

Vue





TechGuide - Alura, FIAP e PM3

Nível 1





HTML - Fundamentos:

- HTML é uma linguagem de marcação que define a estrutura do seu conteúdo. HTML consiste em uma série de elementos que você usa para mostrar algo de uma determinada maneira ou agir de uma certo modo. As tags podem criar um hiperlink de uma palavra ou imagem para outro lugar, podem colocar palavras em itálico, podem aumentar ou diminuir a fonte e assim por diante.
- Aprender quais tags são necessárias para um HTML básico
- Criar um parágrafo de texto
- Exibir uma imagem
- Conhecer a diferença entre 'h1', 'h2', 'h3', etc
- Criar um texto com hyperlink
- Criar um formulário com campos relevantes
- Criar uma lista de itens ordenada ou não ordenada
- Criar uma lista de itens dentro de uma lista suspensa (dropdown list)
- Conectar com um arquivo de CSS
- Criar uma tabela
- Adicionar IDs e classes

Conteúdos

-  MDN Web Docs: HTML básico (https://developer.mozilla.org/pt-BR/docs/Learn/Getting_started_with_the_web/HTML_basics)
-  Rafaella Ballerini: 5 MINUTOS DE HTML PARA INICIAR EM PROGRAMAÇÃO! (<https://www.youtube.com/watch?v=3oSlqlqzN3M>)
-  Rafaella Ballerini: LANDING PAGE COM HTML e CSS! (<https://www.youtube.com/watch?v=llF6vD-RlJE>)
-  Rafaella Ballerini: FORMULÁRIOS COM HTML e CSS! (<https://www.youtube.com/watch?v=wwqOJ2o84S4>)

Conteúdos Alura:**

-  HTML: o que é, a importância para a Web, como aprender e um Guia para iniciantes (<https://www.alura.com.br/artigos/html>)
-  Alura+: Fazendo deploy de um projeto na Vercel (<https://cursos.alura.com.br/extra/alura-mais/fazendo-deploy-de-um-projeto-na-vercel-c9239>)
-  HTML, CSS e Javascript, quais as diferenças? (<https://www.alura.com.br/artigos/html-css-e-js-definicoes>)
-  Formação HTML e CSS (<https://www.alura.com.br/formacao-html-e-css>)

- **Curso** Apostila: Desenvolvimento Web com HTML, CSS e JavaScript (<https://www.alura.com.br/apostila-html-css-javascript>)
- **Desafio** 7 Days of Code: HTML e CSS (<https://7daysofcode.io/matricula/html-css>)

CSS - Fundamentos:

- Cascading Style Sheets (CSS) é uma linguagem usada para descrever a apresentação de um documento escrito em uma linguagem de marcação como HTML ou XML. CSS pode ser usado para estilos de texto de documentos muito básicos — por exemplo, para alterar a cor e o tamanho de títulos e links. Ele pode ser usado para criar um layout — por exemplo, transformar uma única coluna de texto em um layout com uma área de conteúdo principal e uma barra lateral para informações relacionadas. Pode até ser usado para efeitos como animações.
- Aprender a estrutura visual de uma página, com 'margin' e 'padding'
- Estabelecer o tamanho com 'width' e 'height'
- Aprender sobre a posição de um elemento ('static', 'relative' ou 'absolute')
- Aprender sobre o 'display' de exibição de um elemento ('block', 'inline', 'inline-block')
- Aprender a posicionar imagens em relação ao texto
- Aprender sobre alinhamento
- Aprender sobre estilo de fontes
- Aprender as diferenças e vantagens de usar as diferentes unidades de medida em CSS (% , relativas, etc)
- Conectar com os elementos (IDs, classes) de um arquivo HTML
- Alterar características de um elemento quando o mouse passar por cima dele ('hover')
- Aprender box-sizing
- Aprender Flexbox
- Aprender Grid

Conteúdos

- **Site** MDN Web Docs: Iniciando com CSS (https://developer.mozilla.org/pt-BR/docs/Learn/CSS/First_steps/Getting_started)
- **Site** MDN Web Docs: Seletores (https://developer.mozilla.org/pt-BR/docs/Learn/CSS/Building_blocks/Selectors)
- **Site** MDN Web Docs: Cascata e herança (https://developer.mozilla.org/pt-BR/docs/Learn/CSS/Building_blocks/Cascade_and_inheritance)
- **Site** MDN Web Docs: Propriedade 'display' (<https://developer.mozilla.org/pt-BR/docs/Web/CSS/display>)
- **Site** MDN Web Docs: Conceitos básicos de Flexbox (https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox)
- **Site** MDN Web Docs: Grid Layout (https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Grid_Layout/Basic_Concepts_of_Grid_Layout)
- **Artigo** Entendendo CSS: Display Inline, Block e Inline-Block (<https://dev.to/sucodelarangela/entendendo-css-display-inline-block-e-inline-block-lic>)
- **Artigo** Entendendo CSS: Pseudo-Classes e Pseudo-Elementos (<https://dev.to/sucodelarangela/entendendo-css-pseudo-classes-e-pseudo-elementos-b83>)
- **Artigo** Como aplicar opacidade em background-image sem afetar textos (<https://dev.to/sucodelarangela/como-aplicar-opacidade-em-background-image-sem-afetar-textos-31fj>)

- **YouTube** Rafaella Ballerini: O QUE É CSS? (<https://www.youtube.com/watch?v=LWU2OR19ZG4>)
- **YouTube** Rafaella Ballerini: FLEXBOX CSS! Como posicionar elementos na página web - parte 1 (https://www.youtube.com/watch?v=KbjLtEgmZ_E)
- **YouTube** Rafaella Ballerini: FLEXBOX CSS! Como posicionar elementos na página web - parte 2 (https://www.youtube.com/watch?v=hjz6ezV9_uc)
- **YouTube** Matheus Castiglioni: Arquitetura CSS - BEM (<https://www.youtube.com/watch?v=yKPXW9aSxQI>)
- **YouTube** Matheus Castiglioni: CSS Grid Layout (<https://www.youtube.com/watch?v=HBIBNAtFcdw>)
- **YouTube** Mario Souto - Dev Soutinho: Como centralizar no CSS (<https://www.youtube.com/watch?v=Cu-HP-gvgggg>)

Conteúdos Alura:**

- **Podcast** Hipster 09 - CSS: Cansei de Ser Simples (<https://www.hipsters.tech/css-cansei-de-ser-simples-hipsters-09/>)
- **Artigo** CSS: o que é, como usar no HTML e um Guia para iniciar (<https://www.alura.com.br/artigos/css>)
- **Artigo** HTML, CSS e Javascript, quais as diferenças? (<https://www.alura.com.br/artigos/html-css-e-js-definicoes>)
- **Artigo** Guia de unidades no CSS (<https://www.alura.com.br/artigos/guia-de-unidades-no-css>)
- **Artigo** CSS: Guia do Flexbox (<https://www.alura.com.br/artigos/css-guia-do-flexbox>)
- **Artigo** Como fazer Grids e a Responsividade na Web (<https://www.alura.com.br/artigos/como-fazer-grids-e-a-responsividade-na-web>)
- **Artigo** Como criar animações incríveis com CSS (<https://www.alura.com.br/artigos/animacoes-com-css>)
- **Artigo** Houdini CSS: um jeito mágico de criar estilos personalizados (<https://www.alura.com.br/artigos/houdini-css>)
- **YouTube** Alura: CSS FlexBox: Dicas para começar (<https://www.youtube.com/watch?v=326-B1avuYo>)
- **YouTube** Alura: Box Model e Sizing no CSS (<https://www.youtube.com/watch?v=qcNUxyOWXlw>)
- **YouTube** Alura: Curso de HTML5 e CSS3 (<https://www.youtube.com/watch?v=78AyiuxYceg>)
- **Curso** Apostila: Desenvolvimento Web com HTML, CSS e JavaScript (<https://www.alura.com.br/apostila-html-css-javascript>)
- **Curso** Formação HTML e CSS (<https://www.alura.com.br/formacao-html-e-css>)
- **Curso** Formação CSS: aprofunde em estilos (<https://cursos.alura.com.br/formacao-css-estilos>)
- **Curso** HTML e CSS: responsividade com mobile-first (<https://www.alura.com.br/curso-online-html-css-responsividade-mobile-first>)
- **Curso** CSS: cursos para transformar designs com Grid, Flexbox e Sass (<https://cursos.alura.com.br/formacao-css-cursos-transformar-designs-grid-flexbox-sass>)
- **Desafio** 7 Days of Code: HTML e CSS (<https://7daysofcode.io/matricula/html-css>)
- **Desafio** 7 Days of Code: Responsividade (<https://7daysofcode.io/matricula/responsividade>)

JavaScript - Fundamentos:

- JavaScript é a linguagem de programação mais popular do mundo e é uma das principais tecnologias da World Wide Web, juntamente com HTML e CSS. Ela possui tipagem dinâmica,

orientação a objetos baseada em protótipos e funções de primeira classe. Ela é multi-paradigma e suporta estilos de programação orientados a eventos, funcionais e imperativos.

