fazer do mesmo jeito que nós fizemos com o rodapé, vamos criar uma nova seção só, “section.container”, e dentro da seção container podemos vir dentro de componentes.html e cadê? Esse cartão que nós temos aqui. Vamos simplesmente pegar todo esse cartão, copiar e colar aqui dentro. E agora vamos só importar os arquivos dos componentes que nós vamos utilizar aqui, vamos ver quais são.

[08:21] Precisamos do cartão e do item, então vamos lá, logo depois de chamada vamos importar “link”, “assets/css/componentes/cartao.css”, e também “link”, “assets/css/componentes/item.css”. Salvei o arquivo html, voltei para o navegador, recarreguei a página e nosso cartão está completamente pronto, só falta arrumar agora os espaçamentos em cima e embaixo.

[08:52] Para isso vamos criar só um arquivo rápido aqui, junto com chamada.css, vamos criar um arquivo chamado “como-conseguir.css”. Dentro dele agora vamos colocar só o espaçamento entre uma seção e outra, que vai vir do “.como-conseguir {“, “margin-bottom: 5.625rem;”. Salvei esse arquivo, no html vamos importar também esse arquivo, “link”, “assets/css/como-conseguir.css” e agora aplicar aqui dentro, junto com a classe container, “como-conseguir”.

[09:36] Salvei o html, voltei para o navegador e, perfeito, a seção já está pronta, com os espaçamentos corretos e agora também o espaçamento entre uma seção e outra. Vamos para a próxima seção.

# 03Seção produto

[00:00] Partindo para a próxima seção, que vai ser a seção de produtos, vamos começar pelo título composto, que é o que já temos aqui. Vou criar mais uma section, “section.container”, dentro dessa section podemos colocar “h2.titulo” e esse título também, se olharmos no Figma, ele está centralizado, então já podemos usar aquela nossa classe “.alinhamento--meio>”. E dentro desse h2 nós temos o “span”, que vai ser o “.titulo--destaque”.

[00:34] Rodando esse código, agora só precisamos arrumar aqui a identação. E qual é o título? Conheça nossos produtos, “Conheça nossos” e “produtos” aqui dentro do span. Essa próxima parte, vamos ver aqui no Figma, eu quero montar com vocês usando coisas super legais do Emmet.

[00:57] Nós temos aqui seis cartões com o preço que vai ser a mesma coisa e o botão de comprar, que vai ser a mesma coisa também em todos. O que eu quero mostrar para vocês é o seguinte, nós temos a ul e dentro da ul nós temos seis lis. Para dizer que temos seis lis, vamos usar “li”, o operador de multiplicação, que é o asterisco, e o número de lis que queremos, que é o seis. Rodando esse código, ele vai nos gerar uma ul com seis lis dentro.

[01:24] Só que vamos apimentar um pouco as coisas, o conteúdo vai ser o mesmo daquelas lis, pelo menos o preço, então para poder dizer que o conteúdo das lis vai ser igual, podemos usar chaves, e dentro dessas chaves nós colocamos o conteúdo que vai ser repetido em cada um das lis, por exemplo, “20,00”. Rodando esse código, ele vai gerar uma ul com seis lis dentro, e cada li vai ter 20,00 escrito, 20 reais, por exemplo.

[01:55] Uma coisa que podemos fazer também com esses valores é colocar, por exemplo, qual li é essa, se é a primeira a li, a segunda li, terceira li, já que temos um multiplicador de seis, podemos fazer um contador de li.

[02:11] Para isso vamos usar o cifrão aqui dentro do conteúdo, rodar esse código e olha só o que ele nos mostrou, ele colocou 1, 2, 3, 4, 5, 6, que é a quantidade de lis que nós temos, que é a quantidade de vezes que nós replicamos a li aqui.

[02:27] Isso aqui vai ser bem útil para nós quando formos colocar, por exemplo, o caminho das imagens, já que os caminhos, cada imagem que nós colocamos, salvamos lá no começo, têm o nome produto 01, produto 02, produto 03, então podemos usar esse operador de contagem para poder nomear o caminho das imagens. Vamos ver o que mais podemos fazer com o Emmet.

