**SUMÁRIO**

[1. programação reativa 1](#_Toc57428053)

[2. sobre vuejs 1](#_Toc57428054)

[3. cdn e npm 2](#_Toc57428055)

[3.1. VUEJS CDN 2](#_Toc57428056)

[4. introdução vuejs com cdn 3](#_Toc57428057)

[5. ARQUITETURA inicial VUEJS – com CDN 5](#_Toc57428058)

[5.1. INSTÂNCIA VUEJS 5](#_Toc57428059)

[5.2. ESTRUTURA DE UMA INSTÂNCIA VUE 5](#_Toc57428060)

[5.2.1. TEMPLATE E ATRIBUTO EL 5](#_Toc57428061)

[5.2.2. ATRIBUTO DATA 6](#_Toc57428062)

[5.2.3. ATRIBUTO METHODS E COMPUTED 7](#_Toc57428063)

[5.2.4. ATRIBUTO WATCH – OBSERVADOR 8](#_Toc57428064)

[5.2.5. DIRETIVAS 9](#_Toc57428065)

[5.3. MANIPULANDO EVENTOS 10](#_Toc57428066)

[6. BIBLIOGRAFIA 11](#_Toc57428067)

# programação reativa

Programação reativa é programar com fluxos (streams) de dados assíncronos, isto é, que fluxos de eventos que ocorrem a independer absolutamente de outros referencias programados de antemão. É sobre criar sistemas que “reagem” em tempo real, e de forma adaptativa – é sobre a maximização da dinamicidade.

A forma tradicional de desenvolver aplicações se resume a criar tarefas que se comunicam em tempos exatamente determinados, com respostas invariáveis, e sem escalabilidade, ou seja, em outras palavras, seguindo regras rígidas pré-definidas.

Por exemplo, imagine um sistema abstrato, organizado em um fluxo de dados, sequencial, de A até Z. Se ocorresse uma falha, na fila de processos, que impedisse o fluxo de prosseguir em ordem alfabética, até o seu final conclusivo Z, o que ocorreria? Em uma programação tradicional, isto seria uma falha que impossibilitaria a continuação da operação. No entanto, em uma programação reativa hipotética, o sistema poderia compreender que, se uma falha ocorreu na sequência de, por exemplo, D para E, seria possível contactar e pular para F, afim de alcançar igualmente, a partir deste, o elemento Z.

Ou seja, a programação reativa é uma que é inteligente o suficiente para isolar informações afim de construir um ecossistema escalável e adaptável a mudanças assíncronas, e para propagar as atualizações ao resto dos receptores reativos.

*“É um paradigma de programação orientado a fluxo de dados e propagação de mudança.”*

# sobre vuejs

O Vue.js (comumente conhecido como Vue, pronunciado "view") é um framework progressivo do JavaScript de código aberto (open source) para a construção de interfaces de usuário. A integração em projetos que usam outras bibliotecas de JavaScript é facilitada com o Vue porque ele foi projetado para ser adotado de forma incremental. O Vue também pode funcionar como uma estrutura de aplicativos web capaz de alimentar aplicativos avançados de uma única página.

# cdn e npm

## VUEJS CDN

A primeira forma, mais simples, para fins de aprendizado e prototipação, com VueJS, é utilizar o link CDN diretamente nas aplicações.

Com este, obtemos acessos aos recursos necessários e mais basilares para trabalhar com Vue em aplicações já existentes, tornando-o um elemento complementar, ou do zero, mas sem todo o arcabouço possível de ser obtido com o uso do NPM.

VUEJS NPM

O NPM é um poderoso gerenciador de pacotes utilizado para administrar as bibliotecas e frameworks utilizados em uma aplicação.

Este, que significa ***Node Package Manager***, é um poderoso gerenciador de pacotes que faz parte do Node.js. Esse, por sua vez, é um ambiente para a execução de JavaScript no lado do servidor de hospedagem. Em outras palavras, ele permite utilizar a linguagem JavaScript no back-end da aplicação, o que não era possível antes da existência do Node.