- Conhecer os tipos primitivos
- Declarar variáveis, considerando a diferença entre 'var', 'let' e 'const'
- Usar estruturas condicionais ('if', 'else')
- Conhecer os operadores de atribuição e comparação ('=', '==', '===')
- Usar estruturas de repetição e laços ('while', 'for')
- Usar funções, passando parâmetros e argumentos
- Manipular arrays e listas
- Aprender o conceito de Orientação a Objetos
- Fazer um CRUD
- Obter dados de uma API
- Fazer chamadas assíncronas usando 'Async/Await', 'Promise', etc

Conteúdos

- **Site** MDN Web Docs: Um primeiro mergulho no JavaScript (https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps)
- **Site** MDN Web Docs: Trabalhando com texto — strings em JavaScript (https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/Strings)
- **Site** MDN Web Docs: if...else (<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/if...else>)
- **Site** MDN Web Docs: while (<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/while>)
- **Site** MDN Web Docs: for (<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/for>)
- **Site** MDN Web Docs: Arrays (https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/Arrays)
- **Site** MDN Web Docs: Template strings (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Template_literals)
- **Artigo** Definindo Funções em Javascript (<https://blog.matheuscastiglioni.com.br/definindo-funcoes-em-javascript/>)
- **YouTube** Mario Souto - Dev Soutinho: Como manipular arrays e objetos em JavaScript (<https://www.youtube.com/watch?v=yS7AcF-xRUg>)
- **YouTube** Marco Bruno: O que é Json e como criar um objeto (<https://www.youtube.com/watch?v=oCY5YEEjlwE>)
- **YouTube** Mario Souto - Dev Soutinho: Como usar Async/Await? Promises no JavaScript? (<https://www.youtube.com/watch?v=q28IfkBd9F4>)
- **YouTube** Mario Souto - Dev Soutinho: Como pegar dados de uma API? Como fazer AJAX? | Pegando dados de serviços via JavaScript (<https://www.youtube.com/watch?v=85vJXFpXLQw>)

Conteúdos Alura:**

- **Artigo** JavaScript: o que é, como aprender e um Guia da linguagem mais popular do mundo (<https://www.alura.com.br/artigos/javascript/>)
- **Podcast** Hipster 38 - O Reino encantado do JavaScript (<https://www.hipsters.tech/o-reino-encantado-do-javascript-hipsters-38/>)

- **Podcast** Hipster 169 - JavaScript: manual de sobrevivência 2020 (<https://www.hipsters.tech/javascript-manual-de-sobrevivencia-2020-hipsters-169/>)
- **Podcast** Hipster 236 - Evolução do JavaScript (<https://www.hipsters.tech/evolucao-do-javascript-hipsters-ponto-tech-236/>)
- **Artigo** Strings com JavaScript: o que são e como manipulá-las (<https://www.alura.com.br/artigos/strings-com-javascript-o-que-sao-e-como-manipular>)
- **Artigo** Como utilizar operadores de comparação em Javascript (<https://www.alura.com.br/artigos/operadores-matematicos-em-javascript>)
- **Artigo** Para que serve um Array? (<https://www.alura.com.br/artigos/javascript-para-que-serve-array>)
- **Artigo** Quando usar forEach e map? (<https://www.alura.com.br/artigos/javascript-quando-devo-usar-foreach-e-map>)
- **Artigo** Manipulação de array com map, filter e reduce (<https://www.alura.com.br/artigos/manipulacao-de-array-com-map-filter-e-reduce>)
- **Curso** Formação Desenvolva aplicações Web com JavaScript (<https://cursos.alura.com.br/formacao-javascript-front-end>)
- **YouTube** Alura: O que é JavaScript? (<https://www.youtube.com/watch?v=NaVSbnnV75Q>)
- **YouTube** Alura: Destructuring em JavaScript (<https://www.youtube.com/watch?v=f8a-qwKC5yk>)
- **YouTube** Alura: Classes x funções no Javascript (<https://www.youtube.com/watch?v=i0hhj-k9L6s>)
- **Desafio** 7 Days of Code: JavaScript e DOM (<https://7daysofcode.io/matricula/javascript-e-dom>)
- **Desafio** 7 Days of Code: JavaScript e DOM com API (<https://7daysofcode.io/matricula/javascript-e-dom-api>)
- **Desafio** 7 Days of Code: JavaScript e DOM - Wordle (<https://7daysofcode.io/matricula/javascript-e-dom-wordle>)

DOM - Fundamentos:

- O Document Object Model (DOM) é uma interface de programação para documentos web. Ele representa a página para que os programas possam alterar a estrutura, o estilo e o conteúdo do documento. O DOM representa o documento como nós e objetos; dessa forma, linguagens de programação podem interagir com a página.
- Entender como funciona a árvore do DOM
- Acessar e manipular elementos do HTML e CSS
- Acessar os pais e filhos de um elemento
- Inserir um novo elemento na árvore
- Remover um elemento da árvore
- Esperar por um evento em certo elemento da página usando 'addEventListener()'

Conteúdos

- **Site** MDN Web Docs: Introdução ao DOM (https://developer.mozilla.org/pt-BR/docs/Web/API/Document_Object_Model/Introduction)
- **Site** MDN Web Docs: querySelector() (<https://developer.mozilla.org/pt-BR/docs/Web/API/Document/querySelector>)
- **Site** MDN Web Docs: addEventListener() (<https://developer.mozilla.org/pt-BR/docs/Web/API/EventTarget/addEventListener>)

- **YouTube** Felipe Aguiar - Laboratório de javascript: Javascript DOM 01 - DOM - A base da manipulação e criação de páginas dinâmicas (<https://www.youtube.com/watch?v=G31EuXNMDPk>)

Conteúdos Alura:**

- **Curso** Curso JavaScript: manipulando o DOM (<https://cursos.alura.com.br/course/javascript-manipulando-elementos-dom>)
- **Artigo** Guia para manipular eventos no DOM com JavaScript: técnicas e exemplos práticos (<https://www.alura.com.br/artigos/eventos-do-dom>)
- **Desafio** 7 Days of Code: JavaScript e DOM (<https://7daysofcode.io/matricula/javascript-e-dom>)
- **Desafio** 7 Days of Code: JavaScript e DOM com API (<https://7daysofcode.io/matricula/javascript-e-dom-api>)
- **Desafio** 7 Days of Code: JavaScript e DOM - Wordle (<https://7daysofcode.io/matricula/javascript-e-dom-wordle>)

Conceitos SPA:

- Um aplicativo de página única (SPA - single-page application) é uma aplicação web ou site que interage com o usuário reescrevendo dinamicamente a página web atual com novos dados do servidor web, em vez do método padrão de um navegador que carrega novas páginas inteiras. O objetivo são transições mais rápidas, que tornam o site mais parecido com uma aplicação nativa.
- Entender o que é uma SPA
- Estabelecer rotas para outras páginas
- Conhecer frameworks SPA
- Comunicação com APIs

Conteúdos

- **Site** Wikipédia: Aplicativo de página única (SPA) (https://pt.wikipedia.org/wiki/Aplicativo_de_p%C3%A1gina_%C3%BAnica)
- **Site** MDN Web Docs: SPA (Single-page application) (inglês) (<https://developer.mozilla.org/en-US/docs/Glossary/SPA>)
- **Artigo** What I wish I had known about SPAs (inglês) (<https://stackoverflow.blog/2021/12/28/what-i-wish-i-had-known-about-single-page-applications/>)

Conteúdos Alura:**

- **Podcast** Hipsters.tech: Single Page Applications (<https://www.hipsters.tech/single-page-applications-hipsters-16/>)
- **YouTube** Alura: O que é uma Single-Page Application? (SPA) (<https://www.youtube.com/watch?v=opxYpCKzILk>)
- **YouTube** Alura: Como configurar o React Router para sua SPA (<https://www.youtube.com/watch?v=4Tb8dp5GYql>)
- **Curso** Imersão React: AluraCord - Aula 02 (<https://cursos.alura.com.br/imersoes/aulas/aula-02-state-novas-paginas-e-navegacao-spa-vs-a-tradicional-c69>)
- **Site** Primeiras aulas do curso React Router: navegação em uma SPA (<https://www.alura.com.br/conteudo/react-router-navegacao-spa>)
- **Curso** Curso React: desenvolvendo em React Router com JavaScript (<https://cursos.alura.com.br/course/React-desenvolvendo-react-router-javascript>)

- **Curso** Curso React: conhecendo a biblioteca React Router (<https://www.alura.com.br/curso-online-react-biblioteca-react-router>)

Vue - Componentes:

- O Vue permite definir componentes usando a Option API ou a Composition API.
- Para que servem e como funcionam componentes
- Conheça a especificação da Sintaxe SFC (Single File Component)

Conteúdos

- **YouTube** Vue.js (O competidor de peso do Angular e React) // Dicionário do Programador (<https://www.youtube.com/watch?v=bEl6yN3vd-U&t=161s&pp=ygUOYXByZW5kYSB2dWVqcyA%3D>)
- **Site** Vue: Documentação - Registrando Componentes (<https://vuejs.org/guide/components/registration.html>)
- **Site** Vue: Documentação - Single-File Components (<https://vuejs.org/guide/scaling-up/sfc.html>)

Conteúdos Alura:**

- **Artigo** Angular vs React vs Vue.js (<https://www.alura.com.br/artigos/angular-vs-react-vs-vue-js>)
- **Artigo** Vue.js: o que é, como funciona e como começar a usar esse framework JS (<https://www.alura.com.br/artigos/vue-js>)
- **Artigo** Vue 3: Conhecendo mais de perto (<https://www.alura.com.br/artigos/vue-3-conhecendo-mais-de-perto>)
- **Podcast** Vue.js - Hipsters Ponto Tech #288 (<https://www.alura.com.br/podcast/vue-js-hipsters-ponto-tech-288-a1387>)
- **Site** Preparando o ambiente com Vue!" (<https://cursos.alura.com.br/extra/alura-mais/preparando-o-ambiente-com-vue--c1359>)
- **Curso** Vue3: explorando o framework (<https://cursos.alura.com.br/course/vue3-comecando-framework>)

Vue - Props:

- Props são um objeto que é injetado em componentes e fornece alguns dados que podem ser compartilhados entre outros componentes em um fluxo de dados unidirecional, de um elemento pai para um elemento filho. Props são de somente leitura.
- Como passar props
- Como manipular props

Conteúdos

- **Site** Vue: Documentação - Props (<https://vuejs.org/guide/components/props.html>)
- **Site** Introdução à CLI do Vue e a componentes de arquivo único no Vue.js (<https://learn.microsoft.com/pt-br/training/modules/vue-cli-components/>)
- **Artigo** Domine as Props no Vue.js: Da Declaração à Validação (<https://marcosviniciosneves.medium.com/domine-as-props-no-vue-js-da-declara%C3%A7%C3%A3o-%C3%A0-valida%C3%A7%C3%A3o-cd44952cde23>)

Conteúdos Alura:**

- **Podcast** Vue.js na Wirecard (<https://www.alura.com.br/podcast/vue-js-na-wirecard-hipsters-on-the-road-09-a427>)

Vue - Diretivas:

- Explore as diretivas básicas oferecidas pelo Vue, como v-if, v-else, v-show, v-bind, v-model e v-for.
- Depois de se familiarizar com as diretivas integradas, suba de nível criando suas próprias diretivas personalizadas.
- Para finalizar sua jornada, domine os argumentos e modificadores das diretivas.