[02:54] Podemos, por exemplo, mexer com atributos desses elementos. Só rapidamente apagar essa li, posso botar aqui vezes seis, e dentro de cada li vou colocar uma imagem. E nós queremos mexer com o caminho dessa imagem, por exemplo. Para mexer com o atributo src, o source, podemos usar os colchetes, e agora dentro dos colchetes podemos colocar os atributos, que vão ser src. Então “[src=”assets/img/produto-01.png”]”.

[03:35] Só que podemos fazer melhor que isso, ao invés de 01, usamos 0 contador. Rodando esse código rapidamente aqui no Emmet, olha o que ele nos gerou. Nós temos seis lis com uma imagem, que são seis produtos e cada um desses produtos vai ter o caminho do arquivo certo, que é o produto 01, 02, 03, 04, 05, 06, para cada um dos cartões.

[04:01] E podemos também alterar esses alts, como, por exemplo, aqui, “alt=’Foto do produto’”. Rodando esse código do Emmet, ele nos gerou agora seis imgs, com produto 01, 2, 3, 4 e os alts, com todos contendo o “Foto do produto”.

[04:21] Agora vamos montar esses produtos decentemente. Então vamos criar aqui, nós temos uma “ul>”, dentro dessa ul vão ser seis lis, dentro de cada li vai ter que ter a classe “.produto”, porque cada uma das lis vai representar um produto. E dentro desse produto nós temos uma imagem, que vai ser “.produto\_\_imagem” e agora temos que alterar esse atributo dessa imagem, colocando “[src=’assets/img/produto-0$.png”]”. Aproveitando e mudando o “alt” para “Foto do produto”.

[05:09] E agora, no mesmo nível daquela imagem, nós temos aquele container só para o conteúdo, então é “+div.produto\_\_conteudo>” e agora aqui dentro nós temos um “h3” que vai ser o “produto\_\_titulo” e titulo--destaque”, porque aquele título do cartão vai ter o destaque igual tem o título em produtos que colocamos.

[05:37] Continuando a escrever o código aqui, no mesmo nível vai ser um “p.produto\_\_preco” e o que podemos fazer já é deixar esse preço definido, então vamos usar as chaves para poder deixar o nosso conteúdo escrito, “R$”. Só que não podemos deixar R$ direto, porque cifrão é o contador.

[06:02] Para podermos usar o símbolo cifrão temos colocar uma barra invertida antes desse cifrão. Agora o Emmet sabe que esse cifrão não é para ler como um contador, mas sim como um cifrão mesmo, o símbolo. Agora podemos colocar aqui “20,00”. E no mesmo nível nós temos aquele nosso botão de comprar, que é um “a.produto\_\_comprar>” e dentro nós temos o “span.produto\_\_comprar--icone”.

[06:38] Rodando esse código agora no Emmet, perfeito. Foi todo o código gerado perfeitamente, só faltou colocar o valor desse a, vou colocar rapidamente, que tem o valor de “Comprar”. Voltei aqui, rodar de novo o Emmet.

[06:56] Nós temos nosso valor já escrito com o cifrão arrumado, nós temos cada produto com o valor 01, 02, 03, 04 e nós temos o botão de comprar pronto. Agora só falta colocar os títulos.

[07:09] Vamos lá, pegar cada um desses títulos dos cartões e colocar aqui. Esses nomes em latim. Próximo cartão, esse é o cartão 3, cartão 4, cartão 5 e agora o último cartão. Salvei o html, agora falta importarmos as classes desses componentes, então falta do produto e só.

[07:54] Vamos aqui no topo, depois de como-conseguir nós criamos um “link”, “assets/css/componentes/produto.css”. Salvei o html, fui para o meu navegador, recarreguei, estão todos nossos produtos aqui, perfeitamente colocados e importados.

[08:15] O que falta fazermos só é arrumar os espaçamentos entre uma seção e outra. Então vamos criar um novo arquivo, que vai ser “ofertas.css” e vamos colocar aqui uma classe “.ofertas {“, “padding-bottom: 3.9rem;”.

[08:40] Salvei esse arquivo, fui para o index, aproveitar já o embalo que eu estou no topo, eu vou importar esse arquivo, “assets/css/ofertas.css” e aqui embaixo na seção vamos colocar, junto de “container”, “ofertas”. Salvei o html, fui para o navegador, recarreguei a página, legal, o espaçamento está pronto, agora podemos fazer a próxima seção.