Trata-se de uma ferramenta crucial para realizar um desenvolvimento organizado e otimizado, pois permite instalar dependências em um projeto, mantê-las atualizadas e muito mais.

# introdução vuejs com cdn

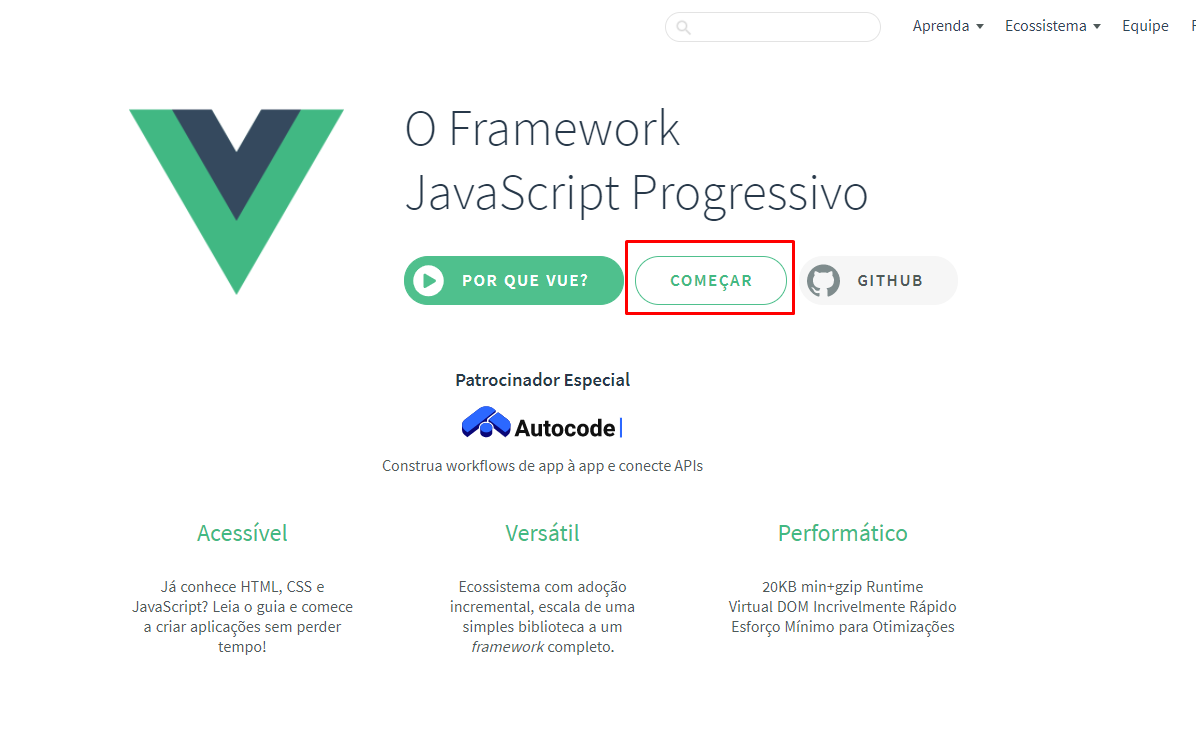


Figura 1 Página VueJS.org

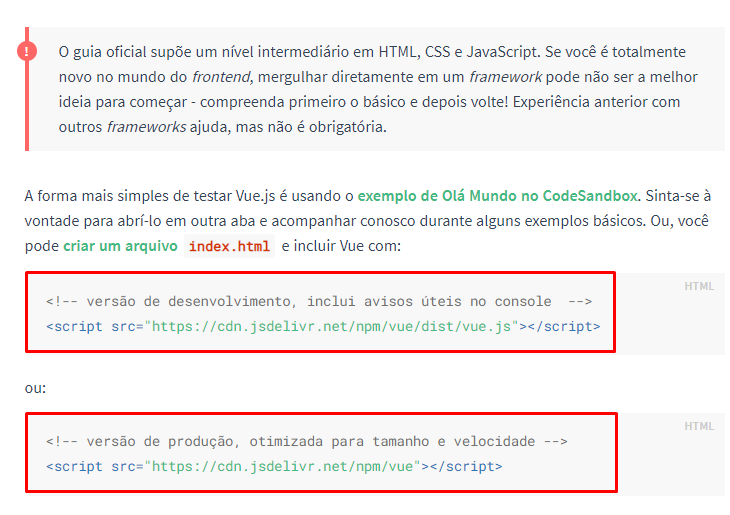


Figura 2 VueJS via CDN

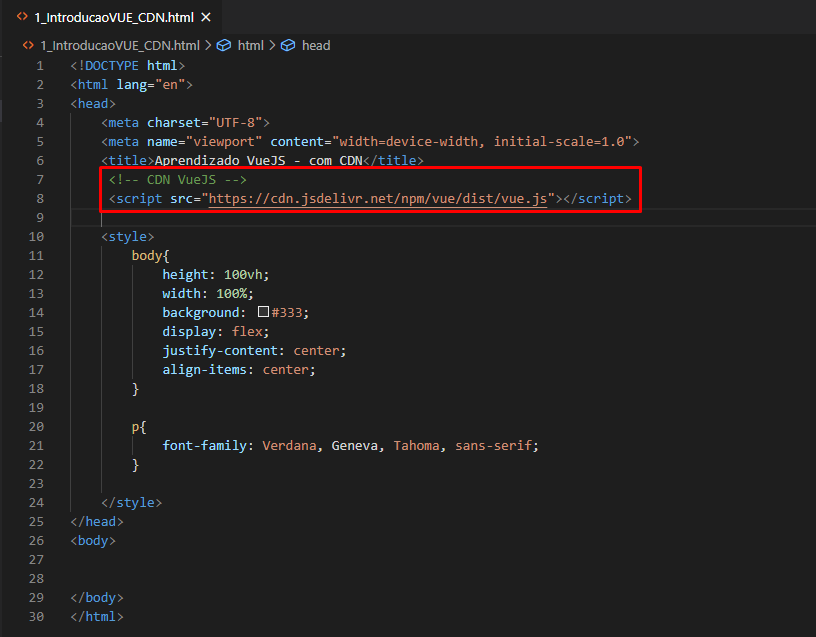


Figura 3 Implementação do CDN

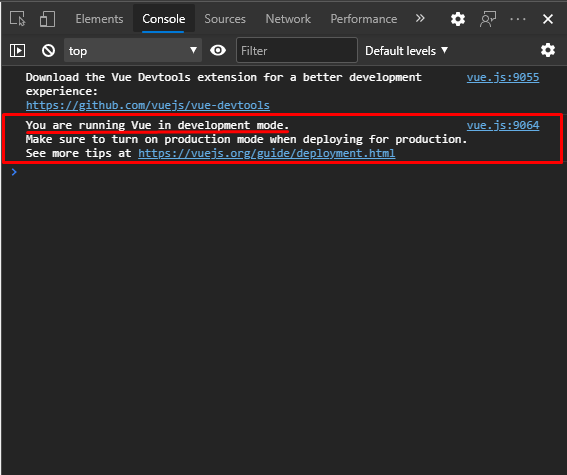


Figura 4 Mensagem Console

# ARQUITETURA inicial VUEJS – com CDN

## INSTÂNCIA VUEJS

Embora não seja estritamente associado com o padrão MVVM, o design do Vue foi parcialmente inspirado por ele. Como convenção, muitas vezes usamos a variável vm (abreviação de ViewModel) para se referir à instância Vue. Uma aplicação Vue consiste em uma instância Vue raiz criada com **new Vue** que, por conseguinte, sendo uma classe, será uma com atributos diversos.

Podemos dizer, e como ficará claro aos poucos, que o Vue preza pela organização da sua arquitetura.