Conteúdos

- **Site** Vue: Documentação - Renderização condicional (<https://vuejs.org/guide/essentials/conditional.html>)
- **Site** Vue: Documentação - Renderização de listas (<https://vuejs.org/guide/essentials/list.html>)
- **Site** Vue: Documentação - Diretivas customizadas (<https://vuejs.org/guide/reusability/custom-directives.html#custom-directives>)
- **Site** Exibições de página dinâmicas com o Vue.js (<https://learn.microsoft.com/pt-br/training/modules/vue-dynamic-rendering/>)

Vue - Options API:

- Aprenda a estruturar um componente Vue usando a Options API, abrangendo opções como 'data', 'methods', 'computed', e 'watch'.
- Domine os hooks do ciclo de vida do componente ('created', 'mounted', 'updated', e 'destroyed') para controlar comportamentos em diferentes etapas da vida do componente.
- Pratique o gerenciamento de estado reativo dentro dos componentes, utilizando opções como 'data' e 'computed', e aprenda como sincronizar estados entre componentes usando 'props' e 'emit'.

Conteúdos

- **Site** Vue: Documentação - O básico de um componente (<https://vuejs.org/guide/essentials/component-basics.html>)
- **Site** Mozilla: Usando propriedades computadas do Vue (https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/Vue_computed_properties)
- **Artigo** Vue3: Options API vs Composition API (<https://dev.to/sucodelarangela/vue3-options-api-vs-composition-api-1j09>)
- **Artigo** Vue3: Como declarar estados reativos (<https://dev.to/sucodelarangela/vue3-como-declarar-estados-reativos-1lc7>)

Conteúdos Alura:**

- **Artigo** VueJS: Ciclo de vida dos componentes (<https://www.alura.com.br/artigos/vuejs-ciclo-vida-componentes>)

Vue - Composition API:

- Inicie componentes com a função setup(). Ela é o coração da Composition API.
- Aprenda a usar ref e reactive para criar e gerenciar estados reativos.
- Domine os hooks do ciclo de vida. Eles são vitais para controlar comportamentos em diferentes estágios do componente.

Conteúdos

- **Site** Vue: Documentação - O básico de um componente (<https://vuejs.org/guide/essentials/component-basics.html>)
- **Site** Vue: Documentação - Ref e reatividade (<https://vuejs.org/guide/essentials/reactivity-fundamentals.html#ref>)
- **Site** Vue: Documentação - Ciclo de vida usando hooks (<https://vuejs.org/guide/essentials/lifecycle.html>)
- **Site** Vue: Documentação - Propriedades computadas (<https://vuejs.org/guide/essentials/computed.html>)
- **Artigo** Vue3: Options API vs Composition API (<https://dev.to/sucodelarangela/vue3-options-api-vs-composition-api-1j09>)
- **Artigo** Vue3: Como declarar estados reativos (<https://dev.to/sucodelarangela/vue3-como-declarar-estados-reativos-1lc7>)

Conteúdos Alura:**

- **Artigo** VueJS: Ciclo de vida dos componentes (<https://www.alura.com.br/artigos/vuejs-ciclo-vida-componentes>)
- **Curso** Vue3: composition API e Vuex (<https://www.alura.com.br/curso-online-vue3-composition-api-vuex>)

Vue - Template:

- Comece aprendendo a sintaxe básica de um template Vue e como renderizar dados dinâmicos.
- Aprofunde-se no uso de diretivas de template como v-if, v-for e v-bind para criar interfaces dinâmicas.
- Aprenda a definir e usar componentes dentro dos seus templates para reutilizar e organizar o código.

Conteúdos

- **Site** Vue: Documentação - Template (<https://vuejs.org/guide/essentials/template-syntax.html>)

Conteúdos Alura:**

- **Curso** Vue3: avançando no framework (<https://www.alura.com.br/curso-online-vue3-avancando-framework>)

Vue - Create App:

- Inicie criando um novo projeto Vue com Vue CLI ou Vite.
- Crie componentes Vue e entenda como estruturá-los usando Options API ou Composition API.
- Veja como a navegação com Vue Router e gerenciamento de estado com Pinea podem ser úteis em SPAs escritas em Vue.

Conteúdos

- **Site** Vue: Playground - Experimente Vue (<https://play.vuejs.org/#eNp9kVFLwzAQx7/KeS9TmBuyt1EHKgP1QUUFX/JS2lvXmSYhucxC6Xf32tLqwxIEJPf/X!>)
- **Site** Vue: Documentação - Criando uma aplicação (<https://vuejs.org/guide/quick-start.html#creating-a-vue-application>)
- **Site** Vue: Documentação - Ciclo de vida da aplicação (<https://vuejs.org/guide/essentials/application.html>)

Vue Forms:

- Comece aprendendo as bases dos formulários no Vue, incluindo v-model e como vincular diferentes tipos de campos de entrada (input binds).
- Depois de dominar as bases, aprenda como o Vue 3 lida com as entradas do usuário e as diferenças em relação às versões anteriores.
- Mergulhe no mundo da Vue Formulate, uma das bibliotecas mais completas para trabalhar com formulários no Vue. Compreenda como usar seus recursos para construir formulários complexos com facilidade.
- Continue sua jornada com a VeeValidate, uma biblioteca para validação de formulários que oferece muitas opções de validação e permite que você crie suas próprias regras.
- Explore a Vuelidate, outra biblioteca de validação poderosa que oferece uma abordagem um pouco diferente da VeeValidate.
- Além de usar bibliotecas, aprenda a criar suas próprias regras de validação personalizadas no Vue.
- Aprofunde-se na criação de formulários dinâmicos, onde os campos podem mudar com base nas ações do usuário.
- Descubra como integrar a gestão dos formulários com Vuex/Pinia para um gerenciamento de estado mais sólido e escalável.

Conteúdos

- **Site** Vue: Documentação - Inputs (<https://vuejs.org/guide/essentials/forms.html>)
- **Site** Vue Formulate (<https://vueformulate.com/>)
- **Site** Vee Validate (<https://vee-validate.logaretm.com/v4/>)
- **Site** Vuex (<https://vuex.vuejs.org/>)
- **Site** Pinia (<https://pinia.vuejs.org/>)

Vue - Eventos:

- Inicie seu aprendizado sobre eventos no Vue, entendendo a diferença entre eventos do DOM e eventos personalizados do Vue.
- Aprenda a usar a diretiva v-on para adicionar manipuladores de eventos a elementos do DOM e como passar argumentos para os manipuladores de eventos.
- Mergulhe nos eventos de teclado e mouse, entendendo como capturá-los e como usar modificadores de eventos para simplificar o código.
- Descubra como emitir e ouvir eventos personalizados usando o método \$emit e a diretiva v-on.
- Aprofunde-se na compreensão do bubbling e capturing de eventos no Vue, e quando usar o .native modifier.
- Conheça o conceito de "Event Bus" para comunicação entre componentes não relacionados.

Conteúdos

- **Site** Vue: Documentação - Manipulando eventos (<https://vuejs.org/guide/essentials/event-handling.html>)
- **Site** Eventos em apps Vue (https://www.w3schools.com/vue/vue_events.php)

Vue Router:

- Comece instalando e configurando o Vue Router em seu projeto. Aprenda a criar um arquivo de rotas e a configurar o Vue para usar o Vue Router.

- Domine a criação de rotas básicas, entenda a correspondência de rotas com os componentes Vue e como usar a diretiva para criar links.
- Avance para o roteamento dinâmico, aprendendo a usar parâmetros de rota para criar rotas flexíveis.
- Entenda o roteamento aninhado e como usar o componente para renderizar componentes de rota aninhados.
- Mergulhe no roteamento nomeado e aprenda como ele pode tornar o seu código mais fácil de entender e manter.
- Explore os guardas de rota e como eles podem ser usados para controlar o acesso a certas rotas.
- Aprenda a criar transições suaves entre as rotas com o elemento do Vue.

Conteúdos

- **Site** Vue Router: Documentação (<https://router.vuejs.org/>)

Conteúdos Alura:**

- **Curso** Vue3: explorando o framework (<https://cursos.alura.com.br/course/vue3-comecando-framework>)

Nível 2

Vue - Composition API a fundo:

- Comece entendendo a função setup() e o conceito de reatividade no Vue 3.
- Domine ref e reactive, os principais elementos para gerenciar estados reativos.
- Aprenda a usar computed para trabalhar com propriedades calculadas.
- Explore o uso de watch e watchEffect para reagir às mudanças de estado.
- Entenda os hooks do ciclo de vida para controlar o comportamento do componente em diferentes fases.
- Crie e exponha funções a partir da função setup.
- Aprenda a usar props dentro da setup e como lidar com as diretivas no template.
- Implemente a lógica reutilizável através de Composables.
- Aprenda as melhores práticas e padrões comuns ao usar a Composition API.

Conteúdos

- **Site** Vue: Documentação - Options VS Composition API (<https://vuejs.org/guide/introduction.html#which-to-choose>)
- **Site** Vue: Documentação - Tutoriais práticos para se aprofundar na Composition API (<https://vuejs.org/tutorial/#step-1>)
- **Site** Vue Composition API (https://www.w3schools.com/vue/vue_composition-api.php)

Conteúdos Alura:**

- **Curso** Vue3: composition API e Vuex (<https://www.alura.com.br/curso-online-vue3-composition-api-vuex>)

Vue Devtools:

- Comece instalando a extensão Vue DevTools no seu navegador.