# 04Seção vídeo

[00:00] Vamos agora para a próxima e última seção, que é a seção de vídeos. E a seção de vídeos nós vamos usar a mesma estrutura Emmet que nós montamos na seção passada.

[00:13] Então vamos, logo abaixo dessa section, criar outra “section” que vai ter “.container” e aqui dentro temos aquele título composto, com também o alinhamento ao meio e logo dentro temos um “span” que vai ser nosso “.titulo--destaque”. Rodando aqui, só vou dar os espaçamentos para enxergarmos melhor. E o nome do título vai ser veja aqui nossos vídeos, “Veja aqui nossos”, “vídeos”.

[00:53] Próximo passo agora vai ser montar aquela estrutura de vídeos. Temos uma lista de vídeos, uma “ul”. Dentro dessa ul nós temos quatro lis, são quatro cartões de vídeo, então “li\*4”, e cada li vai ser um cartão de vídeo. E dentro desse cartão de vídeo também tem aquele espaçamento lateral, que vai ser um “.container”, e nós temos duas divisões dentro desse cartão, que vai ser uma só para imagens.

[01:27] Não podemos esquecer que não tem como colocar pseudo-elemento em tag de imagem, então para isso vamos ter que ter uma div com uma img dentro. Como queremos ter uma div com img aqui dentro e depois mais conteúdo fora, do mesmo nível da div, vamos usar um operador, que são os parênteses. Coloco aqui parênteses, é como se rodasse um código do Emmet dentro de outro código Emmet. Então vamos montar só a estrutura daquela div com o vídeo mesmo.

[01:57] Então “div.cartao-de-video\_\_video” e dentro colocamos “>img”. E a img, podemos colocar já o source dela, que vai ser aquele Picsum, então “[src=”[https://picsum.fotos/278/252”]”](https://picsum.fotos/278/252%E2%80%9D%5D%E2%80%9D). Na verdade, podíamos ter dado “Ctrl + C”, “Ctrl + V” no nosso componente que tínhamos criado ali no componentes.html, mas não tem importância, já fiz isso aqui. Alt vai ser “Capa do vídeo”.

[02:38] E agora que acabamos de fazer a div, podemos continuar o código Emmet ali como estávamos fazendo antes. No mesmo nível nós temos um “h3”, que é “cartao-de-video\_\_titulo” e o conteúdo vai ser o mesmo, que até podemos usar chaves e escrever “Vídeo XYZ”. E o “p” e a data de publicação vai ser a mesma também em todos por enquanto, então “Publicado em março de 2019”.

[03:09] Rodando esse código Emmet, agora nós temos aqui toda a seção de vídeos pronta. Vamos salvar index.html, vamos importar o arquivo de vídeo, então lá no topo, depois de ofertas.css vamos importar aqui “link”, “assets/css/componentes/video.css”.

[03:33] Salvei aqui, vou para o navegador, recarreguei, dá um tempinho para carregar, pronto. Terminamos a seção de vídeos e com isso também terminamos toda a página.

# 05Documentação dos componentes

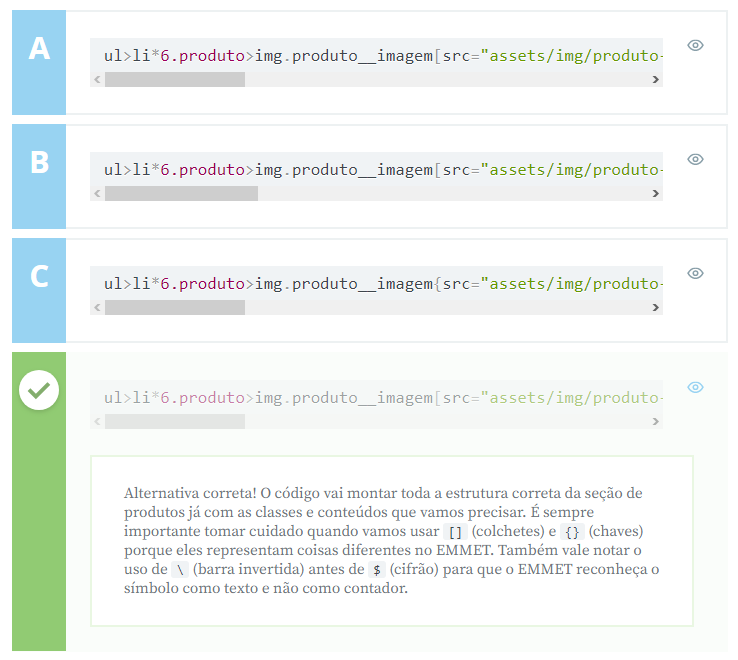
Durante esta aula nós vimos várias técnicas para implementação dos componentes, e uma dessas técnicas era utilizando a ferramenta EMMET. No decorrer do curso, vimos várias funcionalidades e fomos evoluindo cada uma delas para níveis mais complexos.