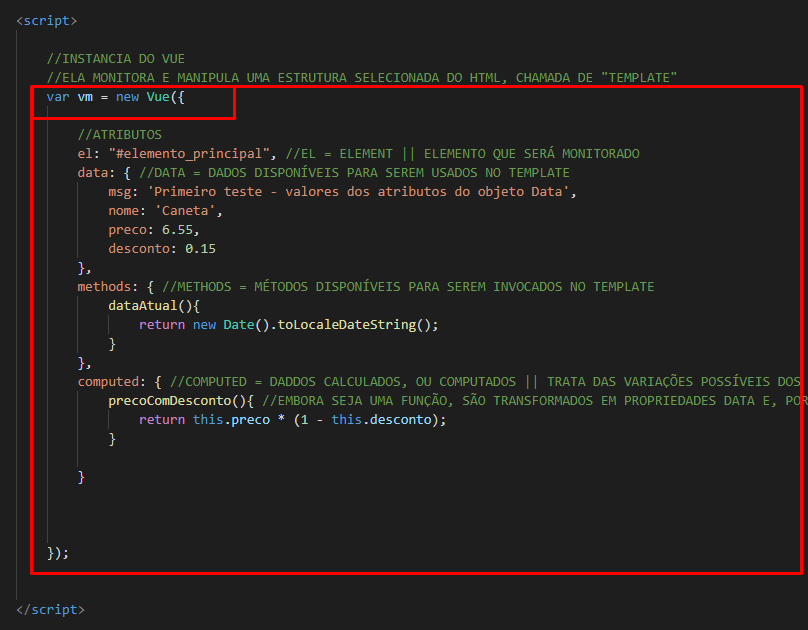


Figura 5 Instanciando o objeto Vue

## ESTRUTURA DE UMA INSTÂNCIA VUE

### TEMPLATE E ATRIBUTO EL

Primeiro, chama-se **Template** o elemento da DOM que é monitorado e manipulado a partir da Vue Instance, com seus métodos e atributos. Este é selecionado, para monitoramento, a partir do **atributo “el”**, que significa “element”.

Assim, o primeiro e principal atributo, de classe, para que o Vue se torne funcional, é o referente ao elemento que será modificado dinamicamente. Se, por exemplo, a estrutura a ser modificada é uma div, de id “elemento\_principal”, o atributo “el” irá receber o valor “#elemento\_principal”.

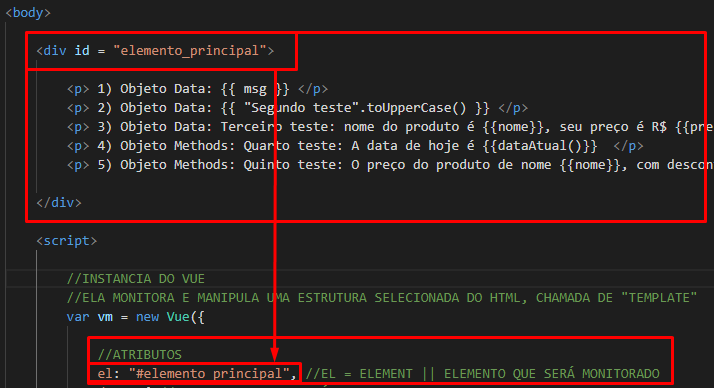


Figura 6 O atributo element el

### ATRIBUTO DATA

O atributo data, da classe Vue, é um objeto que possui atributos referentes ao estado presente do objeto Vue. São estes, que quando modificados, devem representar mudanças dinâmicas no layout do elemento selecionado. Em outras palavras, são os dados a serem utilizados pelo Template.