- Familiarize-se com a interface, aprendendo sobre as abas de Componentes, Vuex, Roteamento e Performance.
- Aprenda a inspecionar componentes, observando seus props, dados, e slots.
- Entenda como monitorar a store do Vuex | Pinia e o histórico de roteamento.
- Pratique a edição no local do estado da aplicação e o "time-travel debugging".
- Ajuste as configurações ao seu gosto e integre as DevTools com seu ambiente de desenvolvimento.

Conteúdos

- [Site](https://chrome.google.com/webstore/detail/vuejs-devtools/nhdogjmejiglipccpnnnanhbledajbpd) Vue.js devtools (<https://chrome.google.com/webstore/detail/vuejs-devtools/nhdogjmejiglipccpnnnanhbledajbpd>)

Vue - Estilizando componentes:

- Comece pelo básico, aplicando folhas de estilo CSS externas ao seu projeto Vue.
- Vá um pouco mais fundo e aprenda sobre o atributo "scoped" no Vue, que permite que você direcione estilos especificamente para um único componente.
- Explore bibliotecas que permitem que você escreva estilos em JavaScript, oferecendo recursos dinâmicos e reutilizáveis.
- Veja como os pré-processadores CSS, como SASS ou LESS, podem ser integrados ao seu projeto Vue para estilização avançada.
- Experimente bibliotecas populares de componentes Vue, como Vuetify ou BootstrapVue, que vêm com seus próprios conjuntos de estilos e temas.
- Conheça a biblioteca vue-styled-components, uma implementação do Styled Components para Vue, que permite escrever CSS real em JavaScript.
- Dê uma olhada no Tailwind CSS, um framework de CSS de utilidade, e aprenda como pode ser integrado com Vue para um desenvolvimento de interface de usuário eficiente.
- Com o Vue 3, explore a vinculação de estilos e como eles podem ser usados para aplicar estilos dinâmicos com base nos dados do componente.

Conteúdos

- [Site](https://vuejs.org/guide/essentials/class-and-style.html) Vue: Documentação - Classes e estilos (<https://vuejs.org/guide/essentials/class-and-style.html>)
- [Site](https://vuejs.org/api/sfc-css-features.html#scoped-css) Vue: Documentação - Escopo do CSS (<https://vuejs.org/api/sfc-css-features.html#scoped-css>)
- [Site](https://vuejs.org/api/sfc-css-features.html#css-modules) Vue: Documentação - Módulos CSS (<https://vuejs.org/api/sfc-css-features.html#css-modules>)
- [Site](https://vuejs.org/api/sfc-spec.html#pre-processors) Vue: Documentação - Pré Processadores (<https://vuejs.org/api/sfc-spec.html#pre-processors>)
- [Site](https://bootstrap-vue.org/) Bootstrap Vue (<https://bootstrap-vue.org/>)
- [Site](https://tailwindcss.com/docs/guides/vite#vue) Tailwind CSS (<https://tailwindcss.com/docs/guides/vite#vue>)

Conteúdos Alura:**

- [Site](https://www.alura.com.br/conteudo/tailwind-css-estilizando-pagina-classes-utilitarias) Tailwind CSS: estilizando a sua página com classes utilitárias (<https://www.alura.com.br/conteudo/tailwind-css-estilizando-pagina-classes-utilitarias>)

Vue - Comunicando-se com APIs:

- Entenda o que são APIs, como funcionam e como se comunicam com os clientes web.

- Aprenda os conceitos básicos de AJAX e como o Vue se integra com AJAX para recuperar dados de uma API.
- Mergulhe no Axios, uma popular biblioteca JavaScript para realizar solicitações HTTP. Aprenda a instalar e a usar o Axios em um projeto Vue para fazer chamadas GET, POST e outras.
- Conheça o unfetch, uma alternativa leve e moderna para fazer chamadas de API. Veja como ele pode ser usado em um projeto Vue.
- Adentre no mundo do GraphQL, um poderoso paradigma de API. Aprenda a usar o Apollo Client, uma solução completa para o gerenciamento de estado de dados GraphQL em aplicações Vue.
- Explore o Vue Relay, uma estrutura para trabalhar com GraphQL no Vue. Entenda as vantagens que ele oferece e como ele difere do Apollo Client.
- Aprenda a lidar com as respostas das APIs, incluindo a manipulação de dados recebidos e a tratamento de erros.

Conteúdos

- **Site** Axios (<https://axios-http.com/docs/intro>)
- **Site** Unfetch (<https://github.com/developit/unfetch>)
- **Site** Vue Apollo (<https://apollo.vuejs.org/guide/>)
- **Site** Vue Relay (<https://github.com/vue-relay/vue-relay>)

Conteúdos Alura:**

- **Artigo** REST: Conceito e fundamentos (<https://www.alura.com.br/artigos/rest-conceito-e-fundamentos>)
- **Artigo** Guia essencial do GraphQL para pessoas que desenvolvem frontend (<https://www.alura.com.br/artigos/guia-graphql-pessoas-que-desenvolvem-frontend>)

JavaScript - Callbacks e Promises:

- Uma promessa (Promise) é um proxy para um valor não necessariamente conhecido quando a promessa é criada. Isso permite que métodos assíncronos retornem valores como métodos síncronos - em vez de retornar imediatamente o valor final, o método assíncrono retorna uma promessa de fornecer o valor em algum momento no futuro.
- Uma função de Callback é uma função passada para outra função como um argumento, que é então invocado dentro da função externa para completar algum tipo de rotina ou ação.
- Uma função assíncrona (async) é uma função declarada com a palavra-chave `async`, e a palavra-chave `await` é permitida dentro dela. As palavras-chave `async` e `await` permitem que o comportamento assíncrono e baseado em promessas seja escrito em um estilo mais limpo, evitando a necessidade de configurar explicitamente as cadeias de promessas.
- Entender o conceito de assincronicidade em programação
- Escrever código assíncrono entendendo o conceito de promessas em JavaScript
- Utilizar os métodos, palavras-chaves e objetos do JavaScript para manipulação de promessas como 'Async/Await', '.then()', 'Promise', etc
- Aprender em quais situações é necessário o uso de programação assíncrona
- Fazer chamadas em APIs com `fetch()`

Conteúdos

- **Site** MDN Web Docs: Introdução ao JavaScript Async (<https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Asynchronous/Introducing>)
- **Site** MDN Web Docs: Funções assíncronas (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/async_function)

- **Site** MDN Web Docs: Promise (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Promise)
- **Site** MDN Web Docs: Função Callback (https://developer.mozilla.org/pt-BR/docs/Glossary/Callback_function)
- **YouTube** Mario Souto - Dev Soutinho: Como usar Async/Await? Promises no JavaScript? (<https://www.youtube.com/watch?v=g28IfkBd9F4>)
- **YouTube** Mario Souto - Dev Soutinho: Callbacks, Síncrono, Assíncrono e Event Loop no JavaScript (<https://www.youtube.com/watch?v=6lbBaM18X3g>)
- **Artigo** Entendendo como fazer AJAX com a FetchAPI (<https://medium.com/@omariosouto/entendendo-como-fazer-ajax-com-a-fetchapi-977ff20da3c6>)
- **YouTube** Mario Souto - Dev Soutinho: Como pegar dados de uma API? Como fazer AJAX? | Pegando dados de serviços via JavaScript (<https://www.youtube.com/watch?v=85vJXFpXLQw>)
- **Artigo** Revolução no Node.js: adeus ao Axios e fetch API na versão 17.5.0 (<https://www.alura.com.br/artigos/revolucao-node-js-adeus-axios-fetch-api-versao-17-5-0>)

Conteúdos Alura:**

- **Artigo** Async/await no JavaScript: o que é e quando usar a programação assíncrona? (<https://www.alura.com.br/artigos/async-await-no-javascript-o-que-e-e-quando-usar>)
- **Site** Alura+: JavaScript assíncrono e Fetch (<https://cursos.alura.com.br/extra/alura-mais/javascript-assincrono-e-fetch-c93>)
- **Curso** Curso JavaScript: consumindo e tratando dados de uma API (<https://cursos.alura.com.br/course/javascript-consumindo-tratando-dados-uma-api>)

JavaScript - Manipulação de Erros:

- O tratamento de erros refere-se aos procedimentos de resposta e recuperação de condições de erro presentes em um aplicativo de software. Em outras palavras, é o processo composto de antecipação, detecção e resolução de erros de aplicação, de programação ou de comunicação.
- Conhecer e tratar as exceções mais comuns
- Saber quais os tipos de erros e em quais situações eles podem ocorrer
- Entender como o Node.js faz o manejo de erros
- Usar 'try' e 'catch' para tratamento de erros
- Em que ocasiões e de que forma utilizar o **throw**
- Criar exceções específicas de acordo com a necessidade de sua aplicação

Conteúdos

- **Site** MDN Web Docs: Controle de Fluxo e Manipulação de Erro (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Control_flow_and_error_handling#declara%C3%A7%C3%B5es_de_manipula%C3%A7%C3%A3o_de_fluxo_e_manipula%C3%A7%C3%A3o_de_erros)
- **Site** MDN Web Docs: Error (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Error)
- **Site** Microsoft TechNet: JavaScript: Tratamento de erros (<https://social.technet.microsoft.com/wiki/pt-br/contents/articles/32455.javascript-tratamento-de-erros.aspx>)
- **Artigo** Exceções e Erros em JavaScript (<http://gabrielprates.com/2017/06/02/excecoes-e-erros-em-js.html>)
- **Artigo** Criando Exceptions para impressionar no Teste Técnico (<https://dev.to/he4rt/criando-exceptions-para-impressionar-no-teste-tecnico-2nie>)