Vamos recriar a seção de ofertas usando EMMET:



Temos uma lista com 6 produtos. Cada um dos produtos precisa de uma imagem que segue o padrão de nome produto-01.png produto-02.png etc. Todas as imagens têm um texto alternativo "Foto do produto". Cada um desses produtos precisa de um espaço que vai ser reservado para os detalhes, como nome do produto, preço fixo de R$ 20,00 e um botão de comprar que vai ter um ícone junto.

Qual alternativa contém o código EMMET correto para a criação deste componente?



# 06Faça como eu fiz

Praticar ajuda bastante no aprendizado de um novo conceito. Assim, é muito importante que você implemente o que foi apresentado nesta aula.

Não deixe de sanar suas dúvidas antes de dar continuidade ao curso. Estaremos te esperando no fórum da Alura caso alguma dúvida surja.

# 07Projeto do curso

Caso queira, você pode baixar [aqui](https://github.com/alura-cursos/styleguide-css/archive/Aula6.zip) o projeto completo implementado neste curso.

# 08O que aprendemos?

Nesta aula, aprendemos:

* Como implementar tudo o que fizemos até agora no curso
* Estruturas mais complexas com EMMET

# 09Conclusão

[00:00] Chegamos ao final do curso e eu gostaria de parabenizar você por chegar até aqui. Também queria aproveitar esse tempo para poder fazer um apanhado do que nós vimos nesse curso.

[00:09] Começando lá no começo, que tínhamos aquela pergunta do que eram style guides, para que servem style guides. Já vimos o que fazem style guides e qual a importância de se ter um style guide no momento de desenvolver uma página.

[00:21] Vimos também como fazemos para arrecadar as informações desses style guides e colocar dentro do nosso projeto. No caso das cores, tamanho de fonte, família de fonte, peso de fonte, nós colocamos tudo dentro de um arquivo que foi o variaveis.css.

[00:38] E também vimos como fazemos para importar outros arquivos CSS dentro de outro arquivo CSS, como foi o caso do base. Nós importamos variáveis e o reset.css para que no momento em que formos criar uma página nova, não precisemos incorporar esses três arquivos lá no head, podemos simplesmente importar um arquivo que os outros dois já vão vir juntos no import.

[01:01] O que vimos também é como cada componente funciona, vimos o que faz um elemento ser um componente e separamos cada um desses componentes em arquivos CSS separados, e nós depois só colocamos aqui em pastas separadas.

[01:18] Com isso também vimos um pouco sobre arquitetura CSS, o que cada pasta representa, o que cada arquivo representa. Seções ficam separadas da pasta de componentes e assim por diante.

[01:31] E no processo desenvolvimento da página também vimos outras ferramentas que auxiliaram muito o desenvolvimento, que foi o Emmet e o IntelliSense CSS.

[01:40] E com todo esse conjunto nós conseguimos montar completamente a página home mobile aqui da Casa Verde. Montamos os cabeçalhos, as seções, cada um dos componentes que usamos nessa página e assim por diante, como também boas práticas e técnicas que podemos usar no dia a dia.

[02:03] Agora com o curso finalizado eu recomendo que você faça os exercícios, que ajudam bastante no momento de fixar a informação do que aprendemos no curso. Também, caso você tenha dúvidas, caso você queira participar de uma comunidade, você queira pedir ajuda ou ajudar outras pessoas, você pode acessar o Fórum da Caelum.

[02:22] E também recomendo a você pegar todo esse aprendizado que nós tivemos no curso e tentar fazer a segunda página, que é o detalhe do produto mobile. Usar tudo o que nós aprendemos fazendo a primeira página, aplicar nessa segunda página. Muito obrigado por ter feito esse curso comigo e até breve.