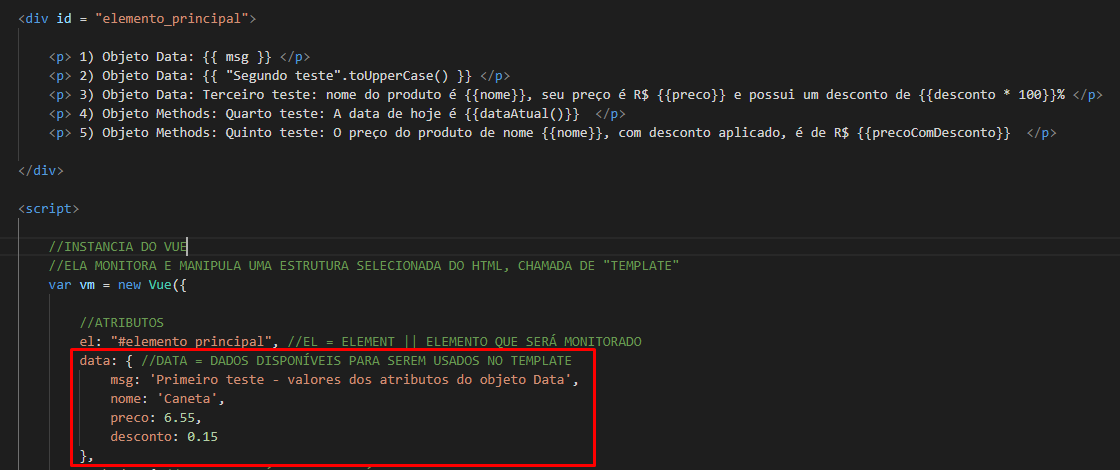


Figura 7 Atributo Data

### ATRIBUTO METHODS E COMPUTED

Tratam-se de métodos, isto é, de funções utilizadas para realizar determinadas tarefas um número necessário ou desejado de vezes. Métodos podem ser chamados incontáveis vezes, executando o mesmo processo.

É difícil distinguir um de outro, de início, mas, embora o Computed possua funções, assim como o Methods, o primeiro é apenas executado quando há uma alteração detectada no estado do objeto, isto é, uma propriedade computada somente será reavaliada quando algumas de suas dependências forem alteradas.

No geral, e para simplificar, na prática o atributo Computed irá conter as funções que irão realizar cálculos e tarefas utilizando os valores do estado atual do Objeto. Assim, se os valores mudarem, naturalmente estas funções serão processadas novamente, para que reprocessem suas tarefas com as novas mudanças. Devido a esta forma de funcionamento, **dados computados são**, por padrão, **getter-only**, isto é, somente recuperados.

Veja abaixo, e note que o método Computed foi utilizado para realizar um cálculo com os valores do estado presente do objeto. Quando chamado uma vez, o retorno será armazenado em um **cache**, e assim, nas próximas vezes que for necessário, será utilizado o valor contido no cache de acesso rápido, ao invés de reprocessar inteiramente a função para obter o mesmo valor. Este cache, por motivos claros, manterá o mesmo valor até o momento em que precisar ser alterado, isto é, quando os valores que determinaram seu valor sofrerem uma alteração.

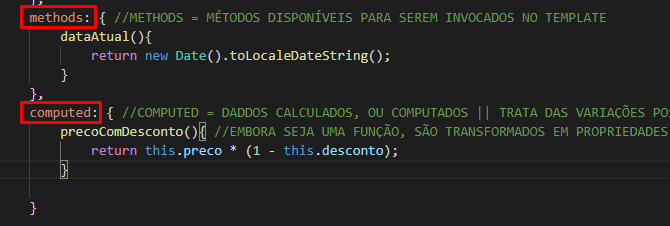


Figura 8 Method e Computed

### ATRIBUTO WATCH – OBSERVADOR

Enquanto dados computados são mais adequados na maioria dos casos, há momentos em que um observador personalizado é necessário. Por isso o Vue fornece uma maneira mais genérica para reagir a alterações de dados, o ***watch***. Isto é particularmente útil quando se precisa executar operações assíncronas ou operações complexas antes de responder a uma alteração de dados.

Vamos comparar melhor ambos, o Computed e o Watch. Como já descrito, dados computados utilizam o Data, isto é, o estado presente do Objeto para realizar cálculos e tarefas, salvam sua resposta em um cache de acesso rápido, e apenas operam novamente quando o valor de cache deve ser alterado. Ou seja, reagem a alterações de forma assíncrona.