Conteúdos Alura:**

- **Artigo** Lidando com erros no Node.js (<https://www.alura.com.br/artigos/lidando-com-erros-node-js>)
- **Curso** Curso Node.js: Criando sua primeira biblioteca (<https://cursos.alura.com.br/course/javascript-node-js-criando-primeira-biblioteca>)

Vue - Testes:

- Testes unitários são testes que validam a funcionalidade de partes individuais do seu código.
- Testes E2E simulam a experiência do usuário ao interagir com a aplicação, garantindo que todos os componentes funcionem juntos como esperado.
- Vue Test Utils é a biblioteca oficial para testes de componentes Vue.
- Aprenda a escrever testes básicos para seus componentes Vue, verificando se eles são renderizados corretamente.
- Os métodos dos componentes são partes cruciais do código que muitas vezes contêm lógica complexa. Aprenda a testá-los efetivamente.
- A injeção de dependência pode facilitar o teste de partes do seu código que dependem de serviços ou outros componentes.
- Se você estiver usando Vuex, Pinia ou Vue Router em seu aplicativo, aprender a lidar com eles nos testes é crucial.
- Cypress é uma ferramenta comum para realizar testes E2E em aplicações web, incluindo Vue. Instale e configure o Cypress para o seu projeto.
- Comece a escrever testes que simulam ações do usuário, como clicar em botões, preencher formulários e navegar entre as páginas.
- Os comandos personalizados ajudam a simplificar os testes, evitando a repetição de códigos de teste.
- Aprenda a manipular e testar diferentes estados da sua aplicação, como usuários logados e deslogados.
- Integre seus testes E2E ao seu processo de integração contínua/deploy contínuo (CI/CD) para garantir que sua aplicação esteja sempre funcionando como esperado.

Conteúdos

- **Site** Vue: Documentação - Testes de unidade (<https://vuejs.org/guide/scaling-up/testing.html#unit-testing>)
- **Site** Vue: Documentação - Testes fim-a-fim (<https://vuejs.org/guide/scaling-up/testing.html#e2e-testing>)

Conteúdos Alura:**

- **Artigo** Testes automatizados: como melhorar qualidade e confiabilidade do seu software (<https://www.alura.com.br/artigos/testes-automatizados>)
- **Artigo** Desmistificando testes de unidade no Vue (<https://www.alura.com.br/artigos/desmistificando-testes-de-unidade-no-vue>)
- **Artigo** Entrega e integração contínua de aplicações Vue (<https://www.alura.com.br/artigos/entrega-e-integracao-continua-de-aplicacoes-vue>)
- **YouTube** Alura: Testes em Javascript - Alura Live 50 (<https://www.youtube.com/watch?v=r1rndQwFLMY>)
- **Site** Primeiras aulas do curso Cypress: automação de testes E2E (<https://www.alura.com.br/conteudo/cypress-automacao-testes-e2e>)

Cypress:

- Criar e executar testes

Conteúdos

- **Site** Cypress: Documentação (inglês) (<https://docs.cypress.io/guides/getting-started/installing-cypress>)
- **Artigo** Um Overview sobre Cypress.io — Framework de Automação de Testes end-to-end (<https://medium.com/@faelbercam/um-overview-sobre-cypress-io-framework-de-automa%C3%A7%C3%A3o-de-testes-end-to-end-dc438b9ee7a1>)
- **YouTube** Embaixadora da Qualidade: Guia Rápido Sobre Cypress (<https://www.youtube.com/watch?v=ZWF7QT3ogk0>)
- **YouTube** Otavio Lemos: Guia Rápido Sobre Cypress (<https://www.youtube.com/watch?v=e5SuUUFIDss>)

Conteúdos Alura:**

- **Site** Primeiras aulas do curso Cypress: automação de testes E2E (<https://www.alura.com.br/conteudo/cypress-automatizando-testes-e2e>)
- **Curso** Curso Cypress: automação de testes E2E (<https://www.alura.com.br/curso-online-cypress-automacao-testes-e2e>)
- **Curso** Formação Carreira QA: processos e automação de testes (<https://www.alura.com.br/formacao-carreira-tester-qa>)

Babel - Fundamentos:

- Babel é um compilador JavaScript. É uma ferramenta usada principalmente para converter código ECMAScript 2015+ em uma versão compatível com versões anteriores do JavaScript em navegadores ou ambientes atuais e mais antigos.
- Compreender como Babel converte a sintaxe para JavaScript

Conteúdos

- **Site** Documentação: What is Babel? (inglês) (<https://babeljs.io/docs/en/>)
- **Artigo** O que é babel? (<https://coodesh.com/blog/dicionario/o-que-e-babel/>)
- **Artigo** React e Babel — Como melhorar o desempenho de suas aplicações com plugins, presets e benchmark! (<https://medium.com/nossa-coletividade/react-e-babel-como-melhorar-o-desempenho-de-suas-aplica%C3%A7%C3%B5es-com-plugins-presets-e-benchmark-fb35700a52e9>)
- **YouTube** Matheus Castiglioni: React, Babel e Webpack, o que é tudo isso? (<https://www.youtube.com/watch?v=y5B0MXmt428>)
- **YouTube** web3escola: Introdução ao React - Babel (<https://www.youtube.com/watch?v=jSGg2j2UNr4>)

Conteúdos Alura:**

- **Curso** Curso JavaScript: salvando dados localmente com IndexedDB (<https://www.alura.com.br/curso-online-javascript-es6-orientacao-a-objetos-parte-3>)

Nível 3

Vue - Performance:

- Aprenda a usar ferramentas de profiling, como a Vue Devtools e o Performance tab no Chrome Devtools, para entender onde seu aplicativo pode estar sofrendo em termos de desempenho.
- Estude técnicas de otimização de carga de página, como lazy-loading, prefetching e preloading.
- Compreenda a importância de escolher a arquitetura certa para a sua aplicação Vue. Aprenda sobre padrões de projeto e como eles podem afetar a performance.
- Familiarize-se com as técnicas de redução do tamanho do bundle e tree-shaking para remover código desnecessário da sua aplicação.
- Aprenda a implementar a divisão de código para carregar apenas o código necessário para o usuário em um determinado momento.
- Aprofunde-se em técnicas de otimização de atualizações, como imutabilidade, estabilidade de props e uso de chaves em listas para minimizar re-renderizações desnecessárias.
- Entenda como garantir a estabilidade de props para evitar atualizações desnecessárias dos componentes.
- Familiarize-se com as diretivas v-once e v-memo e quando utilizá-las para otimizar a renderização de componentes.
- Aprenda sobre otimizações gerais como a virtualização de listas grandes, reduzindo a sobrecarga de reatividade para grandes estruturas imutáveis, e evitando abstrações de componentes desnecessárias.
- Entenda como a virtualização pode ser usada para exibir eficientemente grandes listas de dados, reduzindo a sobrecarga de renderização.

Conteúdos

- [Site](https://web.dev/performance-optimizing-content-efficiency/) Otimizando a eficiência do conteúdo (<https://web.dev/performance-optimizing-content-efficiency/>)
- [Site](https://vuejs.org/api/built-in-directives.html#v-once) Vue: Documentação - v-once (<https://vuejs.org/api/built-in-directives.html#v-once>)
- [Site](https://vuejs.org/api/built-in-directives.html#v-memo) Vue: Documentação - v-memo (<https://vuejs.org/api/built-in-directives.html#v-memo>)
- [Site](https://vuejs.org/guide/best-practices/performance.html#profiling-options) Vue: Documentação - profiling (<https://vuejs.org/guide/best-practices/performance.html#profiling-options>)
- [Site](https://vuejs.org/guide/scaling-up/ssr.html) Vue: Documentação - SSR (<https://vuejs.org/guide/scaling-up/ssr.html>)
- [Site](https://vuejs.org/guide/best-practices/performance.html#code-splitting) Vue: Documentação - Code Splitting (<https://vuejs.org/guide/best-practices/performance.html#code-splitting>)
- [Site](https://github.com/Akryum/vue-virtual-scroller) Listas virtualizadas com vue-virtual-scroller (<https://github.com/Akryum/vue-virtual-scroller>)

Vue - Componentes assíncronos:

- Compreenda o que são e por que usar componentes assíncronos no Vue.
- Aprenda a criar um componente assíncrono básico usando a função defineAsyncComponent.
- Domine a renderização de um estado de loading enquanto o componente assíncrono é carregado.
- Implemente o tratamento de erros ao carregar o componente assíncrono.
- Aplique lazy loading nos seus componentes para melhorar a performance da aplicação.
- Use componentes assíncronos com Vue Router para carregar rotas sob demanda.

Conteúdos

- [Site](https://vuejs.org/guide/components/async.html#basic-usage) Vue: Documentação - Componentes assíncronos (<https://vuejs.org/guide/components/async.html#basic-usage>)

Vue - Teleporte:

- Entenda o que é o Teleport e qual problema ele resolve no Vue.
- Aprenda a usar o componente para renderizar conteúdo em um local diferente do DOM.
- Domine a propriedade 'to' do Teleport, usada para especificar o local de destino no DOM.
- Explore como usar diretivas como v-if para controlar a renderização do conteúdo teleportado.
- Experimente casos de uso avançados, como teleportar modais, tooltips, notificações e muito mais.

Conteúdos

- [Site](https://vuejs.org/guide/built-ins/teleport.html) Vue: Documentação - Teleporte (<https://vuejs.org/guide/built-ins/teleport.html>)
- [Artigo](https://medium.com/@marcosviniciosneves/vue-teleport-8f9788ca9ca1) Vue Teleport (<https://medium.com/@marcosviniciosneves/vue-teleport-8f9788ca9ca1>)

Vue Slots:

- Comece compreendendo o conceito de slots e como eles são usados para criar layouts personalizáveis e reutilizáveis em componentes Vue.
- Aprenda a usar slots anônimos, que são os slots padrão que você usa quando deseja inserir conteúdo personalizado no componente.
- Mergulhe em slots nomeados, que permitem que você personalize diferentes partes do layout do componente.
- Domine os scoped slots, que permitem o acesso aos dados do componente filho dentro do slot.
- Familiarize-se com a diretiva v-slot e como usá-la para criar slots nomeados e scoped slots.
- Descubra como usar slots dinâmicos para criar layouts ainda mais flexíveis.
- Entenda a diferença entre a nova sintaxe de slots no Vue 3 e a antiga no Vue 2 para garantir a compatibilidade com versões anteriores.

Conteúdos

- [Site](https://vuejs.org/guide/components/slots.html) Vue: Documentação - slots (<https://vuejs.org/guide/components/slots.html>)

Quasar Framework:

- Entenda o que é o Quasar e como ele se encaixa no ecossistema Vue.
- Configure o Quasar em seu ambiente de desenvolvimento local e aprenda como criar um novo projeto Quasar.
- Aprenda a usar os componentes pré-fabricados do Quasar, que são otimizados para eficiência e personalização.
- Domine o sistema de layout e roteamento do Quasar para construir aplicativos complexos e interativos.
- Aprenda a gerenciar o estado de sua aplicação Quasar com Vuex, a biblioteca de gerenciamento de estado do Vue.js.
- Conheça a CLI do Quasar, uma ferramenta poderosa para gerar, desenvolver e gerenciar seus projetos Quasar.
- Conheça as diferentes opções para o deploy de aplicações feitas com Quasar.

Conteúdos

- **Site** Quasar Framework (<https://quasar.dev/>)

Conteúdos Alura:**

- **Artigo** PWA Criação e Publicação com Quasar e Vue.js (<https://www.alura.com.br/artigos/pwa-criacao-publicacao>)

Nuxt:

- Entenda o que é Nuxt.js, como ele funciona e quando usá-lo.
- Aprenda como instalar e configurar um projeto Nuxt.js a partir do zero.
- Familiarize-se com a estrutura de pastas de um projeto Nuxt.js e o que cada pasta faz.
- Compreenda a geração automática de rotas em Nuxt.js e como trabalhar com rotas dinâmicas.
- Aprenda a integrar soluções de gerenciamento do estado global em uma aplicação Nuxt.js.
- Entenda as diferenças entre SSR e SSG e como usá-los em Nuxt.js.
- Aprenda a usar plugins Nuxt.js para adicionar funcionalidades globais à sua aplicação.
- Explore como o middleware pode ajudá-lo a executar código antes de renderizar a página.
- Conheça as diferentes opções de deploy de uma aplicação Nuxt.js.

Conteúdos

- **Site** Nuxt: The Intuitive Vue Framework (<https://v2.nuxt.com/docs/get-started/installation>)

Pinia:

- Aprenda os conceitos básicos do Vuex, o problema que ele resolve e quando usá-lo.
- Compreenda como o estado é armazenado no Vuex e como acessar o estado global em seus componentes Vue.
- Aprenda a usar getters para acessar valores derivados do estado.
- Descubra como realizar ações assíncronas através de actions.
- Aprenda como usar a store fora de componentes com o hook useStore
- Explore os guias de migração para mudar do vuex para o Pinia
- Mergulhe no universo do SSR combinando o poder do Pina com o Nuxt

Conteúdos

- **Site** Pinia: livro de receitas (<https://pinia.vuejs.org/cookbook/>)
- **Site** Pinia: principais conceitos (<https://pinia.vuejs.org/core-concepts/>)

Vue - Provide / Inject:

- Compreenda o que são e por que você pode precisar deles. Eles são usados para injetar dependências em componentes descendentes sem passar props em todos os níveis.
- Aprenda a usar a opção provide em um componente para fornecer dados que podem ser injetados em componentes descendentes.
- Veja como a opção inject é usada em um componente descendente para acessar dados fornecidos por um componente ancestral.
- Descubra como o Provide / Inject funciona com a Composition API, usando provide e inject funções no método setup.
- Saiba como reagir às mudanças nos valores fornecidos usando um objeto reativo ou ref.

- Aprenda a lidar com situações em que vários ancestrais fornecem a mesma chave.
- Embora seja uma ferramenta poderosa, o Provide / Inject deve ser usado com moderação. Entenda quando é apropriado usá-lo e quando é melhor evitar.

Conteúdos

- **Site** Vue: Documentação - Provide / Inject (<https://vuejs.org/guide/components/provide-inject.html>)

Vue - Suspense:

- O Suspense é um componente embutido para orquestração de dependências assíncronas em uma árvore de componente. Ele pode interpretar um estado de carregamento enquanto espera por várias dependências assíncronas encaixadas em baixo da árvore de componente ser resolvida.
- Conhecer dependências assíncronas

Conteúdos

- **Site** Vue: Documentação - Suspense (<https://pt.vuejs.org/guide/built-ins/suspense.html>)
- **YouTube** Clube Full-Stack: Como usar o Suspense no vue 3 | Como colocar um loading em um componente assíncrono (https://www.youtube.com/watch?v=imR45gszsHw&ab_channel=ClubeFull-Stack)

NativeScript-Vue:

- O NativeScript-Vue é um plugin NativeScript que permite usar Vue.js para criar seu aplicativo móvel.
- Criar apps com interface nativa

Conteúdos

- **Site** Documentação: NativeScript-Vue (<https://nativescript-vue.org/pt-br/docs/introduction/>)
- **Artigo** Por que NativeScript-Vue? (<https://rafaelaugustodev.medium.com/por-que-nativescript-vue-d308e85cdf9a>)

Habilidade Auxiliar: Infraestrutura e Back-end

Git e GitHub - Fundamentos:

- Git é um sistema de controle de versão distribuído gratuito e de código aberto projetado para lidar com tudo, desde projetos pequenos a muito grandes com velocidade e eficiência.
- GitHub é um serviço de hospedagem para desenvolvimento de software e controle de versão usando Git.
- Criar um repositório
- Clonar um repositório
- Fazer commit, push e pull de e para o repositório
- Reverter um commit
- Criar branches e pull requests
- Lidar com merge e conflitos

Conteúdos

- **Site** GitHub Documentação (<https://docs.github.com/pt>)

- **Site** GitHub Pages Documentação (<https://docs.github.com/pt/pages/getting-started-with-github-pages/about-github-pages>)
- **Site** Git School - Visualizing Git (<https://git-school.github.io/visualizing-git/>)
- **Site** Dangit, Git!?! (<https://dangitgit.com/>)
- **YouTube** Rafaella Ballerini: O que é Git e GitHub? - definição e conceitos importantes 1/2 (<https://www.youtube.com/watch?v=DqTITcMq68k>)
- **YouTube** Rafaella Ballerini: Como usar Git e GitHub na prática! - desde o primeiro commit até o pull request! 2/2 (<https://www.youtube.com/watch?v=UBAX-13g8OM>)
- **YouTube** Mario Souto - Dev Soutinho: Git: Entendendo de vez como funciona do melhor e mais visual jeito possível (<https://www.youtube.com/watch?v=4-tfJ-ZyAOQ>)
- **YouTube** Mario Souto - Dev Soutinho: Como colocar seu projeto no ar DE GRAÇA via GitHub! | Hospedagem com GitHub Pages (https://www.youtube.com/watch?v=BU-w2_Aae54)
- **YouTube** CodandoTV(Rods) - 5 coisas que você precisa saber sobre Git (https://youtu.be/MqoqPzjQyCY?si=VL_mT8EowuemptmU)
- **YouTube** CodandoTV(Rods) - README de Sucesso: Transforme seu Projeto ou seu Perfil em Destaque no GitHub (https://youtu.be/v9ZM2PVzctM?si=49ah_HRPlwpSEX4A)

Conteúdos Alura:**

- **Artigo** Git e Github: O que são, Como Configurar e Primeiros Passos (<https://www.alura.com.br/artigos/o-que-e-git-github>)
- **Artigo** Mais git com o hub: a linha de comando do Github (<https://www.alura.com.br/artigos/github-na-linha-de-comando>)
- **Podcast** Hipsters 184: Guia do Iniciante em Github (<https://cursos.alura.com.br/extra/hipsterstech/guia-do-iniciante-em-github-hipsters-184-a378>)
- **Site** GitHub: diferentes maneiras de compartilhar seu projeto (<https://cursos.alura.com.br/extra/alura-mais/github-diferentes-maneiras-de-compartilhar-seu-projeto-c2002>)
- **Site** Websérie: Git e Github para Sobrevivência (<https://www.alura.com.br/webseries/git-e-github-para-sobrevivencia>)
- **Podcast** Hipsters 109: Git e Github (<https://www.alura.com.br/podcast/hipsterstech-git-e-github-hipsters-109-a474>)
- **YouTube** Alura: Git e Github para Sobrevivência 01: Como o Git funciona? (<https://www.youtube.com/watch?v=BAmvmaKQkIQ>)
- **Curso** Curso Git e GitHub: compartilhando e colaborando em projetos (<https://cursos.alura.com.br/course/git-github-compartilhando-colaborando-projetos>)
- **Curso** Curso Git e GitHub: dominando controle de versão de código (<https://cursos.alura.com.br/course/git-github-dominando-controle-versao-codigo>)
- **Desafio** 7 Days of Code: GitHub (<https://7daysofcode.io/matricula/github>)

HTTP - Fundamentos:

- HTTP significa Hyper Text Transfer Protocol. A comunicação entre computadores cliente e servidores web é feita enviando solicitações HTTP e recebendo respostas HTTP.
- Entender a diferença dos verbos HTTP
- Testar os requests e ver os status codes no navegador
- Saber fazer uma requisição HTTP na linha de comando com WGET
- Baixar uma imagem com WGET