Da mesma forma, o observador, ou o Watch, reage a alterações em valores selecionados do Data, mas por meio de uma função cujo nome deve ser igual ao nome do dado modificado. O seu parâmetro será, por padrão, o novo valor do dado. Veja abaixo um exemplo:

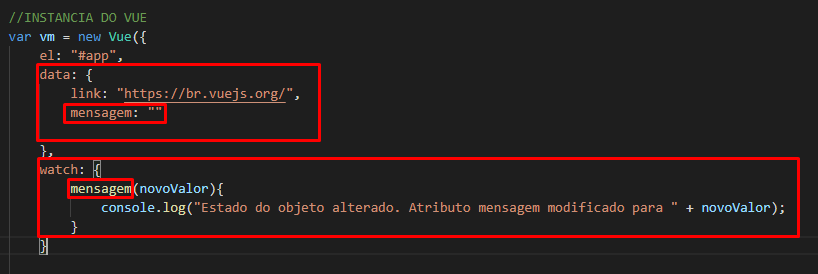


Figura 9 Atributo Watch

No caso acima, a cada modificação no dado “mensagem”, a função do watch é chamada, e imprime no console o novo valor. Mais especificamente, neste caso, é um input com a diretiva v-model, em que a cada digito inserido, o watch reage, e imprime o novo valor. Ou seja, se digitada a palavra “Fulano”, no input, será impresso no console cada letra da palavra, pois cada inserção, até o seu completar, é por si só uma alteração.

Claro que a finalidade acima foi desnecessária, mas são diversas as possibilidades úteis de um observador em uma aplicação que deve ser reativa.

### DIRETIVAS

Diretivas são basicamente como atributos HTML que são adicionados dentro de modelos. **Todos começam com v-**, para indicar que é um atributo especial do Vue. Dentre todas, podemos citar:

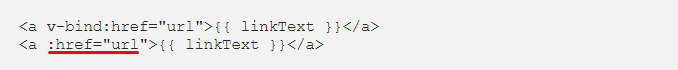
1. **Diretiva v-text:** Em vez de usar a interpolação, você pode usar a diretiva v-text.



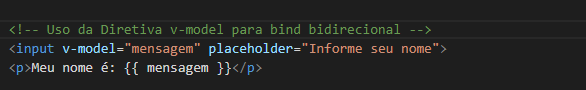
1. **Diretiva v-bind:** A interpolação {{ }} só funciona no conteúdo da tag. Você não pode usá-lo em atributos.



O v-bind é tão comum que existe uma sintaxe abreviada para ele. Ao invés de v-bind:atributo, é possível escrever apenas :atributo. Veja:



1. **Diretiva v-model:** Esta diretiva serve para criar um “Two-Way Binding”, ou seja, uma “ligação bidirecional”; uma atualização assíncrona em duas direções. Funciona perfeitamente com input, checkbox, select, textareae radio, e por isso, é normalmente utilizado com formulários, onde há entrada de dados pelo usuário.



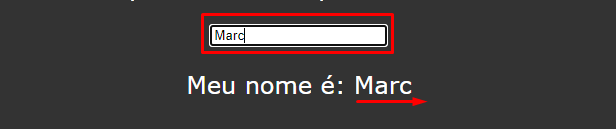
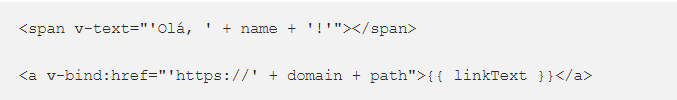


Figura 10 Diretiva v-model

Também é possível usar qualquer expressão JavaScript dentro de uma diretiva:



Dentro de uma diretiva, você pode usar o operador ternário para realizar uma verificação condicional, já que essa é uma expressão. **Existem diretivas dedicadas** que permitem executar condicionais mais organizados: **v-if, v-else e v-else-if**. Ainda, e tão importante quanto, existe a diretiva **v-for** que permite renderizar uma lista de elementos com base nos dados de um Array.

## MANIPULANDO EVENTOS

# BIBLIOGRAFIA

<https://br.vuejs.org/v2/guide/>

https://www.devmedia.com.br/vue-js-tutorial/38042

https://medium.com/tableless/como-criar-e-publicar-uma-biblioteca-em-vue-no-npm-2dff8271ca7d

https://medium.com/vue-js-o-manual-definitivo/aprenda-diretivas-em-7-exemplos-pr%C3%A1ticos-diretivas-vue-js-2-0-a1688bb1839e

https://br.vuejs.org/v2/guide/events.html

https://blog.tiagomatos.com/vue-js-diretiva-v-model/