- Fazer um post

Conteúdos

- **Site** MDN Web Docs: Uma visão geral do HTTP (<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview>)
- **Site** MDN Web Docs: Métodos de requisição HTTP (<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods>)
- **Site** HTTP Cats (<https://http.cat/>)
- **Site** HTTP Dogs (<https://http.dog/>)
- **YouTube** Fabiano Gabardo Lemos: Requisições HTTP - GET, POST, PUT, PATCH DELETE (<https://www.youtube.com/watch?v=kncOJZrnkTg>)
- **YouTube** Programador a Bordo: Protocolo HTTP e TCP/IP - Introdução (<https://www.youtube.com/watch?v=V4XZ81vRGtM>)

Conteúdos Alura:**

- **Artigo** HTTP: Desmistificando o protocolo da Web (<https://www.alura.com.br/artigos/desmistificando-o-protocolo-http-parte-1>)
- **Artigo** Métodos de requisição do HTTP (<https://www.alura.com.br/artigos/metodos-de-requisicao-do-http>)
- **Artigo** Qual é a diferença entre HTTP e HTTPS? (<https://www.alura.com.br/artigos/qual-e-diferenca-entre-http-e-https>)
- **Podcast** Hipsters.tech: HTTP/2: magia com o novo protocolo - Hipsters 13 (<https://www.alura.com.br/podcast/http-2-magia-com-o-novo-protocolo-hipsters-13-a573>)
- **Curso** Curso HTTP: Entendendo a web por baixo dos panos (<https://www.alura.com.br/curso-online-http-entendendo-web-por-baixo-dos-panos>)

JSON:

- JSON significa JavaScript Object Notation (notação de objeto JavaScript). É um formato de texto para armazenar e transmitir dados.
- Criar um objeto
- Transformar um objeto em uma string
- Transformar uma string em objeto
- Manipular um objeto

Conteúdos

- **Site** MDN Web Docs: JSON (https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/JSON)
- **Site** MDN Web Docs: Trabalhando com JSON (<https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Objects/JSON>)
- **Artigo** Introdução ao JSON (<https://juliocarneiro.medium.com/introdu%C3%A7%C3%A3o-ao-json-7825b1a550ff>)
- **YouTube** Marco Bruno: O que é JSON e como criar um objeto (<https://www.youtube.com/watch?v=oCY5YEEjlwE>)

Conteúdos Alura:**

- **Artigo** O que é JSON? (<https://www.alura.com.br/artigos/o-que-e-json>)
- **Curso** Curso MySQL e JSON: persistindo JSON de maneira eficiente (<https://www.alura.com.br/curso-online-mysql-json-persistencia>)

Linha de comando - Fundamentos:

- CLI é um programa de linha de comando que aceita entradas de texto para executar funções do sistema operacional.
- Conhecer os principais comandos

Conteúdos

- **Site** Microsoft Docs: Como usar argumentos de linha de comando para terminal do Windows (<https://docs.microsoft.com/pt-br/windows/terminal/command-line-arguments?tabs=windows>)
- **YouTube** Estevan Maito: Linha de comando básica - 8 comandos mais frequentes - WINDOWS e LINUX (<https://www.youtube.com/watch?v=rKBqXJZocYk>)
- **YouTube** Ninja do Linux: Comandos básicos da linha de comando do Linux (https://www.youtube.com/watch?v=rs_yshFGu8E)

Conteúdos Alura:**

- **Artigo** CMD: dicas para trabalhar no prompt do Windows (<https://www.alura.com.br/artigos/cmd-dicas-para-trabalhar-no-prompt-do-windows>)
- **Artigo** Como configurar variáveis de ambiente no Windows, Linux e macOS (<https://www.alura.com.br/artigos/configurar-variaveis-ambiente-windows-linux-macos>)
- **Site** Primeiras aulas do Curso Windows Prompt: Trabalhando na linha de comando (<https://www.alura.com.br/conteudo/windows-prompt-utilizando-cmd>)
- **Curso** Curso Windows Prompt: Trabalhando na linha de comando (<https://www.alura.com.br/curso-online-prompt>)
- **Curso** Curso Terminal: aprenda comandos para executar tarefas (<https://cursos.alura.com.br/course/terminal-comandos-executar-tarefas>)

Cloud - Fundamentos:

- Cloud, ou computação em nuvem é a distribuição de serviços de computação pela Internet usando um modelo de preço pago conforme o uso. Uma nuvem é composta de vários recursos de computação, que abrangem desde os próprios computadores (ou instâncias, na terminologia de nuvem) até redes, armazenamento, bancos de dados e o que estiver em torno deles. Ou seja, tudo o que normalmente é necessário para montar o equivalente a uma sala de servidores, ou mesmo um data center completo, estará pronto para ser utilizado, configurado e executado.
- Conhecer a diferença entre IaaS, PaaS e SaaS
- Conhecer os maiores provedores de cloud
- Especializar-se em algum provedor

Conteúdos

- **Artigo** Computação em nuvem (<https://medium.com/sysadminas/computa%C3%A7%C3%A3o-em-nuvem-515930304cf9>)
- **Artigo** O que é cloud? (<https://gabriel-faraday.medium.com/o-que-%C3%A9-cloud-991109e708c6>)
- **YouTube** Gabs Ferreira: Por que investir e estudar cloud? (<https://www.youtube.com/watch?v=Z45BTNeZ1l0>)
- **YouTube** Andre Iacono: O que é MICROSOFT AZURE? Qual Certificação começar em 2022? (<https://www.youtube.com/watch?v=f-oVzkvMwnE>)
- **YouTube** AWS: O que é a AWS? (<https://www.youtube.com/watch?v=8JI9wQ8sUdQ>)
- **YouTube** O que é Google Cloud e por que aprender? (<https://www.youtube.com/shorts/Lzq3f1DHWcl>)

- **Artigo** AWS vs Google Cloud vs Azure: o que cada um tem de melhor? (<https://medium.com/data-hackers/aws-vs-google-cloud-vs-azure-o-que-cada-um-tem-de-melhor-52107174f7b7>)
- **YouTube** Código Fonte TV: Azure (A plataforma Cloud da Microsoft) (<https://www.youtube.com/watch?v=YgE-sZaCuJ0>)
- **YouTube** Mundo da Cloud: AWS do Zero ao Expert (<https://www.youtube.com/watch?v=HiBCv9DolxI&list=PLtL97Owd1gkQ0dfqGW8OtJ-155Gs67Ecz>)

Conteúdos Alura:**

- **Podcast** Hipsters.tech: TechGuide - Fundamentos Cloud – Hipsters Ponto Tech #348 (<https://www.hipsters.tech/techguide-fundamentos-cloud-hipsters-ponto-tech-348/>)
- **Artigo** Cloud: o que é, História e Guia da computação em nuvem (<https://www.alura.com.br/artigos/cloud>)
- **Podcast** Hipsters.tech: Uma jornada Para o Cloud - Hipsters Deep Dive 005 (<https://www.alura.com.br/podcast/uma-jornada-para-o-cloud-hipsters-deep-dive-005-a1100>)
- **Podcast** Hipsters.tech: Histórias do Cloud - Hipsters 04 (<https://www.alura.com.br/podcast/historias-do-cloud-hipsters-04-a582>)
- **Artigo** Heroku, Vercel e outras opções de cloud como plataforma (<https://www.alura.com.br/artigos/heroku-vercel-outras-opcoes-cloud-plataforma>)
- **Artigo** AWS: Guia sobre o que é Amazon Web Services, seus Serviços e Certificações (<https://www.alura.com.br/artigos/aws>)
- **Artigo** Terraform: criando máquinas na Azure (<https://www.alura.com.br/artigos/terraform-maquinas-na-azure>)
- **YouTube** Alura: O que é cloud? (<https://www.YOUTUBE.com/watch?v=wev9fMrg-TU>)
- **YouTube** Alura: AWS, Google Cloud e Azure: Por onde começar? | Hipsters.Talks (<https://www.YOUTUBE.com/watch?v=z9k6rsdmWc0&t=300s>)
- **YouTube** Alura: Certificação em Cloud: Azure, AWS, Google | Hipsters.Talks (https://www.YOUTUBE.com/watch?v=W4K82n_WK5g&t=290s)
- **Curso** Formação Começando em Cloud Computing (<https://cursos.alura.com.br/formacao-cloud-computing>)
- **Curso** Formação Amazon Web Services (<https://cursos.alura.com.br/formacao-amazon-web-services>)
- **Curso** Formação Google Certified Associate Cloud Engineer (<https://cursos.alura.com.br/formacao-google-certified-associate-cloud-engineer>)
- **Curso** Formação Certificação AWS Certified Cloud Practitioner (<https://cursos.alura.com.br/formacao-aws-certified-cloud-practitioner>)
- **Curso** Formação Containers com AWS ECS e EKS (<https://cursos.alura.com.br/formacao-containers-aws>)
- **Curso** Formação Google Cloud Platform (<https://cursos.alura.com.br/formacao-google-cloud>)
- **Curso** Formação Certificação Google Certified Associate Cloud Engineer (<https://cursos.alura.com.br/formacao-google-certified-associate-cloud-engineer>)
- **Curso** Formação Azure (<https://cursos.alura.com.br/formacao-conhecendo-azure>)
- **Curso** Formação Certificação AZ-900: Microsoft Azure Fundamentals (<https://cursos.alura.com.br/formacao-certificacao-az-900-microsoft-azure-fundamentals>)

YARN:

- Yarn é um gerenciador de pacotes para seu código. Ele permite que você use e compartilhe código com outros desenvolvedores. O código é compartilhado por meio de algo chamado pacote (às vezes chamado de módulo). Um pacote contém todo o código que está sendo compartilhado, bem como um arquivo package.json que descreve o pacote.
- Gerenciar pacotes
- Gerenciar dependências
- Instalação de pacotes offline
- Comandos
- Arquivo yarn.lock

Conteúdos

- **Site** YARN: Documentação (inglês) (<https://classic.yarnpkg.com/en/docs>)
- **Artigo** Porque você deveria dar uma chance ao Yarn? (<https://rafaell-lycan.com/2017/porque-voce-deveria-dar-uma-chance-yarn/>)
- **YouTube** Lucas Santos: Yarn VS NPM, qual é melhor? (<https://www.youtube.com/watch?v=vxES6rbrd-U>)

Conteúdos Alura:**

- **Artigo** NPM vs Yarn (<https://www.alura.com.br/artigos/npm-vs-yarn>)

Vite:

- Instalação: Instale o Vite globalmente e crie seu projeto Vue com um simples comando create-vite
- Explore a estrutura de diretórios do Vite e entenda como organizar seus componentes e rotas.
- Aprenda a usar os comandos de desenvolvimento e build do Vite para executar e otimizar sua aplicação.

Conteúdos

- **Site** Vite: Documentação (<https://pt.vitejs.dev/guide/why.html>)
- **Site** Vite: Documentação - Interface da Linha de Comando (<https://pt.vitejs.dev/guide/cli.html>)
- **Site** Vite: Documentação - Comparações (<https://pt.vitejs.dev/guide/comparisons.html>)

Habilidade Auxiliar: UX e Design

Design System:

- Um Design System (sistema de design) é uma coleção de componentes reutilizáveis, guiados por padrões claros, que podem ser colocados juntos para construir aplicações.
- Criar e manter bibliotecas que serão consumidas e usadas como padrão para a construção de um projeto
- Design tokens
- Estilos fundamentais
- Construção de componentes
- Microinterações
- Documentação

Conteúdos

- **Artigo** Design System: o que é e quais os benefícios? (<https://medium.com/ipnet-growth-partner/design-system-o-que-e-438773dd811>)
- **Artigo** Afinal, o que é Design System? (<https://brasil.uxdesign.cc/afinal-o-que-%C3%A9-design-system-448c257b0021>)
- **Artigo** O que são Design Tokens (<https://medium.com/pretux/design-tokens-112b2ee11ddf>)
- **YouTube** Caio Gonzalez: O que é Design System? / Guia para começar o seu próprio Design System (<https://www.youtube.com/watch?v=ajqbXpFVaAw>)

Conteúdos Alura:**

- **Artigo** O que é Design System? (<https://www.alura.com.br/artigos/o-que-e-design-system>)
- **Artigo** Design Systems: exemplos práticos (<https://www.alura.com.br/artigos/design-systems-exemplos-praticos>)
- **YouTube** Alura: Design System & Style Guide (<https://www.youtube.com/watch?v=rSLvpOqC5wQ>)
- **YouTube** Alura: Design Systems (com Charles Assunção) (<https://www.youtube.com/watch?v=MLGFctEtmYQ>)
- **Podcast** Hipsters.tech: Design Systems - Hipsters 170 (<https://www.alura.com.br/podcast/hipsterstech-design-systems-hipsters-170-a399>)
- **Podcast** Layers.tech: Design System – Layers ponto tech 36 (<https://www.alura.com.br/podcast/layerstech-design-system-layers-ponto-tech-36-a1056>)
- **Curso** Design System: definindo estilos e tokens (<https://cursos.alura.com.br/course/design-system-definindo-estilos-tokens>)
- **Curso** Design System: projetando e construindo componentes (<https://cursos.alura.com.br/course/design-system-projetando-construindo-componentes>)
- **Curso** Design System: documentando um design system (<https://cursos.alura.com.br/course/design-system-documentando-design-system>)
- **Desafio** Experimente o que é ser um(a) profissional de Ux (<https://www.alura.com.br/challenges/ux-4>)

Figma - Fundamentos:

- Figma é uma aplicação web colaborativa para design de interfaces. O conjunto de recursos do Figma se concentra na interface do usuário e no design da experiência do usuário, com ênfase na colaboração em tempo real, utilizando uma variedade de editores de gráficos vetoriais e ferramentas de prototipagem.
- Criar layouts de páginas e componentes

Conteúdos

- **Site** Figma Básico - Primeiros passos com Figma (<https://www.figma.com/community/file/1190593276621265988>)
- **Artigo** O que é o Figma e por que usar ele? (<https://medium.com/nerdzaio/o-que-%C3%A9-o-figma-e-por-que-usar-ele-a71fbf1dbdd8>)
- **Artigo** Entenda o que é e como usar o Figma para criar designs (<https://blog.b2bstack.com.br/figma/>)
- **YouTube** Marco Bruno: Como usar o Figma! (<https://www.youtube.com/watch?v=qoE-2YFeW-Q>)
- **YouTube** 4 Exemplos de Componentes interativos no Figma (<https://www.youtube.com/watch?v=mGL3jrY-OBc>)

Conteúdos Alura:**

- **Artigo** Figma: o que é a ferramenta, Design e uso (<https://www.alura.com.br/artigos/figma>)
- **Site** Como Front-End utiliza o Figma - Alura+ (<https://cursos.alura.com.br/extra/alura-mais/como-front-end-utiliza-o-figma-c858>)
- **Artigo** Top 5 plugins no Figma para trabalhar com Design System (<https://www.alura.com.br/artigos/top-5-plugins-figma-trabalhar-com-design-system>)
- **Artigo** 5 plugins essenciais que você precisa ter no seu Figma (<https://www.alura.com.br/artigos/5-plugins-essenciais-do-figma>)
- **Artigo** 10 truques incríveis e pouco conhecidos no Figma (<https://www.alura.com.br/artigos/10-truques-incriveis-pouco-conhecidos-figma>)
- **Podcast** Layers.tech: Figma do Design ao Código 03 (<https://www.alura.com.br/podcast/layerstech-figma-do-design-ao-codigo-layers-ponto-tech-03-a726>)
- **YouTube** Como fazer estilos locais mais rapidamente no Figma? (https://www.youtube.com/watch?v=fCHd2_lAcQQ)
- **Curso** Formação Figma (<https://www.alura.com.br/formacao-figma>)

Componentes de design:

- Conhecer os componentes descrevem um layout ou interface

Conteúdos

- **Artigo** Componentes Principais UI (<https://medium.com/@prikuer/componentes-principais-ui-8b538a3aaaa5>)
- **Artigo** Compartilhando CSS em projetos do Nx (<https://www.alura.com.br/artigos/compartilhando-css-projetos-nx>)
- **YouTube** danielmacedoco: UI Design e Design System - Visão geral e prática sobre alguns componentes (<https://www.youtube.com/watch?v=VpgQNeSltpA>)
- **YouTube** 4 Exemplos de Componentes interativos no Figma (<https://www.youtube.com/watch?v=mGL3jrY-OBc>)

Conteúdos Alura:**

- **Curso** Curso Design System: criando componentes e documentando (<https://www.alura.com.br/curso-online-design-system-componentes-documentando>)
- **Curso** Curso Figma: conhecendo componentes da interface (<https://www.alura.com.br/curso-online-figma-conhecendo-componentes-interface>)

Sistemas de cores:

- Definir uma paleta de cores que faça sentido para determinada interface

Conteúdos

- **Artigo** Cores em UI: Um Guia Rápido Para Usar em Seus Projetos (<https://medium.com/acla/cores-em-ui-um-guia-r%C3%A1pido-para-usar-em-seus-projetos-31ccffe3e16b>)
- **Artigo** A Psicologia das cores e sua relação com o UX Design (<https://brasil.uxdesign.cc/a-psicologia-das-cores-e-sua-rela%C3%A7%C3%A3o-com-o-ux-design-af02460639cd>)
- **YouTube** kacio.design: Aplicação de Cores - Princípios do UI (https://www.youtube.com/watch?v=C_VhzEyqT6U)

Conteúdos Alura:**

- **Curso** Curso Cores para Designers: escolhendo e trabalhando com cores em um projeto (<https://www.alura.com.br/curso-online-cores-para-seu-projeto>)
- **Site** Primeiras aulas do curso Cores: sistemas básicos e paletas (<https://www.alura.com.br/conteudo/fundamentos-da-cor>)
- **Curso** Cores: sistemas básicos e paletas (<https://www.alura.com.br/curso-online-fundamentos-da-cor>)
- **Curso** Curso Design editorial: criação de materiais gráficos (<https://www.alura.com.br/curso-online-design-editorial>)

Como usar fontes:

- Escolher a fonte mais adequada para determinado projeto

Conteúdos

- **Artigo** Você sabe usar tipografia em UI Design? (<https://medium.com/ui-lab-school/voc%C3%AA-sabe-usar-tipografia-em-ui-design-9ce4ccdbab43>)
- **Artigo** As 20 fontes mais usadas pelos designers gráficos (<https://apenasumchico.medium.com/as-20-fontes-mais-populares-de-todos-os-tempos-38a6c121dfb>)
- **Artigo** Usando ícones SVG como fontes com o IcoMoon (<https://dev.to/sucodelarangela/usando-icone-svg-como-fontes-com-o-icomoon-563e>)
- **YouTube** Nadine Fronza: Como escolher a melhor fonte para seu projeto (<https://www.youtube.com/watch?v=MSKPUtldadI>)
- **YouTube** Thiago Abreu: Como Escolher Fontes para um Trabalho de Design (<https://www.youtube.com/watch?v=I-TrNbZi-aU>)

Conteúdos Alura:**

- **Artigo** 10 tipografias gratuitas para um design incrível (<https://www.alura.com.br/artigos/10-tipografias-gratuitas-design-incrivei>)
- **Site** Tipografia para Web (<https://tipografiaparaweb.netlify.app/>)
- **Site** Primeiras aulas do curso Tipografia: conhecendo o que há por trás dos tipos (<https://www.alura.com.br/conteudo/tipografia-conceito>)
- **Curso** Curso Tipografia: conhecendo o que há por trás dos tipos (<https://www.alura.com.br/curso-online-tipografia-conceito>)
- **Curso** Curso Design editorial: criação de materiais gráficos (<https://www.alura.com.br/curso-online-design-editorial>)